**Elmi redaktor :** t.e.n., dos. Xalil ismay1lov



**QAFQAZ UNiVERSiTETi**

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

(“Qafqaz” Universiteti)

# Borland

**C++**

# ila Obyektyönlü

**Proqramla§dirma**

**Azarbaycan Respublikasi Tahsil Nazirliyinin 20.10.2006-ci il tarixli, 760 sayli amri ila dars vasaiti kimi tövsiya edilmi§dir.**

## Etibar Seyidzada

**BAKI - 2007**

**Rayçilar :** f.-r.e.d., prof. Faxraddin isayev (“Qafqaz” Universiteti)

t.e.n., Abzetdin Adamov (“Qafqaz” Universiteti)

**Korrektor :** Vafa Seyidova

**Dizayner :** Sahib Kaz1mov

Seyidzada Etibar Vaqif oglu

##### Borland C++ ila Obyektyönlü Proqramla§dirma

 Seyidzada E.V. 2007

**MÜND8RiCAT**

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

##### F8SiL 9

##### OBYEKTYÖNLÜ PROQRAMLA$DIRMA 9

* 1. Proqram Layihalandirma 9
  2. Proqram Xüsusiyyatlari 10
  3. Modul Strukturunun �artlari 14
  4. Obyektyönlü Proqramla§d1rman1n 8saslar1 15

##### F8SiL 25

##### C-D8 YENiLiKL8R V8 C-Y8 8LAV8L8R 25

* 1. Eyni Adl1 Müxtalif Arqumentli Funksiyalar 25
  2. Operatorlar1n Tayini 31
  3. Aktiv Qiymat Vermak 33
  4. Taqdimat (Referans) Tip Tayinedicisi 37
  5. Gizlanmi§ Dayi§kanlari Görmak 41
  6. C++-da Prototiplarin Tayin Edilmasi 43
  7. Struktur Tiplar 44
  8. �arh Operatoru 45
  9. new va delete Opratorlar1 46
  10. inline Makrolar1 49

##### F8SiL. 51

##### OBYEKTL8R 51

* 1. Obyekt Nadir? 51
  2. Layihalandirici 55
  3. Müraciat Haqq1 57
  4. Yoxedici (Destructor) 62
  5. Standart Obyekt Tiplari. 67
  6. Layihalandirici Üzarina Yüklama 67
  7. Obyektlara Manimsatma 74

##### F8SiL 85

##### OBYEKTL8RiN XÜSUSiYY8TL8Ri 85

* 1. Obyekt Üzvlari Olan Obyektlar 85
  2. Friend (Dost) Tayinedicisi 90
  3. Obyeklarin Operatorlara Yüklanmasi 97
  4. this Lokal Dayi§kani 101
  5. Ümumi Ortaq Dayi§kanlar 103
  6. Statik (Static) Funksiyalar 107
  7. const Funksiyalar1 112
  8. iç-iça Tayinlar 112
  9. Obyekt Göstaricilari. 115
  10. Obyekt Massivi 117

##### F8SiL 121

##### OBYEKT TÖR8TM8K. 121

5.1. Töratma 8maliyyat1 121

* 1. Siniflarin Töradilmasi 122
  2. Müraciat Haqlar1 va Nüfuz Etma 127
  3. Dinamik Yüklama 129
  4. Qaydal1 Funksiyalar 133
  5. Misallar 138
     1. Curve. 138
     2. LineDemo 143
  6. C++ Metod Çag1r1§ Sistemi 161

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

* 1. [Mövcud Olandan Töranan Siniflar 164](#_TOC_250039)

1. [F8SiL 169](#_TOC_250038)

[$ABLONLAR HAZIRLAMAQ 169](#_TOC_250037)

* 1. [�ablonlar 169](#_TOC_250036)
  2. [�ablon Funksiyalar 172](#_TOC_250035)
  3. [�ablon Obyektlar 176](#_TOC_250034)

1. [F8SiL 181](#_TOC_250033)

[AXINLAR 181](#_TOC_250032)

* 1. Ax1n Nadir? 181
  2. Standart Ax1nlar. 181
  3. Ax1nlara Nizamlanm1§ Malumat Yaz1lmas1 185
     1. [Geni§lik Nazarati 185](#_TOC_250031)
     2. [Yerla§ma Nazarati 187](#_TOC_250030)
     3. [Bo§luq Nazarati 188](#_TOC_250029)
     4. Tam 8dadlarin 8saslar1na Nazarat. 189
     5. [Haqiqi 8dadara Nazarat 190](#_TOC_250028)
  4. Ax1nlardan Nizamlanm1§ Malumat Oxunmas1 192 7.5 Sahvlara Nazarat. 195
  5. Fayl Üzarindaki Ax1nlar. 195
     1. [Fayla Yazma 196](#_TOC_250027)
     2. [Fayldan Oxuma 200](#_TOC_250026)
  6. Obyektlar va Ax1nlar 202

1. [F8SiL. 211](#_TOC_250025)

[CLASS KiTABXANASI 211](#_TOC_250024)

* 1. Container Class Kitabxanas1 211
  2. [Tayin Olunmu§ Siniflar 212](#_TOC_250023)
  3. [Tayinlar va Tiplar 214](#_TOC_250022)
     1. Tip va Sinif Kodlar1 214
     2. Sahv Kodlar1n1n Tayini. 216
     3. Ba§l1q Fayllar1 va Tayin Edilmi§ Siniflar 217
  4. [Siniflar. 218](#_TOC_250021)
     1. [Object 218](#_TOC_250020)
     2. [Error 227](#_TOC_250019)
     3. [Sortable 228](#_TOC_250018)
     4. [String 229](#_TOC_250017)
     5. [BaseDate 232](#_TOC_250016)
     6. [Date 235](#_TOC_250015)
     7. [BaseTime 239](#_TOC_250014)
     8. [Time 242](#_TOC_250013)
     9. [Association 244](#_TOC_250012)
  5. [Malumatlar Sturukturu Siniflari 247](#_TOC_250011)
     1. [Container 247](#_TOC_250010)
     2. [Stack 251](#_TOC_250009)
     3. [Deque 255](#_TOC_250008)
     4. [Queue 258](#_TOC_250007)
     5. [PriorityQueue 259](#_TOC_250006)
     6. [Collection 261](#_TOC_250005)
     7. [List 263](#_TOC_250004)
     8. [DoubleList 265](#_TOC_250003)
     9. [HashTable 268](#_TOC_250002)
     10. [Btree 270](#_TOC_250001)

[8.5.11 Bag 277](#_TOC_250000)

8.5.12 Set 278

* + 1. Dictionary 278
    2. AbstractArray 279
    3. Array 283
    4. SortedArray 285
  1. Yenilayicilar (Iterators) 290

8.6.1 DoubleListIterator 293

* 1. Misal 294

**ÖN SÖZ**

Bu kitabda obyektyönlü proqramla§d1rman1n asaslar1, Borland C++ proqramla§d1rma dilinin xüsusiyyatlari §arh edilmi§dir.

Vasaitdan obyektyönlü proqramla§d1rmag1 öyranmak istayan talabalar, müallimlar, hamçinin proqramç1lar faydalana bilarlar. Vasait mümkün qadar sada dilda yaz1lm1§d1r. Kitabdak1 mövzular misallarla mü§ayiat olunmu§dur ki, bu da mövzunu asan manimsamaya kömak edir.

Onu da qeyd edak ki, kitab respublikam1zda bu mövzuda azarbaycan dilinda yaz1lm1§ ilk vasaitdir. Buna göra da kitabda bir çox yeni terminlarin istifadasinda

<atinliklar qar§1ya ç1xm1§ va bu çatinliklarin aradan qald1r1lmas1na cahd göstarilmi§dir.

Kitab1n haz1rlanmas1nda laz1mi §arait yaratd1g1na göra Qafqaz Universitetinin rahbarliyina, dayarli maslahatlarina göra f.-r.e.d., professor Faxraddin isayeva, t.e.n., dossent Xalil ismay1lova, t.e.n. Abzetdin Adamova tashih etdiyina göra Vafa Seyidovaya, dizayner Sahib Kaz1mova, kitab1n çap1nda göstardiklari dastaya göra Mü§fiq ibrahimova va Sadi 8bdürrahmanova öz samimi ta§akkürümü bildiriram.

##### Müallif

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

# I F8SiL

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

## OBYEKTYÖNLÜ PROQRAMLA$DIRMA

### Proqram Layihafandirma

Kompüter texnologiyas1n1n paralel inki§af edan iki asas sahasi vard1r: **texniki vasitafar** (hardware) va **proqram taminati** (software). Texniki vasitalar na qadar süratla inki§af etsa da, proqram taminat1 ila tachiz edilmadan heç bir faydas1 olmaz. Proqram1n texniki vasitalar olmadan i§lamasini da dü§ünmak olmaz. Lakin yax§1 texniki vasita olmazsa, uygun olaraq yax§1 proqram da yaz1lmaz. istifadaçilarin istadiklarini texniki vasitalarla yerina yetirmalari üçün isa har zaman yax§1 proqram taminat1na ehtiyac vard1r.

Proqram taminat1 çox ahamiyyatli olan, lakin bunun qiymati çox gec ba§a dü§ülan bir mövzu olmu§dur. Yax1n keçmi§da kompüter ehtiyac1 olan §axslar i§larini göra bilacak proqram, sonra da bu proqram1 i§ladacak bir kompüter axtar1r va proqram1 kompüterin ayr1lmaz bir xüsusiyyati kimi görürdülar. Bu bir növ kompüter sat1c1lar1n1n ehtiyac duyulan proqramlar1 özlarinin

yazmalar1ndan va öz kompüterlari xaricinda bu proqram1 satmamalar1ndan qaynaqlan1rd1. Haz1rda isa müstaqil proqramla§d1rma ila ma§gul olan firmalar1n qurulmas1 bu sahv dü§üncani aradan qald1rm1§d1r. Bu hamçinin proqram taminat1 va texniki vasitalarin bir-birindan farqli bir §ey olduqlar1 haqiqatinin anla§1lmas1n1 tamin etmi§dir.

### Proqram Xüsusiyyatlari

* + - * **Dogruluq (correctness) –** verilan tap§1r1qlar1n tam olaraq yerina yetirilmasidir. Proqram1 layihalandirmadan avval onun hans1 tap§1r1qlar1 yerina yetiracayini müayyan etmak laz1md1r. Proqram haz1r olduqdan sonra bu tayin olunan xüsusiyyati tam tamin etmalidir;
      * **Dayaniqliq (robustness) –** gözlanilmaz hadisalar naticasinda proqram1n icras1 kasilmamali, sahv amaliyyatlar1 yerina yetirmamalidir. Proqram, an yax§1 halda olsa bela, üzarina qoyulan tap§1r1qlardan ba§qa i§lari görmamalidir. Proqramç1n1n sahvlarina göra proqram1n icras1n1n kasilmamasi üçün tadbirlar görülmalidir;
      * **Geni§fanabilmak (extendibility) –** galacakda verilan tap§1r1qlar1n dayi§dirilmasi va ya

yenilarinin alava edilmasi asan olmal1d1r. *Bunun üçün:*

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

* sada layihalar haz1rlanaraq mürakkab layihalardan qaçmaq laz1md1r (design simplicity);
* proqram1 bir markazdan as1l1 olaraq idara etmak avazina modul sturukturundan istifada edarak yerli bir idara etma formas1 seçilmalidir (decentralization).
  + - * **Takrar istifada olunma (reusability) –** haz1rlanan layihanin, yaz1lan proqram1n va ya heç olmazsa modullar1n ba§qa proqramlar tarafindan istifada edila bilmasidir. Buna layiha daxilinda istifada edilan elementlarin yeni layihada da istifada edila bilmasini alava etmak laz1md1r;
      * **Uygunluq (compatibility) –** proqram1n müxtalif kompüter sistemlarinda ortaq xüsusiyyatlara malik olmas1d1r. *Bunun üçün müxtJlif standartlarzn tJtbiq edilmJsi lazzmdzr:*
        + malumatlar fayl1 format1n1n uygunlugu;
        + malumatlar strukturunun uygunlugu.
      * Menyu, dialoq, rasm, düyma kimi **istifadaçi mühitinin** (user interface) uygunlugu;
      * **Manbafarin istifada edilmasi (efficiency) –** kompüterin malik oldugu bütün avadanl1qlar1 samarali §akilda tam istifada etmasidir. istifada eda bilmadikda da digar proqramlar üçün istifadasiz qalmas1na yol vermasidir;
      * **Da§inabilma (portability) –** bir proqram haz1rlanm1§ oldugu kompüterdan ba§qa digar kompüterlarda da istifada oluna bilmalidir. Bu iki formada ola bilar:
        + **Qaynaq uygunlugu (source compatible) –** proqram1n yaz1ld1g1 amaliyyat sistemindan ba§qa bir sistema da§1n1b yenidan komplyasiya olunaraq i§lamasi;
        + **ikilik kod uygunlugu (binary compatible) –** proqram1n yaz1ld1g1 mühitda komplyasiya olunaraq **icra oluna bilan fayl** (executable file) alda edildikdan sonra ba§qa bir mühita da§1naraq i§ladilmasi. Proqram1n takmilla§dirilmasi bax1m1ndan asli asas götürülarak proqram kodunun da§1nabilan olmas1d1r.
      * **Nazarat oluna bilma (verifiability) –** bir proqram1n sahv hallarla qar§1la§mas1 zaman1 onun icras1n1n davam etmasina va hatta heç icra olunmamas1na sabab olan sahvlar ortaya ç1xd1g1 zaman istifadaçiya va proqramç1ya sahvin hans1

sababdan ba§ verdiyi mövzusunda kifayat qadar

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

malumat1n verilmasidir;

* + - * **Tamliq (integrity) –** icazasiz müdaxilalar va dayi§dirmalar qar§1s1nda proqram, malumatlar, fayl kimi proqram komponentlarinin qorunmas1d1r. Masalan, malumatlar va ya indeks fayllar1n1n itmasi (silinmasi) zaman1 proqram icra olunarkan bunu müayyanla§dirib bildirir;
      * **Asan istifada edilma (easy of use) –** proqramdan istifada edan §axsin onu asanl1qla öyranmasi, istifada eda bilmasi, naticalarini tadqiq eda bilmasi, sahvlarini düzalda bilmasidir;
      * **Birlikda i§famak (interoperability) –** bir proqram1n ehtiyac1 oldugu ba§qa bir proqram1 çag1ra bilmasi xüsusiyyatidir. Bu halda iki proqram ard1c1l olaraq i§lamakla barabar bir-biri ila malumat mübadilasi eda bilmalidirlar.

Yuxar1da göstarilan §artlari tamin etmak asasan proqramç1 mühandisin vazifasi olmaqla barabar, modul strukturundan istifada etmak *geni§fJnJ bilmJ*, *tJkrar istifadJ olunma*, *uygunluq*, *da§zna bilmJ* problemlarini hall etmaya imkan verir. Bu halda an az1 bazi proqram modullar1n1 yenidan yazmaga ehtiyac qalm1r.

### Modul Strukturunun $artlari

* **Parçalanabilma (modular decomposability) –** bir problemi alt hissalara ay1raraq layihalandirmakdir. Masalan, riyazi amaliyyatlar1n yerina yetirildiyi bir proqramda massivlarin istifada edilmasi üçün bir massiv modulu tayin edarak massivla alaqadar amaliyyatlar1n ham1s1n1 bu modulda yazmaq. Eyni §akilda ehtiyac olarsa, matris, vektor, kompleks adad kimi tayinlar üçün da modul yazaraq problemi kiçik hissalara ay1rmaq;
* **Birla§dirilabilma (modular composability) –** bir-birindan xabarsiz haz1rlanan modullar1n bir yera y1g1lmas1 zaman1 çat1§mayan va ya tam olmayan modullar1n olmamas1d1r;
* **Aydinliq (modular understandability) –** müxtalif §axslar tarafindan yaz1lmas1na baxmayaraq oxundugu zaman proqram1n ayd1n olmas1d1r. Proqram1n yenidan bax1lmas1 va ya takmilla§dirilmasi zaman1 çox ahamiyyatli olan bu xüsusiyyati saxlamaq üçün modul proqram1n1n yaz1lmas1ndan ba§qa proqram daxilinda nayin na üçün istifada edildiyinin, bu istifada naticasinda na olacag1n1n ayd1n bir

§akilda §arh olunmas1 laz1md1r;

* + **Qorunma (modular protection) –** bir modulun i§lama formas1na va malumatlara digar bir modulun icaza verilmi§ hallardan ba§qa müdaxila etmamasi, yaz1lan modullar1n ümumi cahatlari olmas1na baxmayaraq bunlar1n bir- birindan farqlandirilmasi laz1md1r;

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

* + **Davamlifiq (modular continuity) –** problemin tayinindaki kiçik dayi§ikliklar bir va ya bir neça hissanin dayi§masina sabab olarkan proqram sturukturu kimi istifada olunan modullar aras1ndak1 vasitalar (masalan, funksiya prototiplari) dayi§dirilmamalidir.

indi da modullulugun tamin olunmas1 üçün istifada olunacaq obyektyönlü proqramla§d1rman1n xüsusiyyatlarini gözdan keçirak.

### Obyektyönlü Proqramla§dirmanin

**8saslari**

Bir sistem daxilinda müxtalif xarakterli obyektlar ola bilar. Bu obyektlarin tamamila bir-birindan farqli xüsusiyyatlari ila barabar, eyni va ya ox§ar xüsusiyyatlari da vard1r. Xüsusiyyatlar va davran1§lar1n1n müxtalif olamalar1na baxmayaraq eyni imkanlara malik ola bilarlar.

Masalan, bir idarada kompüter, telefon, faks, katiba, mamur, müdir kimi obyektlar ola bilar. 8sasan bunlar1n har biri bir obyektdir. Har birinin öz funksiyas1 vard1r. Lakin ümumi xüsusiyyatlari da vard1r. Katiba, mamur, müdir har biri bir insand1r. Bu insan olma xüsusiyyatidir. Har biri müassisada müxtalif mablagda maa§la i§layirlar. Gördüklari i§ ümumi bir i§dir, lakin har birinin öz i§i vard1r. istak eyni, davran1§lar isa müxtalifdir. i§larini icra edarkan istifada etdiklari malumatlar da eyni daracada farqlidir.

Bu banzatma ila *obyekt - müJyyJn i§fJri yerinJ yetirJn, bu mJqsJdlJ dJ müxtJlif funksiyalardan ibarJt olan bir strukturdur*. Bu struktur daxilinda dayi§kanlar ola bilar. Lakin asasan vazifasini müayyan edacak funksiyalar1 tarkibinda saxlay1r. Bu xüsusiyyata **paketla§dirma** (encapsulation) deyilir.

Bunun digar bir xüsusiyyati da paketla§dirilcak funksiyalar1n neca i§layacayi müayyan edilmadan, sadaca neca istifada edilacayi müayyan edila bilar ki, bu da vacibdir. Buna da **mücarradla§dirma** (abstraction) deyilir. Paketla§dirma va mücarradla§dirma, obyekti müayyan etmak üçün kifayat olan iki funksiyad1r.

Obyeklarin digar xüsusiyyatlarindan biri da törama xüsusiyyatidir. Obyekt tayin edilarkan, avvalcadan tayin olunmu§ ba§qa bir obyekti özüna baza olaraq seça bilar. Bu hal yeni tayin olunan obyektin özüna baza seçdiyi

obyektin xüsusiyyatlarindan istifada etma imkanlar1na uygun galir, bununla barabar yeni obyektin baza obyekti ila eyni xüsusiyyat da§1mas1na sabab olur.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Yeni obyekt yeni xüsusiyyatlar qazana bildiyi kimi, bu xüsusiyyatlari takmilla§dira va dayi§dira da bilar. Bu xüsusiyyata **törama** (derivation), xüsusiyyatlari almaga isa **miras alma** (inheritance) ad1 verilir.

Buna a§ag1dak1 misal1 göstara bilarik:

Bir avtomobil tamirçisi avtomobil ranglama (hava f1rça ila), lehimlama va karbürator sazlama i§larini yerina yetirmi§ olsun.

Avtomobil Tamirçisi

* R�ngl�m�
* Lehiml�m�
* Karbürator Sazlama

Bu avtomobil tamirçisi obyektidir. Bu avtomobil tamirçisinin üç oglu oldugunu va onlar1n har birini yeti§dirdikdan sonra, bir saha üzra ixtisasla§d1rd1g1n1 farz edak.

Avtomobil T�mirçisi

Avtomobil R�ngsaz1

* R�ngl�m�
* Lehiml�m�
* Karbürator Sazlama

- R�ngl�m� (Hava f1rças1 il�)

Avtomobil Qaynaqç1s1

- Lehiml�m� (Oksigen qaynag1 il�)

Karbürator Sazlay1c1s1

- Elektrik i�l�ri

Burada Avtomobil Rangsaz1, Avtomobil Qaynaqç1s1 va Avtomobil Elektriki törama obyektlardir. Avtomobil Tamirçisi isa baza obyektidir. Törama obyektlar (Avtomobil Rangsaz1, Avtomobil Qaynaqç1s1 va Avtomobil Elektriki) baza obyektinin, yani Avtomobil Tamirçisinin xüsusiyyatlarini göstaracaklar. Uygun olaraq avtomobil ranglayacak, lehimlayacak va karbürator tanzimlayacaklar.

Avtomobil Rangsaz1ndan avtomobili ranglamasini istadiyimiz zaman, o avtomobili hava f1rças1 ila ranglayacakdir. Lakin Avtomobil Tamirçisi bu i§i yaln1z adi f1rça ila göracakdi. Avtomobil Tamirçisi ila Avtomomobil Rangsaz1n1n gördüklari i§ ranglama i§idir.

Avtomobil Rangsaz1n1n adi f1rça ila ranglamayi bacarmas1na baxmayaraq hava f1rças1 ila ranglayir. Lakin istadiyi zaman adi f1rça ila da raglaya bilar.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Eyni §akilda Avtomobil Qaynaqç1s1 da oksigen qaynag1 ila lehimlayir. Avtomobil Tamirçisi isa sadaca qövs qaynag1 ila lehimlaya bilir. Avtomobil Qaynaqç1s1 eyni zamanda ranglama i§larini da bacar1r. Bu qabiliyyati Avtomobil Tamirçisindan miras alm1§ va sadaca f1rça ila ranglamayi bacar1r.

Ücüncü qolu ta§kil edan Avtomobil Elektriki isa tamamila ba§qa bir xüsusiyyata malikdir. Avtomobil Tamirçisi elektrik i§larini göra bilmadiyi halda Avtomobil Elektriki bu i§lari göra bilir. Avtomobil Elektriki eyni zamanda ranglama, lehimlama va karbürator tanzimlama i§larini Avtomobil Tamirçisindan öyrandiyi qadar göra bilir. Tabii ki, agar laz1m galarsa, Avtomobil Elektrikinin masalan, ranglama bacar1g1n1n lagv edilmasi uygun görülarsa, heç bir i§ görmamasi tamin oluna bilar.

Burada növbati üç xüsusiyyat nazari calb edir:

1. Töranmi§ obyektlar baza obyektinin xüsusiyyatlarini qoruyub saxlayaraq istifada eda bilirlar;
2. Töranmi§ obyektlar törandiklari obyektlarin (baza obyektlarinin) xüsusiyyatlarini dayi§dira bilirlar;
3. Töranmi§ obyektlar yeni xüsusiyyatlar qazana bilirlar.

Bir sistem daxilinda istar bir obyektdan töranmi§ obyektlar olsun, istarsa da bir-birindan farqli obyektlar olsun, bu obyektlarin ox§ar xüsusiyyatlari ola bilar va bu xüsusiyyatlar eyni adla verilir. Bu da eyni adl1, lakin müxtalif obyektlarin üzvü olan obyektlarin meydana galmasina sabab olur. Bu hal obyektlar aras1ndak1 ox§arl1qlar1 göstarir. Buna **ox§arliq** (polymorphism) deyilir. 8slinda bir obyektdan taranan siniflar aras1nda ox§arl1g1n olmas1 vacibdir.

Obyektyönlü proqramla§d1rmada obyektlarin malik olduqlar1 malumatlar1 va funksiyalar1 qoruyaraq, birba§a istifada etmaya icaza vermamalari, ba§qa bir alt xüsusiyyatdir. Burada miras qoymag1n aksi olan bir amaliyyatdan söhbat gedir. Bir obyekt bazi xüsusiyyatlarini saxlay1b sadaca özü istifada edir. Digar obyektlarin istifada etmasina icaza vermaz va ya mahdudla§d1rar. Bu, dört müxtalif hal ila §arh edila bilar:

1. **Xüsusi (private) –** bir üzv malumat1n va ya funksiyan1n yaln1z üzvü oldugu obyekt daxilinda istifada olunmas1;
2. **Qorunmu§ (protected) –** bir üzv dayi§kaninin va ya funksiyas1n1n üzvü oldugu obyekt xaricinda o obyektdan töranan obyektlarin sadaca istifada eda bilmasi;
3. **Ümumi (public) –** bir üzvün bütün obyektlar

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

tarafindan ortaq istifada edilmasi;

1. **Dost (friend) –** bir obyektin ba§qa bir obyekti dost elan edarak üzvlarinin ham1s1n1n bu obyekt tarafindan istifadasina icaza vermasi.

Obyektyönlü proqramla§d1rmada digar asas xüsusiyyat isa **dinamik alaqafandirmadir** (dinamic binding). Bu xüsusiyyatla töranan bir obyektin ünvan1n1 törandiyi obyektlardan birinin **göstarici** (pointer) dayi§kanina manimsatmak mümkündür. Bu halda ünvan1 manimsadilan obyekt, göstaricisina manimsadildiyi baza obyekti kimi davranacaq, hamçinin özünamaxsus xüsusiyyatlarini da göstaracakdir.

Bu hal1 bela §arh etmak olar: masalan, bir müassisanin bax1§ bölmasina bir avtomobil tamirçisi i§a al1nacaqd1r. Bu i§a (göstarici dayi§kan) Avtomobil Tamirçisindan ba§qa Avtomobil Rangsaz1, Avtomobil Qaynaqç1s1 va Avtomobil Elektriki da müraciat eda bilar va bu i§a al1na bilar. Çünki, bunlar1n kökünda avtomobil tamirçiliyi durur. Malumdur ki, bu i§a al1nacaq §axsin avtomobil ranglama, lehimlama va karbürator tanzimlama i§larindan ba§1 ç1xacaqd1r. Bu i§i üç mütaxassis birlikda da göra bilar.

indi da bu i§a Avtomobil Qaynaqç1s1n1n al1nd1g1n1 farz edak. Bu halda tamirçidan ranglamak talab olunarsa, o f1rçadan istifada edarak ranglama i§ini yerina

yetiracakdir. Çünki tamirçi kimi i§a götürülan Avtomobil Qaynaqç1s1, Avtomobil Tamirçisindan öyrandiyi (miras ald1g1) f1rça ila ranglamayi bacar1r. Lakin lehimlamak talab olundugunda, oksigen qaynag1 ila lehimlayacakdir. Çünki asas bacard1g1 i§ da ela budur.

8gar bu i§a Avtomobil Elektriki al1nsayd1, qaynaq i§larini yaln1z qövs qaynag1 ila göracakdi. Çünki, miras alma yolu ila öyrandiyi lehimlama i§i budur. Bu, ranglama va karbürator tanzimlama i§lari da ola bilar. Avtomobil Elektrikinin bildiyi daha bir §ey vard1r ki, bu da elektrik i§laridir. Lakin tamirçi kimi i§a al1nan Avtomobil Elektrikindan bu i§i görmasi talab olunmaz. Avtomobil Elektriki i§a al1narkan görülacak i§lar aras1nda elektrik i§lari yoxdur.

Bir obyektin baza obyektinin göstaricisina manimsadilarak baza obyekti xüsusiyatlari göstarmasina ***dinamik alaqalandirma*** deyilir. Ümumi maqsadli alqoritmlarin (s1ralama, axtarma kimi) tatbiqinda va ya eyni xüsusiyyatli müxtalif xarakterli obyektlarin birga istifadasinda istifada oluna bilar. Yaln1z bu xüsusiyyat üçün töratma amaliyyat1n1n olmas1n1n vacibliyina diqqat edin.

Töranma xüsusiyyati ila alaqadar olaraq daha bir xüsusiyyat obyektin bir obyektdan deyil, bir neça obyektdan töranmi§ olmas1d1r. Buna **çoxbazalifiq** va ya **çoxdan töranma** (multi inheritance) deyilir. Burada obyekt

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

törandiyi bütün obyektlarin xüsusiyyatlarini göstarir. Masalan, idaraçi va Avtomobil Tamirçisi obyektlarindan töranan Müdir obyekti ham idaraçi, ham da Avtomobil Tamirçisi kimi davrana bilir. Yani ranglaya, karbürator tanzimlaya, i§çi al1b maa§ vera bilir. Bu xüsusiyyatlari dayi§dira va ya yeni xüsusiyyatlari tarkibina ala bilir.



Çox bazal1 obyektlarin ahamiyyati dinamik alaqalandirmada öz aksini tap1r. Çoxbazal1 obyekt törandiyi bütün obyektlarin göstaricilarina manimsadila bilar. Bu, manimsatma naticasinda sadaca manimsadildiyi obyektin xüsusiyyatlarini göstarir. Masalan, Müdir obyekti Avtomobil Tamirçisi göstaricisina manimsadilarsa, Avtomobil Tamirçisi kimi davran1r. Yox agar idaraçi göstarcisina manimsadilarsa, idaraçi kimi davran1r.

Obyektyönlü proqramla§d1rman1n asas anlay1§lar1ndan biri da proqramlar daxilinda müayyan qaliblarin (§ablonlar1n) haz1rlan1b bir neça dafa istifada oluna bilmasidir. Bu xüsusiyyata **§ablonlama** (template) deyilir. �ablonlar yaz1lm1§ bir proqram kodunun ox§ar hallar üçün istifada edilmasini ifada edir. Masalan, tam adadlardan ibarat olan bir massivi s1ralamaq üçün proqram kodunun eyni zamanda haqiqi adadlari, satirlari va hatta yeni töradilacak obyektlari da s1ralaya bilacak bir

§akilda yaz1la bilmasi §ablonlama xüsusiyyatidir.

# II F8SiL

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

## C-D8 YENiLiKL8R V8 C-Y8

**8LAV8L8R**

### Eyni Adli Müxtalif Arqumentli Funksiyalar

C va digar yüksak saviyyali proqramla§d1rma dillarinda arqument kimi daxil edilmi§ eyni tipli iki qiymatdan an böyüyünü geri qaytaran bir funksiya yazmaq üçün növbati üsullardan istifada etmak olar: Birinci üsul - istifada olunan an böyük aral1ql1 tipa göra bir funksiya yaz1b, alt tiplari bu tipa çevirmakdan ibaratdir. Masalan,

long double Max(long double A, long double B)

{ return A < B ? B : A; }

va

long double LD; double D;

int I; char C;

tayin edildikdan sonra

LD = Max(12, 18);

D = (double)Max(12, 18); I = (int)Max(12, 18);

C = (char)Max(‘C’, ‘H’);

tip çevirma operatorlar1 ila yerina yetirila bilar. Bela ki,

I = (int)Max(12e7, 18.36);

amaliyyat1 heç bir sahv göstarmadan komplyasiya olunur. Lakin icra olunarkan müayyan xatalar ba§ vera bilar. Satirlarin an böyüyünün tayin edilmasi üçün bu funksiyadan istifada etmak olmaz. Bu halda komplyator sahvlari ola bilar. Bu cür sahvlar ahamiyyatsiz say1lsa da, proqram sahv icra olunur.

ikinci üsul - har tip üçün ayr1 bir funksiya yazmaqdan ibaratdir. Bu halda C qaydalar1na uygun olaraq har bir funksiya üçün ayr1 addan istifada etmak laz1m galacakdir.

#include <string.h>

char Max\_char(char A, char B)

{ return A < B ? B : A; }

int Max\_int(int A, int B)

{ return A < B ? B : A; }

double Max\_double(double A, double B)

{ return A < B ? B : A; }

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

char \*Max\_string(char\* A, char\* B)

{ return strcmp(A, B) < 0 ? B : A; }

Bu cür tayinlar bütün tiplar üçün yaz1la bilar. Bu dafa yuxar1da göstarilan tayina uygun olaraq

D = Max\_double(12, 18); I = Max\_int(12, 18);

C = Max\_char(12, 18);

§aklinda olmal1d1r. Bu halda tip çevirma amaliyyat1n1n aradan qalxm1§ olduguna diqqat edin. Bunun avazinda yaln1z funksiya ad1 dayi§mi§dir. Bu üsulla

I = Max\_int(12e7, 18.36);

kimi amaliyyatlar1 avvalcadan yerina yetirmak da mümkündür. Böyük funksiyalar1n makro saviyyada tayin edilmasinin proqram1n böyüklüyünü art1rd1g1n1 nazara ald1qda, bu an yax§1 üsul say1la bilar. Bu zaman hans1 tip üçün hans1 funksiyadan istifada edilacayini çox yax§1 bilmak laz1md1r.

Lakin C++-da bir funksiya eyni adla bir neça dafa tayin oluna bilar. Har tayinda funksiya digarindan farqli olaraq müstaqil bir funksiya kimi i§layir. Bu hadisaya **üzarina yüklama** va ya **üst-üsta qoyma** (overloading) ad1 verilir. Bir ad üzarina iki va ya daha art1q ad yüklanacaksa, bu

overload *funksiya\_adt*;

§aklinda göstarilir. Bu §akilda tan1mlanm1§ har funksiyan1n arqument siyah1s1ndak1 tip ard1c1ll1g1 daha avval tayin edilmi§ eyni adl1 funksiyalar1n arqument siyah1s1ndak1 tip ard1c1ll1g1 ila eyni olmal1d1r.

Buna göra Max funksiyas1n1 a§ag1dak1 kimi tayin etmak olar:

// MAXDEC.CPP

#include <string.h>

#include <stdio.h> overload Max;

int Max(int A, int B)

{ return A < B ? B : A; }

char Max(char A, char B)

{ return A < B ? B : A; }

double Max(double A, double B)

{ return A < B ? B : A; }

char \*Max(char\* A, char\* B)

{ return strcmp(A, B) < 0 ? B : A; }

Bu tayindan sonra

C = Max('C', 'H'); I = Max(12, 18);

// char C

// int I

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

manimsatma amaliyyatlar1n1 yerina yetirmak olar. Bu manimsatmalarin yerina yetirilmasindan sonra har Max funksiyas1 digarlarindan farqlanir.

D = Max(12.4, 18.9); // double D S = Max("String", "Massiv"); // char \*S

Bela ki,

|  |  |
| --- | --- |
| char C = Max('C', 'H'); | char Max(char, char); |
| int I = Max(12, 18); | int Max(int, int); |
| double D = Max(12.4, 18.9); | double Max(double, double); |
| char \*S = Max("String", "Massiv"); | char Max(char\*, char\*); |

prototipli funksiyalar çag1r1l1r. Hans1 funksiyan1n çag1r1lacag1n1n tayin edilmasinda funksiyan1n ad1ndan ba§qa, arqument siyah1s1ndak1 tiplar va tiplarin ard1c1ll1g1 asas olmal1d1r. Funksiyan1n qaytard1g1 tip isa bir natica olaraq ortaya ç1x1r. Masalan,

int *funksiya*(char);

char *funksiya*(int);

int *funksiya*(int, int);

parametr siyah1s1ndak1 müxtalifliya göra bir-birindan

farqli olan üç funksiyan1 tayin edarkan,

int *funksiya*(char);

char *funksiya*(char);

parametr siyah1s1ndak1 tip ard1c1ll1g1 eyni oldugu üçün eyni funksiyan1 tayin edir. Qaytar1lan tiplar müxtalif olduguna göra sahv a§kar edilir va bu cür tayin qabul edilmir.

Üst-üsta yüklama ila alaqadar tayin etma amaliyyatlar1 yerina yetirilarkan tip tayinedicilarinda müxtalif tiplarin olduguna diqqat edilmalidir. Yani

int *funksiya*(int);

int *funksiya*(unsigned int);

kimi iki tip tayin edilarkan

int *funksiya*(int);

int *funksiya*(signed int);

eyni funksiyan1 iki dafa tayin etma manas1na galdiyi üçün sahv verir. Eyni §akilda

int *funksiya*(int);

int *funksiya*(const int);

bir-birindan farqli iki tayin olmas1na baxmayaraq, const yeni tip ta§kil etmadiyi üçün birlikda istifada edila bilmazlar.

Yeni tayin edilmi§ tiplardan istifada edarkan funksiyalar üzarina yüklama aparmaq olar. Masalan,

struct Tarix {

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

int gun, ay, il;

};

tayinindan sonra

struct Tarix Max(struct Tarix A, struct Tarix B)

{ if( A.il == B.il ) if(A.ay == B.ay )

return A.gun >= B.gun ? A : B; else return A.ay >= B.ay ? A : B;

else return A.il >= B.il ? A : B;

}

tayin edila bilar.

Borland C++-da overload göstarilmadan da bütün funksiyalar1n üzarina yüklama apar1la bilar. Komplyator Siza bununla alaqadar xabardarl1q edarsa, buna ahamiyyat vermayin va ya #progma warn –ovl drektivindan istifada edarak bu tipli xabardarl1q malumatlar1n1n qar§1s1n1 ala bilarsiniz.

### Operatorlarin Tayini

C++-da operatorlar1 da funksiya kimi qabul etmak

olar.

c = a + b; yerina c = operator+ (a, b);

c = a - b; yerina c = operator– (a, b);

c = a + b – d \* 4; yerina c = operator– (operator + (a, b),

operator\* (d, 4));

ifadalarini yazmaq olar. Burada operator+, operator- va operator\* har biri bir funksiyad1r. C++-da operator sözü funksiya ad1d1r. Bu addan istifada edarkan ondan sonra bir operator i§arasi yazmaq laz1md1r. C va C++-da tayin olunmu§ operator i§aralari Cadval 2.1-da göstarilmi§dir.

**Cadval 2.1** Operatorlar

|  |  |
| --- | --- |
| **Operator i�arasi** | **Operator Ad1** |
| + - \* / % | Riyazi operator |
| = += -= \*= /= %= ^= &= |= <<= >>= | Manimsatma operatorlarn |
| ^ & | | Bit saviyya operatorlarn |
| << >> | Sürü$dürma operatorlarn |
| < > == != <= >= ! && || | Mantiqi operatorlar |
| ++ - - + - | Artnrma/azaltma operatorlarn |
| () | Cevirma operatoru |
| [] | Massiv operatoru |
| & | Ünvan operatoru |
| sizeof | Uzunluq operatoru |
| new, delete | Yadda$ operatoru |

Operatorlara funksiya kimi bax1ld1g1 üçün funksiyalar üçün nazarda tutulmu§ olan üst-üsta yüklama amaliyyat1 operatorlar üçün da istifada edila bilar. Bununla da yeni tayin olunan tiplar üçün operator funksiyalar1n1n üzarina yüklamak olar.

struct vector { double x, y, z; }

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

double operator\*(struct vector A, struct vector B)

{ return A.x \* B.x + A.y \* B.y + A.z \* B.z; }

Bu misalda tayin olunmu§ vector tipindan as1l1 olaraq iki vektorun skalyar hasilini hesablayan vurma operatoru da tayin edilmi§dir.

struct vector V1 = {1, 2, 3};

struct vector V2 = {7, 8, 2}; double skalyarhasil = V1 \* V2;

satirlari proqram satirlaridir. Lakin hala tan1mlanmad1g1 üçün

struct vector V3 = V1 + V2;

kimi bir satir istifada edila bilmaz. Ancaq vector strukturu üçün toplama operatoru tayin edildikdan sonra istifada oluna bilar.

### Aktiv Qiymat Vermak

Müstavi üzarinda qövs, markazi (Cx, Cy), radiusu (R), ba§lang1c (Sa) va son (Ea) bucag1 ila tayin olunur. Bela bir qövsün uzunlugu 2R(Ea-Sa)/360 düsturu ila hesablan1r.

#include <math.h>

double Qovs(double Cx, double Cy, double R, double Sa, double Ea)

{ return 2\*M\_PI\*R\*(Ea-Sa)/360; }

Bu cür bir Qovs alt proqram1 yaz1la bilar. 8gar bu proqram hissasi tam bir çevra üçün istifada olunarsa, Sa yerina 0, Ea yerina isa 360 yaz1lmal1d1r.

double Tam = Qovs(Cx, Cy, R, 0, 360); double Yarim = Qovs(Cx, Cy, R, 0, 90);

Bela ki, çevra çox istifada olunan oldugu üçün 0 va

360 qiymatlari avvalcadan malum parametrlardir. Har dafa bu qiymatlari göstarmaya ehtiyac yoxdur. Çevra üçün

double Tam = Qovs(Cx, Cy, R);

§aklinda istifada olunmas1 daha q1sa yoldur. Bunu funksiya üzarina yüklama üsulundan istifada edarak yenidan yazmaq mümkündür. Lakin bunun yerina funksiya prototipi tayin edilarkan,

double Qovs(double Cx, double Cy, double R,double Sa = 0.0, double Ea = 360.0);

§akilnda parametrlara ***aktiv qiymat*** vermak da mümkündür. Bu tayindan sonra radiusu 50.0 olan bir çevra tayin olunarkan

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

double Tam = Qovs(Cx, Cy, 50.0);

istifada olunarsa, komplyator avvalca

Qovs(double, double, double)

kimi tayin olunmu§ bir funksiya axtaracaqd1r. Tapmad1g1 zaman da

Qovs(double, double, double, ...)

kimi va ilk üç qiymatdan sonrak1lara aktiv qiymat verilmi§ bir funksiya axtaracaqd1r. Tapd1g1 zaman ilk qiymatlari da yazaraq bu funksiyan1 çag1racaqd1r. Yani komplyator

double Tam = Qovs(Cx, Cy, 50.0);

amrini

double Tam = Qovs(Cx, Cy, 50.0, 0.0, 360.0);

kimi istifada edacakdir.

Bu funksiyan1n bütün parametrlarina aktiv qiymat vermak mümkündür. Ya da misalda oldugu kimi, sadaca müayyan parametrlara aktiv qiymat manimsadila bilar. Lakin aktiv qiymat verilmi§ har parametrin sag1ndak1 parametra da aktiv qiymat verilmalidir. Masalan,

void misal(double x = 0, double y = 0); //dogru

void misal(double x, double y = 0); //dogru

void misal(double x = 0, double y); //sahv

Üçüncü misal, aktiv qiymat verilmi§ x parametrinin sag1ndak1 bir parametra (y parametrina) aktiv qiymat verilmadiyi üçün istifada oluna bilmaz.

Daha bir hal1 nazara almaq laz1md1r ki, aktiv qiymat verilarak alda edilan prototiplarla, üzarina yüklama apar1lan funksiyalardan alda edilan prototiplar eyni deyillar.

double Qovs(double Cx, double Cy, double R, double Sa = 0.0, double Ea = 360.0):

tayini ila barabar

double Qovs(double Cx, double Cy, double R);

double Qovs(double Cx, double Cy, double R, double Sa); double Qovs(double Cx, double Cy, double R, double Sa, double Ea);

funksiyalar1 birlikda tayin edila bilmazlar, çünki, anla§1lmazl1q yarana bilar.

|  |  |
| --- | --- |
| int X; |  |
| int &R = X; | /\* & i$arasi R-in taqdimat dayi$kani oldugunu göstarir. X manimsadilmasi ila da R va X bir-biri ila alaqalandirilir. \*/ |
| X = 20; | /\* R = 20; \*/ |
| R = 40; | /\* X = 40; \*/ |
| R++; | /\* X++; ++ operatorunun R-in qiymatini bir  vahid artnrdngnna, sonrakn qiymatini göstarmadiyina diqqat edin. \*/ |

Qovs(100.0, 120.0, 40.0);

funksiyas1n1n çag1r1lmas1 zaman1 hans1 funksiyaya müraciat edilacayi balli olmur.

### Taqdimat (Referans) Tip

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

**Tayinedicisi**

C-da bir dayi§kanin ba§qa bir dayi§kani göstarmasi üçün **göstarici** (pointer), malumatlar1n emal1 üçün da **göstarici amaliyyatlari** (pointer aritmetics) anlay1§1ndan istifada olunur.

int X = 10, Y = 20;

int \*P;

P = &X;

\*P = 12; /\* X = 12; \*/

X = 14; /\* \*P = 14; \*/ P = &Y;

\*P=12; /\* Y = 12; \*/ X = 15;

C++-da bu anlay1§ yerina taqdimat tipi tayin edilir. Bu tipdli tayinlarda dayi§kan ba§qa bir dayi§kanla alaqalandirilir. Lakin göstaricilardan farqli olaraq bu alaqa pozularaq ba§qa bir alaqa qurula bilmaz.

Taqdimat dayi§kanlarinin tayini zaman1 hans1 dayi§kan ila bagl1 olduqlar1n1n bildirilmasi macburidir. Taqdimat dayi§kani sabitla da alaqalandirila bilar.

int &RR = 2;

Haqiqatda isa komplyator müvaqqati bir dayi§kan tayin edarak ona 2 qiymatini manimsadir va sonra bu dayi§kanin taqdimat dayi§kani tarafindan göstarilmasini tamin edir. Bela ki,

int Muveqqeti = 2;

int &RR = Muveqqeti;

Lakin proqramç1n1n bu müvaqqati dayi§kani istifada etma haqq1 yoxdur.

Bu çür tayin etma müxtalif vaxtlarda eyni maqsad üçün istifada edilmi§ dayi§kanlari birla§dirmaya kömak edir.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Bu, funksiyalardan istifada edarkan daha ahamiyyatlidir. indi C-da parametr kimi daxil edilan dayi§kanin qiymatini bir vahid art1ran INC adl1 funksiya va onu çag1ran bir proqram yazaq.

/\* REF.CPP \*/

#include <conio.h>

#include <stdio.h> double Zaman = 0;

void ZamanYaz()

{ printf("Zaman = %lf\n", Zaman); }

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*/ double &Z = Zaman;

void ZamanArtir()

{ Z += 6; }

int main()

{ clrscr(); ZamanYaz();

ZamanArtir(); ZamanYaz();

ZamanArtir(); ZamanYaz();

return 0;

}

/\* INC.C \*/

void INC(double \*D)

{ (\*D)++; }

int main()

{ double X, \*Y, Z;

X = 6; Z = 8; Y = &Z;

INC(&X);

INC(Y);

/\* X = 7; \*/

/\* \*Y = 9 ve ya Z = 9 \*/

/\* INC(\*Y);

Sehvdir \*/

return 0;

}

indi da C++-da taqdimat tayin edicisi ila bu proqram1 yazaq.

Proqram ç1x1§1

Zaman = 0.000000

Zaman = 6.000000

Zaman = 12.000000

/\* INC.CPP \*/

void INC(double &D)

{ D++; }

int main()

{ double X, \*Y, Z;

X = 6; Z = 8; Y = &Z;

INC(X);

/\* X = 7; \*/

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

istayirsinizsa, :: tayinedicisindan, yani görma (scope) operatorundan istifada etmalisiniz. Bela ki,

INC(\*Y);

/\* \*Y = 9 ve ya Z = 9 \*/

/\* INC(Y);

Sehvdir \*/

return 0;

}

### Gizlanmi§ Dayi§kanlari Görmak

/\* SCOPE.CPP \*/

#include <iostream.h>

static int X = 5; /\* X int tipinde ve qiymeti 5 \*/ void f()

{ double X = 27.5e30; /\* X double tipinde ve qiymeti 2.75e31 \*/

cout<<X<<endl;

X = 71e12;

/\* X double tipinde ve qiymeti 7.1e13 \*/

::X = 71;

/\* X int tipinde ve qiymeti 71 (qlobal X-dir) \*/

}

/\* X int tipinde qiymeti 71 \*/

C-dan bildiyimiz kimi bir blok daxilinda tayin olunmu§ dayi§kanin ad1 daha avval tayin olunmu§ dayi§kan ad1 ila üst-üsta dü§arsa, son tayin olunmu§ dayi§kan blokun sonuna qadar öz funksiyas1n1 yerina yetirarak digar dayi§kana müraciatin qar§1s1n1 al1r (ümumi va lokal dayi§kan anlay1§lar1n1 xat1rlay1n).

Uygun olaraq ifadalarda ümumi tayin olunmu§ dayi§kanlarin

/\* SCOPE.C \*/

#include <stdio.h> int X = 5;

/\* X int tipinde ve qiymeti 5 \*/

void f()

{ double X = 27.5e30; /\* X double tipinde ve qiymeti 2.75e31 \*/

X = 71;

/\* X double tipinde ve qiymeti 71 \*/

}

/\* X int tipinde \*/

Y = X \* ::X + X + ::X;

§aklinda istifadasi da mümkündür.

C++-da X = 71 manimsadilmasinin lokal deyil, ümumi tayin olunmu§ X-a manimsadilmasini

### C++-da Prototiplarin Tayin Edilmasi

C++-da C-da istifada olunan klassik stildaki prototip tayinlarina icaza verilmir. Bunlar1n yerina daha müasir prototip tayinlarindan istifada edilir.

double f(a, b, c); int a;

double b; float \*c;

{

...

...

...

### Struktur Tiplar

struct, union va enum strukturlar1 ila typedef tayinedicisi istifada edilmadan tip tayinlarinin istifada edilmasi zaman1 struct, union va enum sözlarinin dayi§kanin ad1ndan avval göstarilmasinin vacibliyi C++- da aradan qald1r1lm1§d1r.

Tip tayini

struct Telebe

{ char Adi[20]; int No;

char Qiymeti;

};

}

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

tayini sahvdir.

double f(int a, double b, float \*c);

{

...

...

...

}

tayini dogrudur. Sadaca prototip tayininda da

double f(int a, double b, float \*c); /\* ve ya \*/ double f(int, double, float\*);

tayini dogrudur.

olarsa, C-da dayi§kanin tayini

struct Telebe A, B, C;

C++-da isa sadaca

Telebe A, B, C;

§aklinda yaz1la bilar.

### $arh Operatoru

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Proqram daxilinda proqramç1n1n verdiyi §arhlar növbati marhalalarda düzali§lar, dayi§ikliklar va digar proqramç1lar1n proqrama müdaxilasi üçün çox ahamiyyatlidir. Yax§1 yaz1lm1§ bir proqram daxilinda proqram satirlarindan daha çox §arh satirlari olur. Bütün proqramla§d1rma mühitlarinda oldugu kimi C-da da §arh operatoru vard1r. Bu /\* ila ba§lay1b \*/ bitan satirlardir. Proqram1n istanilan bir yerinda qoyula bilar. Bu operator C++-da da istifada oluna bilar.

C++-da bundan ba§qa // i§arasi ila ba§layan §arh operatorundan da istifada olunur. Bu i§ara ila ba§layan

§arh satirlari növbati satirdan davam eda bilmaz.

//SERH.CPP

#include <conio.h>

#include <stdio.h>

main()

{ clrscr();

double PI = 3.14;

// Proqrama baslama.

// Ekranin temizlenmesi.

// Pi-nin qiymeti menimsedilir.

double X; // Bucaq qiymeti ucun teyin olunmus deyisken. int Err; // Sehvlere nezaret ucun teyin olunmus deyisken. printf("Bucagin qiymetini derece olaraq daxil ediniz.\n");

Err = scanf("%lf", &X); // X-in qiymeti bucaqla daxil edilmeli.

// Yalniz bir qiymet daxil edilmelidir.

// Err-in qiymeti 1 olmalidir.

if (Err != 1)// Err-in qiymeti 1-den ferqli olarsa, sehf oxuma bas verir.

{ fprintf(stderr, "Yalniz heqiqi eded daxil edin.\n");

return 1;

// Xetali proqram cixisi.

}

fflush(stdin); // Giris yaddasindaki artiq melumatlari silir.

X \*= PI / 180.0;

printf("%lf radyan\n", X); return 0;

// X radyana cevrilir.

// Proqramdan sehv olmadan cixilir.

}

// Proqramin sonu.

### new va delete Opratorlari

Dinamik yadda§dan istifada edarkan icra olunan iki asas amaliyyat **yadda§da yer ayrilmasi** (malloc, calloc) va ayr1lan bu sahanin istifada edildikdan sonra **sarbast** (free) burax1lmas1d1r. Bu amaliyyatlar üçün C-da prototiplari stdlib.h va alloc.h ba§l1q fayllar1nda göstarilan funksiyalardan istifada edilir.

double \*DP = (double\*)malloc(sizeof(double)); int \*IP = (int\*)malloc(sizeof(int));

struct Date {int Gun, Ay, Il;};

struct Date \*SDP = (struct Date\*)malloc(sizeof(struct Date));

ayr1lan bu sahalari sarbast buraxmaq üçün da

free((void\*)DP);

free((void\*)IP); free((void\*)SDP);

amrlari istifada edilir. C++-da bu funksiyalardan ba§qa eyni zamanda

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

double \*DP = new double; int \*IP = new int;

struct Date {int Gun, Ay, Il;};

struct Date \*SDP = new struct Date;

istifada etmak va ayr1lan bu sahalari sarbast buraxmaq üçün da

delete DP; delete IP; delete SDP;

ifadalarindan istifada etmak olar.

Eyni tipda iki va daha art1q yer ay1rmaq üçün (masalan, 100 double tipli elementi olan A massivi üçün)

double \*A = new doble[100];

amrindan istifada etmak olar. Bu amaliyyat naticasinda

100 double tipli adadin yerla§dirilmasi üçün saha ayr1laraq ilk ünvan1 A-ya manimsadilacakdir. Bu manimsatma naticasinda \*(A+3) = 81; va ya A[3] = 81; amaliyyatlar1 ila massivin 4-cü elementina qiymat manimsadilmasi mümkündür. Burada massivin har elementinin double tipinda olmas1na diqqat etmak laz1md1r.

Bela bir sahani sarbast buraxmaq üçün

delete A;

amrindan istifada etmak kifayatdir. (Bazi C++ proqramlar1nda delete amri ila yana§1 massivin ölçüsünü da göstarmak laz1m gala bilar (delete [100]A; kimi).

8gar ayr1lacaq sahaya double tipli qiymatlarin göstaricilari yerla§dirilacaksa,

double \*\*A;

A = new double\*;

kimi istifada olunmal1d1r. Egar double göstaricilarinin massivi istifada edilacaksa,

A = new double \*[100];

kimi istifada olunmal1d1r.

Bundan ba§qa new va delete operatorlar1 üçün massiv kimi yer ay1rmalar1nda istifada edilan elementlarin say1n1n sabit olmas1 vacib deyildir. Elementlarin say1 dayi§kan ola bildiyi kimi, amaliyyat naticasinda da alda edila bilar.

### inline Makrolari

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

inline makrolar1 parametrik makrolara ox§ar §akilda icra olunmas1na baxmayaraq funksiyalar kimi tayin edila bilar. Masalan, funksiya tayini

int max(int a, int b)

{ return a > b ? a : b; }

makro tayini

#define max(a, b) ((a) > (b) ? (a) : (b))

inline tayini a§ag1dak1 kimidir:

inline int max(int a, int b)

{ return a > b ? a : b; }

inline makrolar1n1n funksiyalara ox§ad1g1na diqqat edin. inline makrolar1n1 tayin edarkan sanki, bir funksiya tayin etmi§ olursunuz. Sadaca tayin inline ifadasi ila ba§lay1r.

Digar makrolardan farqli olaraq inline makrolar1nda tip nazarati apar1l1r. Masalan, yuxar1da tayin edilmi§

#define makrosunun istifada edilmasi zaman1

double d = max(3.8, 3.1);

heç bir sahv olmadan icra olunur. inline makrosundan istifada edilarsa, bunun sahvsiz icra olunacag1n1 gözlamak çatindir. Bela ki, max(double, double) kimi tayin olunmu§ bir funksiya va ya makro yoxdur. Buna göra da double tipli adadlar int tipina çevrilarak istifada olunacaq va natica da 3 olacaqd1r.

inline makrolar1n1n yaz1lacag1 yer onlar1n makro olmas1 nazara al1naraq tayin edilmalidir. Yani, proqram kodu hissasinda deyil, tayin etma hissasinda yerla§dirilmalidir. Çünki inline makrolar1 kitabxanalarda saxlan1lm1r.

inline makrolar1 ila süratli icra olunan sada manimsatma va nazarat amaliyyatlar1ndan ibarat funksiyalar yaz1la bilar. inline makrolar1 daxilinda goto, for, do-while, while, break, continue, switch, case amrlarinin istifada edilmasi strukturun böyümasina sabab oldugu üçün, bunlar1n istifada edilmasi maqsadauygun deyildir.

### Obyekt Nadir?

**III F8SiL**

**OBYEKTL8R**

göstarilmadiyi halda, yaln1z üzvlar bir-birlarini çag1ra bilarlar. Digar obyekt va funksiyalar üçün isa bagl1d1rlar.

indi biz yaln1z daira va halqalar1 tan1yan, bunlar1n saha va çevralari ila alaqadar olan obyektlari tayin etdikdan sonra istifada edan proqram1 tartib edak.

// DAIRE.CPP

#include <math.h> struct DAIRE {

double Diametr;

double Cevre(); double Sahe();

};

double DAIRE::Cevre()

{ return Diametr \* M\_PI; }

double DAIRE::Sahe()

{ return Diametr \* Diametr \* M\_PI / 4; }

**Obyekt** (Object), yadda§1n dayi§dirila bilan qiymatlar va ya müayyan funksiyalar1 yerina yetiran adland1r1lm1§ sahasidir. Bu bax1mdan bütün dayi§kanlar bir obyektdir. Obyektlarin davran1§lar1na göra tasniflandirilmasi da **sinif** (class) anlay1§1n1 meydana gatirir. Bu bax1mdan da verilanlarin tiplari bir sinfi ifada edir. Bazi manbalarda sinif yerina obyekt, obyekt yerina isa **nümuna** (instance) anlay1§1ndan istifada edilir.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Obyektlar dayi§kan va funksiyalardan ibarat olan struktur dayi§kanlaridir. Obyekta daxil olan dayi§kanlara **üzv dayi§kanlari** (member variables), funksiyalara da **üzv fonksiyalari** (member functions) ad1 verilir.

C++-da obyektlar iki cür tayin olunur: struct va class sözü ila ba§layan tayin. Bunlar aras1ndak1 yegana farq, agar aksi göstarilmazsa, struct ila tayin olunan obyektlarin bütün üzv dayi§kan va funksiyalar1na digar obyekt va funksiyalar müraciat edarak istifada eda bilarlar. class ila tayin olunan obyektlarda isa aksi

Burada:

* struct strukturu daxilinda funksiyalar1n da tayin edilmasina;
* funksiyalar1n kodla§d1r1lmas1 zaman1 heç bir tayin olmadan Diametr dayi§kaninin istifada edilmasina;
* kodla§d1r1lma apar1larkan funksiya ad1n1n avvalina funksiyan1n aid oldugu sinif ad1n1n DAIRE:: §aklinda alava edilmasina diqqat edin.

Funksiya adlar1n1n avvalina aid olduqlar1 sinfin ad1n1n yaz1lmas1n1n sababi digar siniflarin da eyni adl1 üzv funksiyalar1n1n ola bilacayi ehtimal1d1r. Müxtalif siniflara aid eyni adl1 funksiyalar1 ay1rmag1n yegana yolu sinif ad1 ila barabar görma operatorunun istifada edilmasidir.

Natica olaraq obyektin kodla§d1r1lmas1 ila alaqadar bu qaydalar1 qeyd edmak olar:

1. Funksiyalar da strukturun bir hissasiymi§ kimi

dayi§kanlarla birlikda tayin oluna bilarlar;

1. Funksiya ila eyni struktur daxilinda tayin olunmu§ dayi§kanlar heç bir tayin olmadan funksiya tarafindan istifada oluna bilarlar;
2. Struktura aid funksiyalar yaz1larkan, aid olduqlar1 strukturu göstarmak üçün funksiyan1n ad1n1n avvalina strukturun ad1, aralar1na isa görma operatoru yaz1lmal1d1r.

double KVADRAT::Cevre()

{ return Hundurluk \* 4; }

double KVADRAT::Sahe()

{ return Hundurluk \* Hundurluk; }

Cevre va Sahe funksiyalar1n1n burada KVADRAT

üçün yenidan tayin edildiyina diqqat edin.

//OBJECT1.CPP

#include <stdio.h>

#include <conio.h>

#include "daire.cpp"

#include "kvadrat.cpp" KVADRAT K1, K2; DAIRE D;

main()

{ clrscr();

K1.Hundurluk = D.Diametr = 10.0; K2.Hundurluk = 6;

printf("\nOlculer:\nKvadrat\tHundurluk1 = %lf\t" "Hundurluk2 = %lf\nDaire\tDiametr = %lf\n", K1.Hundurluk, K2.Hundurluk, D.Diametr);

printf("\nSaheler:\nKvadrat\tSahe1 = %lf\t" "Sahe2 = %lf\nDaire\tSahe = %lf\n", K1.Sahe(), K2.Sahe(), D.Sahe());

printf("\nCevreler:\nKvadrat\tCevre = %lf\t"

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

//KVADRAT.CPP

struct KVADRAT { double Hundurluk; double Cevre(); double Sahe();

};

"Cevre2 = %lf\nDaire\tCevre = %lf\n", K1.Cevre(), K2.Cevre(), D.Cevre());

printf("\nKvadrat halqanin sahesi = %lf\n", K1.Sahe() - K2.Sahe()); return 0;

}

DAIRE::DAIRE(double C)

yazilmasi

{ Diametr = C > 0 ? C : -C; }

// Layihelendiricinin

double DAIRE::Cevre()

{ return Diametr \* M\_PI; }

double DAIRE::Sahe()

{ return Diametr \* Diametr \* M\_PI / 4; }

### Layihafandirici

Dayi§kanlar kimi, obyektlari da tayin edarkan onlara ba§lang1c qiymat vermak olar. Bunun üçün obyekt sinfi tayin edilarkan, obyektin yarad1lmas1 zaman1 istifada edilacak xüsusi bir funksiya obyekt sturukturu ila barabar tayin edilir. Bu funksiyan1 digar funksiyalardan farqlandiran asas xüsusiyyati ad1n1n tayin olunan sinif ad1 ila eyni olmas1d1r. Bu funksiya heç bir qiymati geri qaytarm1r. Bu funksiyaya **layihafandirici** (constructor) deyilir.

Bu tayinda bundan avvalki misala alava olaraq

DAIRE sinfinin layihalandiricisi tayin edilmi§dir.

//OBJECT2.CPP

#include <conio.h>

#include <stdio.h>

#include "dairecon.cpp"

main()

{ clrscr();

DAIRE Xarici(30); // Layihelendiricinin istifade edilmesi DAIRE Daxili(20);

printf("\nDaire diametrleri\nXarici -> %lf\tDaxili -> %lf\n", Xarici.Diametr, Daxili.Diametr);

printf("\nDaire halqasinin sahesi = %lf\n", Xarici.Sahe() - Daxili.Sahe());

return 0;

}

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

//DAIRECON.CPP

#include <math.h> struct DAIRE

{ double Diametr;

DAIRE(double); double Cevre(); double Sahe();

// Layihelendiricinin teyin edilmesi

};

Proqram ç1x1§1

Daire diametrleri Xarici -> 30.000000

Daxili -> 20.000000

Daire halqasinin sahesi = 392.699082

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Bu cür istifada ila Diametr üzv dayi§kanina müraciat sadala§ir. Digar tarafdan layihalandiricinin i§a qo§ulmas1 avtomatik olaraq hayata keçirilir. Bunun üçün proqramç1n1n alava cahtlar etmasina ehtiyac qalm1r.

### Müraciat Haqqi

Bir obyektin dayi§kan va funksiyalardan ibarat üzvlarinin digar obyektlar tarafindan birba§a istifada edilmamasi üçün, bu obyektlar digar obyektlara qar§1 qoruna bilarlar.

Bu qorunma 4 müxtalif halda ola bilar:

1. Bir üzv dayi§kan va ya üzv funksiyas1n1n yaln1z üzvü oldugu obyekt daxilinda istifada edilmasi - **xüsusi** (*private*);
2. Bir üzvün obyekt xaricinda yaln1z o obyektdan töranan obyektlar tarafindan istifada edila bilmasi

- **qorunmu§** (*protected*);

1. Bir üzvün bütün obyektlar tarafindan ortaq istifada edila bilmasi - **ümumi** (*public*);
2. Bir obyektin ba§qa bir obyekti “dostu” elan edarak üzvlarinin ham1s1n1n bu obyekt tarafindan istifada edilmasina icaza vermasi - **dost** (*friend*).

Bunlardan ilk üçü çox istifada edilir.

//ADSTRC.H

struct Adlar

{ private:

char Ad[20]; int Yas;

void Boyuk();

public: Adlar(char\*, int);

int Yaz();

int NormalYaz();

};

struct ila obyekt tayinlarinda istifada haqq1 ba§lang1cda public olur. 8ksi göstarilmadikda bu bela da qal1r. class ila obyekt tayinlarinda isa ba§lang1cda private olur. Bundan ba§qa struct va class vasitasila obyekt (sinif) tayinlari aras1nda farq yoxdur. Yuxar1dak1 misaldan göründüyü kimi hans1 üzvün hans1 istifadaçi saviyyasinda tayin edilacayini, tayindan avval private:, protected: va ya public: kimi ifadalardan istifada edarak müayyanla§dirmak mümkündür.

Yuxar1dak1 misalda Ad va Yas dayi§kanlari ila Boyuk funksiyas1 private, digar üzv funksiyalar isa public kimi tayin olunmu§dur.

Eyni obyekti class açar sözündan istifada edarak

a§ag1dak1 kimi tayin etmak olar:

class ad

{

...

...

...

};

struct ad

{ private:

...

...

...

};

Üzv funksiyalar1n1n struct va ya class ila kodla§d1r1lmas1 aras1nda ela bir farq yoxdur.

//ADCLASS.H

class Adlar

{

char Ad[20]; int Yas;

void Boyuk();

public: Adlar(char\*, int);

int Yaz();

int NormalYaz();

};

Üzv funksiyalar1n1n digar üzvlari, xüsusila da üzv dayi§kanlarini sanki, lokal dayi§kanlar kimi istifada etdiklarina diqqat edin.

struct va class tayinlarinin müqayisasini a§ag1dak1 kimi göstarmak olar:

//ADDEC.CPP

#define STRUCT

#ifdef \_\_STRUCT

#include "adstrc.h"

#else

#include "adclass.h"

#endif

#include <conio.h>

#include <stdio.h>

#include <string.h>

#include <ctype.h>

Adlar::Adlar(char\* N, int Y)

{ strcpy(Ad, N);

Yas = Y;

}

struct ad

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

{

...

...

...

};

class ad

{ public:

...

...

...

};

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Kenan Seyidzade KAMRAN Resulov

Bela bir proqram daxilinda A.Boyuk() kimi müraciat sahv qabul edilacakdir. Bu funksiyan1n mövcud olmas1na baxmayaraq, main() funksiyas1n1n ona müraciat haqq1 yoxdur.

void Adlar::Boyuk()

{ char \*p;

for(p = Ad; \*p; ++p) if(islower(\*p))

\*p = toupper(\*p);

}

int Adlar::Yaz()

{ char t[20]; int i;

strcpy(t, Ad);

Boyuk();

i = NormalYaz(); strcpy(Ad, t); return i;

}

int Adlar::NormalYaz()

{ return puts(Ad); }

Adlar A("Kenan Seyidzade", 7); Adlar B("KAMRAN Resulov", 8);

main()

{ clrscr();

A.Yaz();

B.Yaz();

A.NormalYaz(); B.NormalYaz();

return 0;

}

### Yoxedici (Destructor)

Layihalandirici neca ki, obyektin mövcud olmas1 hal1nda onu proqrama haz1rlay1r, **yoxedici** funksiyas1 da obyekt proqram xaricinda qald1g1 zaman onun mahdudla§d1rd1g1 va ya dayi§dirdiyi kompüter mühitlarinin yenidan nizamlanmas1n1 tamin edir. Bu funksiyaya, xüsusila obyektin dinamik yadda§dan istifada etdiyi zaman ehtiyac olur.

Har hans1 bir yoxedicinin ad1 tayin olundugu obyektin tipi (sinfi) ila eynidir. Yaln1z layihalandiricinin ad1 ila qar1§d1rmamaq üçün avvalina “~” (tilde) i§arasi qoyulur. Bu funksiya har hans1 bir qiymat hasil etmadiyi kimi, heç bir parametri da yoxdur.

Proqram ç1x1§1

KENAN SEYIDZADE KAMRAN RESULOV

// MASSIV.CPP

#include <conio.h>

#include <stdio.h>

#include <stdlib.h>

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

class Massiv

{ int \*p; int Olcu;

//Massivin ilk elementinin gostericisi

//Massivin olcusu

public:

Massiv(int s); //Layihelendirici s elementli bir massiv yaradir

~Massiv(); //Yoxedici

int Getir(int i);

//Massivin i-ci elementinin qiymetini verir

void Menimset(int i, int d);

//Massivin i-ci elementine d-ni menimsedir

void Sehv(char\* msg);

//Sehv mesajlarini gosterir

double Orta();

//Massivin orta qiymetini hesablayir

};

Massiv::Massiv(int s)

{ printf("Layihelendirici islemeye basladi.\n");

if (s <= 0) Sehv("Menfi ededler olcu ola bilmez!"); if ((p = new int [Olcu = s]) == NULL)

Sehv("Massive yer ayrila bilmedi!");

printf("Layihelendirici isini dayandirdi.\n\n");

}

Massiv::~Massiv()

{ printf("Yoxedici islemeye basladi.\n");

if (p != NULL) delete p; printf("Yoxedici isini dayandirdi.\n");

}

int Massiv::Getir(int i)

{ if (i < 0 || i > Olcu)

Sehv("Massivin olcusu xaricinde!"); return p[i];

}

void Massiv::Menimset(int i, int d)

{ if (i < 0 || i > Olcu)

Sehv("Massivin olcusu xaricinde!"); else p[i] = d;

}

void Massiv::Sehv(char\* msg)

{ fprintf(stderr, "\a%s\n", msg); exit(4);

}

double Massiv::Orta()

{ int i;

double t;

for (i = 0, t = 0; i < Olcu; i++) t += p[i];

return t / Olcu;

}

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Misallar \*\*\*\*\*\*\*\*\*\*\*\*\*\* Massiv A(3);

void Misal1()

{ printf("\n1-ci Misal\tLokal teyin etme\n\n"); Massiv B(5);

}

#pragma warn -aus

// 'C' is assigned a value that is never used

// mesajinin qarsisini alir

void Misal2()

{ printf("\n2-ci Misal\tLokal gosterici teyin etme\n\n"); Massiv \*C = new Massiv(5);

}

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Proqram ç1x1§1

#pragma warn +aus void Misal3()

{ printf("\n3-cu Misal\tLokal gosterici teyin etme ve silme\n\n");

Massiv \*D = new Massiv(5); delete D;

}

main()

{ clrscr();

printf("main funksiyasi icra olunmaga basladi.\n");

Misal1();

Misal2();

Misal3();

printf("\nUzvlerin istifade olunmasi\n\n"); Massiv E(4);

int i;

for (i = 0; i < 3; i++) A.Menimset(i, rand());

for (i = 0; i < 4; i++) E.Menimset(i, rand());

for (i = 0; i < 3; i++)

printf("A[%d] = %d\n", i, A.Getir(i));

printf("\nOrta qiymetler\n\n A = %lf,\t E = %lf\n\n", A.Orta(), E.Orta());

printf("main funksiyasi icrasini tamamladi.\n\n"); return 0;

}

|  |  |
| --- | --- |
| main funksiyasi icra olunmaga basladi. | A üçün |
| 1-ci Misal Lokal teyin etme |  |
| Layihelendirici islemeye basladi. | B üçün |
| Layihelendirici isini dayandirdi. |  |
| Yoxedici islemeye basladi. | B üçün |
| Yoxedici isini dayandirdi. |  |
| 2-ci Misal Lokal gosterici teyin etme |  |
| Layihelendirici islemeye basladi. | C üçün |
| Layihelendirici isini dayandirdi. |  |
| 3-cu Misal Lokal gosterici teyin etme ve silme |  |
| Layihelendirici islemeye basladi. | D üçün |
| Layihelendirici isini dayandirdi. |  |
| Yoxedici islemeye basladi. | D üçün |
| Yoxedici isini dayandirdi. |  |
| Uzvlerin istifade olunmasi |  |
| Layihelendirici islemeye basladi. | E üçün |
| Layihelendirici isini dayandirdi. |  |
| A[0] = 346 |  |
| A[1] = 130 |  |
| A[2] = 10982 |  |
| Orta qiymetler |  |
| A = 3819.333333, E = 9364.500000 |  |
| main funksiyasn icrasnnn tamamladn. |  |

yüklama üsulu layihalandiricilar üçün da istifada oluna bilar.

|  |  |
| --- | --- |
| Yoxedici islemeye basladi. | E üçün |
| Yoxedici isini dayandirdi. |  |
| Yoxedici islemeye basladi. | A üçün |
| Yoxedici isini dayandirdi. |  |

### Standart Obyekt Tiplari

C-da int, char, double kimi tan1nan standart tiplarün har biri C++-da bir sinif kimi istifada oluna bilar. Bunlara a§ag1dak1 kimi qiymatlar manimsatmak olar. Bu cür obyektlara manimsadilacak qiymat, siniflara uygun olmal1d1r. 8gar uygun deyilsa, onlar1 uygunla§d1rmaq laz1md1r. Bu da bazi sahvlara yol aça bilar.

int x = 5; yerina int x(5);

double pi = 3.14; yerina double pi(3.14);

int Y = X; yerina int Y(X);

//MAS3CON.CPP

#include <conio.h>

#include <stdio.h>

#include <stdlib.h>

class Massiv

{ int \*p; int Olcu;

//Massivin ilk elementinin gostericisi

//Massivin olcusu

public: Massiv();

//Layihelendirici Ilk qiymeti -1000 olan 100

//elementli massiv yaradir.

Massiv(int s); //Layihelendirici Ilk qiymeti -1000 olan s elementli

//massiv yaradir.

Massiv(int s, int d); //Layihelendirici Ilk qiymeti d olan s elementli

//massiv yaradir.

Masalan, bundan avvalki misalda baxd1g1m1z massiv sinfinin layihalandiricisi verilan miqdarda yer ay1r1rd1. 8gar massivin ölçüsü göstarilmazsa, 100 oldugu qabul edilsin va ölçüsündan as1l1 olmayaraq massivin har bir elemenitina -1000 adadi manimsadilsin. Yena laz1m galarsa, ba§lang1c qiymat da verila bilar.

Bunun üçün üç layihalandiriciya ehtiyac vard1r:

1. Heç bir qiymatin verilmamasi hal1;
2. Ölçünün verilmasi hal1;
3. Har ikisinin verilmasi hal1.

float \*f;

f = (float\*)malloc(sizeof(float));

\*f = 7.0/2.0;

yerina float \*f = new float(7.0/2.0);

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

### Layihafandirici Üzarina Yüklama

Bir layihalandirici, üzvü oldugu obyektin ba§lang1c vaziyyatini nizamlayarkan proqramç1n1n istaklari müxtalif ola bilar. Bunun üçün da müxtalif layihalandiricilarin istifada edilmasi laz1m ola bilar. Bunu yerina yetirmak üçün Fasil 2-da §arh edilan üzarina

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

~Massiv();

//Yoxedici

int Getir(int i); //Massivin i-ci elementinin qiymetini verir

void Menimset(int i, int d); //Massivin i-ci elementine d-ni

//menimsedir

void Sehv(char\* msg);

//Sehv mesajlarini gosterir

double Orta();

//Massivin orta qiymetini hesablayir

};

Massiv::Massiv()

{ int s = 100; //Qebul edilen massivin olcusu 100; if ((p = new int [Olcu = s]) == NULL)

Sehv("Massive yer ayrila bilmedi!"); for (; s;)

p[--s] = -1000;

}

Massiv::Massiv(int s)

{ if (s <= 0) Sehv("Menfi ededler olcu ola bilmez!"); if ((p = new int [Olcu = s]) == NULL)

Sehv("Massive yer ayrila bilmedi!"); for (; s;)

p[--s] = -1000;

//s ededini Olcu uzv deyiskenine yazdigi ucun saygac olaraq

//istifade edilmisdir.

}

Massiv::Massiv(int s, int d)

{ if (s <= 0) Sehv("Menfi ededler olcu ola bilmez!"); if ((p = new int [Olcu = s]) == NULL)

Sehv("Massive yer ayrila bilmedi!"); for (; s;)

p[--s] = d;

//s yene saygac olaraq istifade edilmisdir.

}

Massiv::~Massiv()

{ printf("Yoxedici islemeye basladi.\n");

if (p != NULL) delete p; printf("Yoxedici isini dayandirdi.\n");

}

int Massiv::Getir(int i)

{ if (i < 0 || i > Olcu)

Sehv("Massivin olcusu xaricinde!"); return p[i];

}

void Massiv::Menimset(int i, int d)

{ if (i < 0 || i > Olcu)

Sehv("Massivin olcusu xaricinde!"); else p[i] = d;

}

void Massiv::Sehv(char\* msg)

{ fprintf(stderr, "\a%s\n", msg); exit(4);

}

double Massiv::Orta()

{ int i;

double t;

for (i = 0, t = 0; i < Olcu; i++) t += p[i];

return t / Olcu;

}

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Misallar \*\*\*\*\*\*\*\*\*\*\*\*\*\* Massiv A(3);

void Misal1()

{ printf("\n1-ci Misal\tLokal teyin etme\n\n"); Massiv B(5);

}

#pragma warn -aus

// 'C' is assigned a value that is never used

// mesajinin qarsini alir

void Misal2()

{ printf("\n2-ci Misal\tLokal gosterici teyin etme\n\n"); Massiv \*C = new Massiv(5);

}

#pragma warn +aus void Misal3()

{ printf("\n3-cu Misal\tLokal gosterici teyin etme ve silme\n\n"); Massiv \*D = new Massiv(5);

delete D;

}

main()

{ clrscr();

printf("main funksiyasi icra olunmaga basladi.\n");

Misal1();

Misal2();

Misal3();

printf("\nUzvlerin istifade olunmasi\n\n"); Massiv E(4);

int i;

for (i = 0; i < 3; i++) A.Menimset(i, rand());

for (i = 0; i < 4; i++) E.Menimset(i, rand());

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

for (i = 0; i < 3; i++)

printf("A[%d] = %d\n", i, A.Getir(i));

printf("\nOrta qiymetler\n\n A = %lf,\t E = %lf\n\n", A.Orta(), E.Orta());

printf("main funksiyasi icrasini tamamladi.\n\n"); return 0;

}

Proqram ç1x1§1

|  |  |
| --- | --- |
| main funksiyasi icra olunmaga basladi. | A üçün |
| 1-ci Misal Lokal teyin etme |  |
| Yoxedici islemeye basladi. | B üçün |
| Yoxedici isini dayandirdi. |  |
| 2-ci Misal Lokal gosterici teyin etme |  |
| 3-cu Misal Lokal gosterici teyin etme ve silme |  |
| Yoxedici islemeye basladi. | D üçün |
| Yoxedici isini dayandirdi. |  |
| Uzvlerin istifade olunmasi |  |
| A[0] = 346 |  |
| A[1] = 130 |  |
| A[2] = 10982 |  |
| Orta qiymetler |  |
| A = 3819.333333, E = 9364.500000 |  |
| main funksiyasn icrasnnn tamamladn. |  |

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

|  |  |
| --- | --- |
| Yoxedici islemeye basladi. | E üçün |
| Yoxedici isini dayandirdi. |  |
| Yoxedici islemeye basladi. | A üçün |
| Yoxedici isini dayandirdi. |  |

Bu tayindan sonra Massiv sinfinin müxtalif hallar1 üçün a§ag1dak1lar1 qeyd etmak olar:

Massiv X(10, 5);

10 elementli bir massiv obyekti yadda§da yerla§dirilir va har elementa 5 qiymati manimsadilir.

Massiv Y(20);

20 elementli massiv obyekti yarad1l1r va bütün elementlara ba§lang1c qiymat olaraq -1000 manimsadilir.

Massiv Z;

Heç bir ba§lang1c §art verilmadiyi üçün Z massivi üçün 100 elementlik yer ayr1l1r va har elementa -1000 qiymati manimsadilir.

Layihalandirici üzarina yüklamalar ba§lang1c §artlar müxtalif olmas1na baxmayaraq, eyni davran1§l1 hadisalari tayin etmak üçün çox istifada edilan bir yoldur. Layihalandirici xaricinda yoxedici funksiyas1ndan ba§qa

bütün üzv funksiyalar üzarina yüklama amaliyyat1n1

aparmaq mümkündür.

Bununla barabar üzv funksiyalar1n1n va layihalandiricinin parametrlarina aktiv qiymat vermak mümkündür. Bela ki, yuxar1da göstarilan misalda oldugu kimi üç layihalandirici tayin etmakdansa, sadaca

Massiv(int s, int d);

layihalandiricisini tayin edib, s va d-ya aktiv qiymat verilsa idi,

Massiv(int s = 100, int d = -1000);

kimi tayin olunmu§ kod eyni qalmaqla eyni i§i göra bilardi.

### Obyektlara Manimsatma

Bir qayda olaraq obyektlara manimsatmaya ehtiyac yoxdur. Obyektlar üzv funksiyalar tarafindan nazarat olunmal1d1rlar. 8gar obyekt üçün manimsatma operatoru “=” yenidan tayin edilmazsa, manimsatmalar layihalandiricinin çag1r1lmas1 §aklina çevrilarak tatbiq edilir. Bu da yaln1z obyekt ilk dafa yarad1larkan yerina yetirilir.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Masalan, Massiv X = 18; manimsadilmasi Massiv X(18); kimi istifada edilir. Bu da X-e 18 elementlik yer ayr1lmas1 va har elementa -1000 manimsadilmasi demakdir.

};

Adlar::Adlar(char\* ad, int yas)

{ strcpy(Ad, ad);

Yas = yas;

}

void Adlar::Deyisdir(char\* YeniAd)

{ strcpy(Ad, YeniAd); }

void Adlar::Yaz()

{ printf("\nObyekt\n\nAdi : %s\nYasi : %d\n", Ad, Yas); }

main()

{ clrscr();

Adlar A("Etibar Seyidzade", 35); Adlar B = A;

A.Yaz();

B.Yaz();

printf("\nA-nin adi deyisdirildi.\n\n"); A.Deyisdir("Seyidzade Etibar Vaqif oglu"); A.Yaz();

B.Yaz();

return 0;

}

8gar Massiv X = “Setir”; manimsadilarsa, Massiv X(“Setir”); müraciati yaranacaqd1r ki, bu da sahv olacaqd1r.

8gar bir obyekta eyni sinfa daxil olan ba§qa bir obyekt manimsatmak talab olunarsa, avvalca bela bir manimsatmanin olub olmayacag1na nazarat edilir. Tayin olunmu§sa, operator funksiya icra olunur. Tayin olunmam1§sa, layihalandirici siyah1s1nda bela bir layihalandiricinin olub olmamas1na bax1l1r. 8gar orada da yoxdursa, manimsadilacak obyektin üzv dayi§kanlari digarina bir-bir köçürülür.

Proqram ç1x1§1

Obyekt

Adi : Etibar Seyidzade Yasi : 35

Obyekt

//AD2.CPP

#include <conio.h>

#include <stdio.h>

#include <string.h>

class Adlar

{ char Ad[30]; int Yas;

public: Adlar(char\*, int);

void Deyisdir(char\*); void Yaz();

Adlar::Adlar(char\* ad, int yas)

{ Ad = new char[strlen(ad)+1];

if(!Ad) // if(Ad == NULL) menasinda abort();

strcpy(Ad, ad); Yas = yas;

}

Adlar::~Adlar()

{ if(Ad) // if(Ad !== NULL) menasinda

{ delete Ad; Ad = NULL;

}

}

void Adlar::Deyisdir(char\* YeniAd)

{ if(Ad) delete Ad;

Ad = new char[strlen(YeniAd)+1]; if(!Ad) abort();

strcpy(Ad, YeniAd);

}

void Adlar::Yaz()

{ printf("\nObyekt\n\nAdi : %s\nYasi : %d\n", Ad, Yas); }

main()

{ clrscr();

Adlar A("Etibar Seyidzade", 35); Adlar B = A;

A.Yaz();

B.Yaz();

printf("\nA-nin adi deyisdirildi.\n\n"); A.Deyisdir("Seyidzade Etibar Vaqif oglu"); A.Yaz();

B.Yaz();

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Adi : Etibar Seyidzade Yasi : 35

A-nin adi deyisdirildi. Obyekt

Adi : Seyidzade Etibar Vaqif oglu Yasi : 35

Obyekt

Adi : Etibar Seyidzade Yasi : 35

Misalda B = A manimsadilmasi ila A-n1n saxlad1g1 qiymatlar B-ya manimsadilmi§ olur.

Proqram a§ag1dak1 kimi yaz1larsa,

//AD3.CPP

#include <conio.h>

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

class Adlar

{ char \*Ad; int Yas;

public: Adlar(char\*, int);

~Adlar();

void Deyisdir(char\*); void Yaz();

};

Proqram ç1x1§1 a§ag1dak1 kimi olacaqd1r:

return 0;

}

Obyekt

Adi : Etibar Seyidzade Yasi : 35

Obyekt

Adi : Etibar Seyidzade Yasi : 35

A-nin adi deyisdirildi. Obyekt

Adi : Seyidzade Etibar Vaqif oglu Yasi : 35

Obyekt

Adi : Seyidzade Etibar Vaqif oglu Yasi : 35

burax1lmas1 isa çox güman ki, növbati marhalalarda böyük sahvlara yol açacaqd1r. Bela hallar1n meydana galmamasi üçün manimsatma operatoru tayin edilmali va ya layihalandiricilarin daxilina ba§qa bir layihalandirici da alava edilmalidir.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Göründüyü kimi heç bir problem yoxdur. Hatta A ila B üzvlari bir-biri ila alaqali olduqlar1 üçün misalda oldugu kimi birinin ad dayi§ikliyindan digarinin da

//AD4.CPP

#include <conio.h>

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

class Adlar

{ char \*Ad; int Yas;

public: Adlar(char\*, int); Adlar(Adlar&);

~Adlar();

void Deyisdir(char\*); void Yaz();

};

Adlar::Adlar(char\* ad, int yas)

{ Ad = new char[strlen(ad)+1];

if(!Ad) // if(Ad == NULL) menasinda abort();

strcpy(Ad, ad); Yas = yas;

}

Adlar::Adlar(Adlar&Obyekt)

{ Ad = new char[strlen(Obyekt.Ad)+1];

xabari olacaqd1r. Yox edilarkan Ad göstaricisinin göstardiyi yadda§ sahasi avvalca obyektlarin biri tarafindan, sonra da eyni saha ikinci obyekt tarafindan sarbast burax1l1r. Eyni sahanin iki dafa sarbast

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Proqram ç1x1§1

Obyekt

Adi : Etibar Seyidzade Yasi : 35

Obyekt

Adi : Etibar Seyidzade Yasi : 35

A-nin adi deyisdirildi. Obyekt

Adi : Seyidzade Etibar Vaqif oglu Yasi : 35

Obyekt

Adi : Etibar Seyidov Yasi : 34

if(!Ad) abort(); strcpy(Ad, Obyekt.Ad); Yas = Obyekt.Yas;

}

Adlar::~Adlar()

{ if(Ad) // if(Ad !== NULL) menasinda

{ delete Ad; Ad = NULL;

}

}

void Adlar::Deyisdir(char\* YeniAd)

{ if(Ad) delete Ad;

Ad = new char[strlen(YeniAd)+1]; if(!Ad) abort();

strcpy(Ad, YeniAd);

}

void Adlar::Yaz()

{ printf("\nObyekt\n\nAdi : %s\nYasi : %d\n", Ad, Yas); }

main()

{ clrscr();

Adlar A("Etibar Seyidzade", 35); Adlar B = A;

A.Yaz();

B.Yaz();

printf("\nA-nin adi deyisdirildi.\n\n"); A.Deyisdir("Seyidzade Etibar Vaqif oglu"); A.Yaz();

B.Yaz();

return 0;

}

Manimsatma ila alaqadar bu problem manimsatma olmasa bela, obyektlarin funksiyalara parametr kimi göndarilmasi zaman1 meydana galir.

long Axtar(Adlar X);

prototipli funksiyaya

adlar A(“Seyidzade Kenan”, 8);

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

kimi bir parametr ötürüldüyü zaman proqram sanki, X = A; manimsadilmasi ba§ vermi§dir kimi davranacaq va haqq1nda söhbat aç1lan problemlar meydana galacakdir. Buna göra da AD4.CPP proqram1nda oldugu kimi problemin qar§1s1n1 almaq laz1m galacakdir.



Obyektlarin tayin edilmasi üçün bu sababdan laz1m galan digar layihalandiricilarla barabar daha iki layihalandiri da olur:

1. **Standart layihafandirici (default constructor) -** Sinif\_adn(); §aklinda tayin edilan bu layihalandirici heç bir ilkin §art verilmadiyi halda istifada olunur;
2. **Köçürma layihafandiricisi (copy constructor) -** Sinif\_adn(Sinif\_adn&); §aklinda tayin edilan bu layihalandirici isa avvalcadan tayin edilmi§ bir hadisanin qiymatlarini yeni yarad1lmaqda olan hadisaya ötürür. 8vvalki obyektin bir nüsxasini

<1xar1r.

# IV F8SiL

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

## OBYEKTL8RiN XÜSUSiYY8TL8Ri

### Obyekt Üzvlari Olan Obyektlar

Bir obyektin üzv dayi§kanlari aras1nda digar obyektlar da ola bilar. istifada edilib edilmamasindan as1l1 olamayaraq bu obyektlarin digar dayi§kanlardan farqi yoxdur.

class A

{ int I;

...

public: A();

A(A&);

A(int);

...

...

};

class B

{ A a, b;

...

public: B();

B(B&);

B(int, int);

...

...

};

kimi tayin edilmi§ iki sinifdan birincisinin kodla§d1r1lmas1nda indiya qadar §arh etdiklarimiza alava edilacak yeni bir §ey yoxdur. ikinci sinfin kodla§d1r1lmas1nda isa

B::B()

{ ....

....

}

kimi bir layihalandiricinin kodla§d1r1lmas1nda a va b obyektlari üçün standart layihalandiricilar avtomatik olaraq çag1r1l1r. 8gar bunun yerina ba§qa bir layihalandiricinin istifada olunmas1 talab olunarsa,

B::B()

:a(33),

b(12) { ... }

B::B(int u, int v)

:a(u),

b(v) { ... }

§aklinda layihalandirici ba§l1g1n1n yaz1lmas1ndan sonra “:” i§arasi, üzv obyektlarin istanilan layihalandiricilari aras1na isa “,” qoyularaq yaz1la bilar. Layihalandiricisi

Y += dy; Z += dz; return 0;

}

int Noqte::Yaz()

{ printf("(%f, %f, %f)", X, Y, Z); return 0;

}

#define DIFSQR(p) ((p-Ikinci.##p)\*(p-Ikinci.##p)) float Noqte::Mesafe(Noqte& Ikinci)

{ return sqrt(DIFSQR(X) + DIFSQR(Y) + DIFSQR(Z)); }

#undef DIFSQR

class Duzxett

{ Noqte Baslangic, Bitis;

public:

Duzxett(Noqte&, Noqte&); Duzxett(Duzxett&);

float Uzunluq(); int Yaz();

};

Duzxett::Duzxett(Noqte& A, Noqte& B)

:Baslangic(A), Bitis(B)

{}

Duzxett::Duzxett(Duzxett& D)

:Baslangic(D.Baslangic), Bitis(D.Bitis)

{}

float Duzxett::Uzunluq()

{ return Baslangic.Mesafe(Bitis); }

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

tayin olunmayan obyektlar üçün isa standart layihalandirici avtomatik olaraq çag1r1lacaqd1r.

//DOST1.CPP

#include <conio.h>

#include <stdio.h>

#include <math.h>

class Noqte

{ float X, Y, Z;

public:

Noqte(float = 0, float = 0, float = 0); Noqte(Noqte&);

int Dasi(float, float, float); int Yaz();

float Mesafe(Noqte&);

};

Noqte::Noqte(float \_x, float \_y, float \_z)

{

X = \_x; Y = \_y; Z = \_z;

}

Noqte::Noqte(Noqte& N)

{

X = N.X;

Y = N.Y;

Z = N.Z;

}

int Noqte::Dasi(float dx, float dy, float dz)

{ X += dx;

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

duz xettin uzunlugu = 86.023253

### Friend (Dost) Tayinedicisi

int Duzxett::Yaz()

{ printf("["); Baslangic.Yaz(); printf("-"); Bitis.Yaz(); printf("]");

return 0;

}

main()

{ clrscr();

Noqte A;

Noqte B(30, 40, 50);

A.Yaz();

printf("\n");

B.Yaz();

printf("\n\niki noqte arasindaki mesafe = %f\n\n", A.Mesafe(B));

B.Dasi(10, -10, 60);

Duzxett D(Noqte(10, 20, 30), B);

D.Yaz();

printf("\n\nduz xettin uzunlugu = %f\n\n", D.Uzunluq());

return 0;

}

Bir sinfin üzvlarinin digar sinif va funksiyalardan qorunmas1 bazan manfi hallara gatirib ç1xar1r. Bu üzvlarin bütün istifadalara aç1lmas1 da bazan problemlar

<1xara bilar. Bu bax1mdan bir sinfin üzvlari qorunarkan bazi sinif va funksiyalardan qorunmay1b, xüsusi (private tayinli) üzvlari, bu sinif va funksiyalar tarafindan istifada oluna bilar. Qadagan olunmas1na baxmayaraq, bütün xüsusi üzvlara müraciat hüququ verilan funksiyalara **dost-funksiya** (friend-function), siniflara isa **dost-sinif** (friend-class) deyilir.

Proqram ç1x1§1

(0.000000, 0.000000, 0.000000)

(30.00000, 40.00000, 50.00000)

iki noqte arasnndaki mesafe = 70.710678

[(10.00000, 20.00000, 30.00000)-(40.00000, 30.00000, 110.00000)]

//DOST2.CPP

#include <conio.h>

#include <stdio.h>

#include <math.h>

class Noqte

{ float X, Y, Z;

public:

Noqte(float = 0, float = 0, float = 0); Noqte(Noqte&);

int Dasi(float, float, float);

friend int Yaz(Noqte&);

friend float Mesafe(Noqte&, Noqte&);

};

Yaz va Mesafe üzv funksiyas1 deyildir. Normal olaraq prototiplari Noqte sinfinin xaricinda tayin olunmal1d1r. Bu iki funksiya vazifalarina göra Noqte sinfinin xüsusi dayi§kanlarina müraciat etmali olduqlar1ndan bu sinif tarafindan friend olaraq tayin edilmi§dir.

DOST2.CPP davam1

Noqte::Noqte(float \_x, float \_y, float \_z)

{

X = \_x; Y = \_y; Z = \_z;

}

Noqte::Noqte(Noqte& N)

{

X = N.X;

Y = N.Y;

Z = N.Z;

}

int Noqte::Dasi(float dx, float dy, float dz)

{ X += dx;

Y += dy; Z += dz; return 0;

}

int Yaz(Noqte& N)

Yaz va Mesafe funksiyalar1 üzv funksiyas1 olmad1qlar1 üçün kodla§d1r1larkan “Noqte::” ifadasi ba§l1q daxilinda olmamal1d1r. Olarsa, bu sahv qabul edilacakdir.

{ printf("(%f, %f, %f)", N.X, N.Y, N.Z);

return 0;

}

float Mesafe(Noqte& Birinci, Noqte& Ikinci)

{ return sqrt((Birinci.X - Ikinci.X) \* (Birinci.X - Ikinci.X) + (Birinci.Y - Ikinci.Y) \* (Birinci.Y - Ikinci.Y) + (Birinci.Z - Ikinci.Z) \* (Birinci.Z - Ikinci.Z) );

}

DOST2.CPP davam1

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

class Duzxett

{ Noqte Baslangic, Son;

public:

Duzxett(Noqte&, Noqte&); Duzxett(Duzxett&);

float Uzunluq();

friend int Yaz(Duzxett&);

};

Duzxett::Duzxett(Noqte& A, Noqte& B)

:Baslangic(A), Son(B)

{}

Duzxett::Duzxett(Duzxett& D)

Yaz(D);

printf("\n\nduz xettin uzunlugu = %f\n\n", D.Uzunluq());

return 0;

}

Yaz va Mesafe üzv funksiyas1 olmad1qlar1 üçün istifada edilarkan Baslangic.Mesafe(Son); va ya Baslangnc.Yaz(Son); §aklinda istifada edilmadiyina diqqat edin.

:Baslangic(D.Baslangic), Bitis(D.Son)

{}

float Duzxett::Uzunluq()

{ return Mesafe(Baslangic, Son); }

int Yaz(Duzxett& D)

{ printf("["); Yaz(D.Baslangic); printf("-"); Yaz(D.Son); printf("]");

return 0;

}

DOST2.CPP davam1

Bu proqram icra olunmas1 bax1m1ndan avvalkindan farqlanmir. Sadaca proqram1n yaz1lmas1nda istifada olunan funksiyalar1n, istifada olunma mantiqi müxtalifdir. Bu cür tayinlar kitabxana yaratmaq üçün daha çox istifada oluna bilan funksiyalar1n yaz1lmas1na kömak edir.

Proqram ç1x1§1

(0.000000, 0.000000, 0.000000)

(30.00000, 40.00000, 50.00000)

iki noqte arasnndaki mesafe = 70.710678

[(10.00000, 20.00000, 30.00000)-(40.00000, 30.00000, 110.00000)]

duz xettin uzunlugu = 86.023253

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Bir sinfin ba§qa bir sinfi dost elan etmasi isa a§ag1dak1 kimi hayata keçirilir:

main()

{ clrscr();

Noqte A, B(30, 40, 50); Yaz(A);

printf("\n"); Yaz(B);

printf("\n\niki noqte arasindaki mesafe = %f\n\n", Mesafe(A, B));

B.Dasi(10, -10, 60);

Duzxett D(Noqte(10, 20, 30), B);

class A

{ friend class B;

...

...

...

};

class B

{ ....

....

....

operatoru qar§1s1nda tasirsizdir. Bu funksiyalar üzv funksiyalar olmaq macburiyyatindadirlar.

Digar bir farq da, dost funksiyalar1n dost olduqlar1 bir sinfin üzvlarina birba§a müraciat etmamalaridir.

};

class Noqte

{ float X, Y, Z;

public:

Noqte(float, float, float);

friend void Yaz(Noqte&);

};

Bir sinfin iki va daha art1q dost-sinif va dost- funksiyas1 ola bilar. Dost-sinif, dost-funksiya tayinlarinin private, public va ya protected saviyyalarinda olmas1n1n ela bir ahamiyyati yoxdur.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Dost-sinif tayinlarinin maqsadi dost elan edilan sinfin (masalan, B sinfi), dost elan edan sinfin (masalan, A sinfi) bütün xüsusi üzvlarina müraciat haqq1n1 tamin etmakdir. Belalikla, aralar1nda heç bir ox§arl1g1n olmamas1na baxmayaraq bir sinfin digar sinifdan istifada etmasi tamin olunmu§ olur. Bu haqq sadaca friend ila tayin olunmu§ siniflara verilir. Bu sinifdan töranan siniflarin va ya bu sinfin dost elan etdiyi digar dost-sinif va dost-funksiyalar1 eyni hüquqdan faydalana bilmazlar.

Dost-sinif tayini birtarafli tayindir. Yani, A B-ni dost elan etmi§dir, B A-n1n xüsusiyyatlarini istifada eda bilar. Lakin A B-nin xüsusiyyatlarini istifada eda bilmaz. Çünki, B A-n1 dost elan etmami§dir.

Dost-funksiya tayini layihalandirici (constructor) va yoxedici (destructor) funksiyas1 ila manimsatma “=”

tayinindan sonra

#include <stdio.h>

void Yaz(Noqte&)

{ printf(“(%d, %d, %d)”, X, Y, Z); }

kimi kodla§d1r1la bilmaz. Ona göra ki, Yaz funksiyas1n1n istifada etdiyi X, Y, Z dayi§kanlarinin na olduqlar1 balli deyildir. (Biza göra bu dayi§kanlar Noqte sinfinin dayi§kanlaridir. Lakin Yaz funksiyas1 bu dayi§kanlari birba§a istifada eda bilmaz. 8gar a§ag1dak1 kimi kodla§d1rma apar1larsa,

#include <stdio.h>

void Yaz(Noqte& \_noqte)

{ printf(“(%d, %d, %d)”, \_noqte.X, \_noqte.Y, \_noqte.Z); }

X, Y, Z-in \_noqte obyektinin üzvlari oldugu ba§a dü§ülacakdir; Bu üzvlarin private tayininin qar§1l1g1 olaraq Yaz funksiyas1n1n dost tayini bu xüsusi üzvlara müraciati tamin edir.

### Obyeklarin Operatorlara Yüklanmasi

Tayin olunmu§ yeni siniflarin operatorlara yüklanmasi mümkündür.

Vector sinfini tayin etdikdan sonra indi da iki vektorun cami va farqi üçün + va - operatorlar1n1 tayin edak.

Vector& operator+(Vector& A, Vector& B)

{ return Vector(A.X + B.X, A.Y + B.Y, A.Z + B.Z); }

Vector& operator-(Vector& A, Vector& B)

{ return Vector(A.X - B.X, A.Y - B.Y, A.Z - B.Z); }

Bu tayinlardan sonra

Vector A(3, 4);

Vector B(8, 6.5 / 3, 1); Vector C = A + B; Vector D = A - B;

float f = 67;

Vector E = A – Vector(16, 1, -10) + Vector(f, f \* 3, f / 4);

ifadalarini yazmaq olar.

Bir vektoru haqiqi adadla geni§ladan vurma operatorunu a§ag1dak1 kimi yaza bilarik:

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Vector& operator\*(Vector& A, float R);

{ return Vector(A.X \* R, A.Y \* R, A.Z \* R); }

Bu cür tayindan sonra

class Vector

{ public:

float X, Y, Z;

Vector(float = 0, float = 0, float = 0); Vector(Vector&);

};

Vector::Vector(float a, float b, float c)

{ X = a; Y = b; Z = c;

}

Vector::Vector(Vector& \_vector)

{ X = \_vector.X; Y = \_vector.Y; Z = \_vector.Z;

}

Vector A(4, 5, 3);

Vector B = A \* 2;

Vector(float = 0, float = 0, float = 0); Vector(Vector&);

friend Vector& operator+(Vector&, Vector&); friend Vector& operator-(Vector&, Vector&); friend Vector& operator\*(Vector&, Vector&); friend Vector& operator\*(Vector&, float); friend Vector& operator\*(float, Vector&); friend Vector& operator-(Vector&);

};

Vector& operator+(Vector& A, Vector& B)

{ return Vector(A.X + B.X, A.Y + B.Y, A.Z + B.Z); }

dogru

Vector C = 2 \* A;

isa sahvdir. Çünki, Vector& operator\*(Vector&, float); ila Vector ila float qiymatinin hasili tayin edilmi§dir. float ila Vector qiymatlarinin hasili ayr1l1qda tayin edilmalidir.

inline Vector& operator\*(float R, Vector& A)

{ return A \* R; }

Bir vektorun istiqamatini tars çeviran unary

operatorunu tayin etmak üçün

Operatorlar1n ilk parametrlarinin sinfin özünün olmas1 hal1nda onlar1 üzv funksiya kimi tayin edib istifada etmak da mümkündür.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Vector& operator-(Vector& A)

{ return Vector(-A.X, -A.Y, -A.Z); }

yazmaq olar.

class Vector

{ float X, Y, Z;

public:

Vector(float = 0, float = 0, float = 0); Vector(Vector&);

Vector& operator+(Vector&); Vector& operator-(Vector&); Vector& operator\*(Vector&); Vector& operator\*(float);

friend Vector& operator\*(float, Vector&); Vector& operator-();

};

Vector& Vector::operator+(Vector& B)

{ return Vector(X + B.X, Y + B.Y, Z + B.Z); }

Bir sinfin xüsusi (private) üzvlarinin da operator funksiyalar1 daxilinda istifada olunmas1 talab olunarsa, bu operator funksiyalar1n1n dost (friend) kimi tayin olunmas1 laz1md1r.

class Vector

{ float X, Y, Z;

public:

Vector& Vector::operator-()

{ return Vector(-X, -Y, -Z); }

return 0;

}

### this Lokal Dayi§kani

Bu misalda bundan avvalki misala alava olaraq Hesab üzv funksiyas1ndan istifada edilmi§dir. Yaz funksiyas1 yaln1z üzvün koordinatlar1n1, Hesab funksiyas1 isa “Bu noqte (x, y, z) movqeyindedir” malumat1n1 ekrana ç1xaracaqd1r. Hesab funksiyas1 nöqtanin koordinatlar1n1 ekrana ç1xararkan Yaz funksiyas1n1 çag1rmal1, çag1rarkan da yaz1lacaq nöqtani parametr olaraq göndarmalidir.

Burada Yaz funksiyas1na parametr vermak üçün ox§ar bir obyekt yarad1l1r. Bu amaliyyat vaxt va alava yadda§ talab edir. Buna baxmayaraq layihalandiricilar har zaman ox§ar obyekt yarada bilmirlar.

//THISNKT.CPP

#include <stdio.h> class Noqte

{ float X, Y, Z; public:

Noqte(float, float, float);

friend void Yaz(Noqte&); void Hesab();

};

Noqte::Noqte(float x, float y, float z)

{ X = x; Y = y; Z = z;

}

void Noqte::Hesab()

{ printf("Bu noqte ");

Yaz(Noqte(X, Y, Z)); printf(" movqeyindedir\n");

}

void Yaz(Noqte& \_noqte)

{ printf("(%f, %f, %f)", \_noqte.X, \_noqte.Y, \_noqte.Z); }

main()

{ Noqte A(1, 2, 3);

A.Hesab();

Lakin Hesab va buna ox§ar bütün funksiyalar hans1 obyekta aid olduqlar1 haqq1nda malumata malikdirlar. Bu malumat üzv funksiyas1n1n aid oldugu obyektin göstaricisi malumat1ndan ibaratdir. Bu göstaricinin ad1 har bir obyekt üçün this qabul olunur. Bu dayi§kan üzv funksiyalar1 üçün avtomatik olaraq tayin olunur. Baxd1g1m1z misalda Hesab funksiyas1 üçün bu tayin Noqte \*const this §aklindadir.

istifadasi a§ag1dak1 kimidir:

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

void Noqte::Hesab()

{ printf("Bu noqte ");

this göstaricisindan istifada edarak obyektin üzv funksiya va dayi§kanlarina müraciat etmak da mümkündür. Hatta, bazan macburidir. Üzv funksiya daxilinda obyektinkilarla eyni adda olan dayi§kanlar varsa, bu funksiya obyektin eyni adl1 dayi§kanlarina müraciat eda bilmaz. Bu halda **görma** (scope::) operatorundan da istifada etmak olmaz. Bela hallarda this dayi§kanindan istifada edilir.

Yaz(\*this);

printf(" movqeyindedir\n");

}

istak sinif tayini daxilinda bildirilir. Bu dayi§kanin tayini zaman1 tipdan avval static sözü alava edilarak yerina yetirilir.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Noqte::Noqte(float X, float Y, float Z)

{ this->X = X; this->Y = Y; this->Z = Z;

}

### Ümumi Ortaq Dayi§kanlar

//STATIC1.CPP

#include <conio.h>

#include <stdio.h>

class Misal

{ int a; static int b;

public:

Misal(int a, int b)

{ this->a = a; this->b = b;

}

int De\_a()

{ return a; } int De\_b()

{ return b; }

};

int Misal::b; main()

{ clrscr();

Misal A(5, 10), B(8, 15);

printf("A a = %d, b = %d\n", A.De\_a(), A.De\_b());

printf("B a = %d, b = %d\n", B.De\_a(), B.De\_b());

printf("A a = %d, b = %d\n", A.De\_a(), A.De\_b()); return 0;

}

Har obyekt malik oldugu malumatlar1 bütün obyektlardan qoruyur. 8gar bazi malumatlar1n bütün proqramla§d1rma üzvlari tarafindan sarbast istifada edilmasi talab olunarsa, bu dayi§kanlar qlobal dayi§kanlar kimi tayin edilir.

8gar bir dayi§kanin müayyan bir sinfa mansub olan obyektlar tarafindan istifada edilmasi talab olunarsa, bu

Proqram ç1x1§1

A a = 5, b = 15; B a = 8, b = 15; A a = 5, b = 15;

Bu tayin ila b dayi§kaninin bütün Misal obyektlarinda ortaq istifada edilacayi, digar obyektlarin isa bu dayi§kana birba§a müraciat eda bilmayacayi bildirilir. Lakin bu tayin kifayat deyildir. b dayi§kani bundan ba§qa sinif xaricinda da

int Misal::b;

§aklinda tayin edilmalidir.

Bir qayda olaraq dayi§kanlarin tayini a§ag1dak1 kimidir:

class sinif\_adn

{ static tip1 dayi$kan1; static tip2 dayi$kan2;

...

Obyektlar üçün istifada olunan static açar sözü C- daki lokal dayi§kanlar ila eyni mantiqa malikdir. Yani obyekt, proqram icra olunmaga ba§lad1g1 zaman varl1g1n1 ortaya qoyur: yadda§dan yer istayir va ilk qiymatini al1r. Proqram icras1n1 tamamlad1qdan sonra varl1g1na son qoyulur. Lokal tayinlar kimi funksiya icra olunmaga ba§lad1g1 anda varl1g1n1 göstarib, icras1n1 tamamlad1qdan sonra varl1g1na son vermir. Eyni formada üzv dayi§kanlari da obyekt mövcud oldugu anda (layihalandirici icra olunduqdan sonra) varl1g1n1 göstarib, obyekt yox oldugu anda (yoxedici icra olunduqdan sonra) varl1qlar1na son qoyulur. static ila tayin edilan bu üzvlar da eyni §akilda obyektlarin ömür müddatindan as1l1 qalmayaraq, proqram1n ömür müddatindan as1l1 hala galirlar.

static dayi§kanlar ola bildiyi kimi, static obyektlarin olamas1 da mümkündür.

};

int Mesafe(float a)

{ static Noqte A(3, 7, -12);

...

}

tip1 sinif\_adn::dayi$kan1; tip2 sinif\_adn::dayi$kan2;

...

8gar laz1m galarsa, bu dayi§kanlara ba§lang1c qiymat da manimsatmak olar.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

int Misal::b = 30;

Yaln1z burada diqqat edilmasi vacib olan hal, har hans1 bir obyekt mövcud olmazdan avval, static obyektlar mövcud olacag1 üçün bu obyektlar yarad1larkan mövcud olmayan dayi§kanlardan istifada olunmamal1d1r.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

### Statik (Static) Funksiyalar

int Mesafe(float a)

{ static Noqte A(a, 7, -a); //Sehv. a-nin qiymeti yoxdur.

...

Noqte B(a, 7, -a);

...

//Dogru. B static deyildir.

}

Üzv dayi§kanlar kimi üzv funksiyalar da static olaraq tayin edila bilarlar. Bu tayin bax1m1ndan üzv funksiyalar1n1 iki müxtalif qrupa ay1rmaq olar: **statik** (static) va **avtomatik** (auto) funksiyalar. Normal olaraq har hans1 bir tayin olmazsa, tayin olunmayan bütün üzv funksiyalar1 avtomatik funksiya kimi qabul edilir. Avtomatik funksiyalar1n asas xüsusiyyati, üzv olduqlar1 sinfin bütün üzv dayi§kan va funksiyalar1n1 birba§a istifada eda bilmalaridir. Virtual funksiyalar da daxil olmaqla, indiya kimi tayin edib istifada etdiyimiz bütün üzv funksiyalar1n1n avtomatik funksiya oldugunu xat1rlayaraq, yaz1larkan üzv dayi§kan va funksiyalar1n1 neca istifada etdiklarina diqqat edin.

Statik funksiyalar da üzv funksiyas1 kimi tayin edilib istifada olunmalar1na baxmayaraq avtomatik funksiyalardan iki alamatina göra farqlanirlar. Birincisi, statik funksiyalar üzvü olduqlar1 sinfin üzv dayi§kan va

funksiyalar1ndan avtomatik olanlar1 birba§a istifada eda bilmazlar. ikincisi, statik funksiyalar1 çag1rmaq üçün bir obyekta ehtiyac vard1r.

Statik funksiyalar sinif daxilda tayin edilarkan funksiya prototipinin avvalina static açar sözü alava edilir. Funksiyan1n gövdasinin yaz1lmas1 isa digar funksiyalar kimidir. Yaln1z sinfin üzv dayi§kan va funksiyalar1ndan statik kimi tayin edilmayanlar birba§a istifada oluna bilmazlar. Bundan ba§qa this lokal dayi§kani da funksiyalar1n daxilinda istifada oluna bilmaz.

class sinif\_adn

{ ...

static funksiya\_tipi funksiya\_adn(parametr\_siyahnsn);

...

};

{ ...

funksiya\_tipi sinif\_adn::funksiya\_adn(parametr\_siyahnsn);

...

...

}

Statik funksiyalar iki formada çag1r1la bilar: 8gar statik funksiyan1n aid oldugu sinfin bir obyekti mövcuddursa, bu obyekta asaslanaraq bir statik funksiya sanki, avtomatik bir funksiya kimi çag1r1la bilar. Va ya statik funksiyan1n ad1ndan avval aid oldugu sinfin ad1n1

//Static funksiya avtomatik funksiyadan cagrilir

Static\_Message();

}

//statik funksiyanin yazilmasi void Static\_Misal::Static\_Yaz()

{ /\*

printf("x = %d\t", X);

//Error: Memeber X cannot be used without an object

//Sehv: X uzvu obyekt olmadan istifade oluna bilmez printf("\*\* %d \*\*\n", X\_Misal());

//Error: Use . or -> to call 'Static\_Misal::X\_Qiymeti()'

//Sehv: 'Static\_Misal::X\_Qiymeti()' cagirilarken

//. ve ya -> istifade edin

//Qeyd: Bunun ucun da yeni bir obyekte ehtiyac vardir.

\*/

printf("\ny = %d\n", Y);

//Static funksiyasi static funksiyasindan cagirilir Static\_Message();

}

void Static\_Misal::Static\_Message()

{ printf("\nStatic\_Misal, Static\_Message funksiyalari\n"); }

main()

{ clrscr(); Static\_Misal A(8);

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

va :: görma (scope) operatorunu yazaraq, statik funksiyan1 çag1rmaq mümkündür.

//STATIC.CPP

#include <conio.h>

#include <stdio.h>

class Static\_Misal

{ int X; static int Y;

public: Static\_Misal(int);

int X\_Qiymeti(); void Auto\_Yaz();

static void Static\_Yaz();

static void Static\_Message();

};

int Static\_Misal::Y = 300; Static\_Misal::Static\_Misal(int x)

{ X = x; }

int Static\_Misal::X\_Qiymeti()

{ return X; }

//avtomatik funksiyanin yazilmasi void Static\_Misal::Auto\_Yaz()

{ printf("x = %d\t", X);

printf("\*\* %d \*\*\n", X\_Qiymeti()); printf("\ny = %d\n", Y);

Proqram ç1x1§1

A.Auto\_Yaz(); A.Static\_Yaz();

printf("\n");

//Static\_Misal::Auto\_Yaz();

//Error: Use . or -> to call "Static\_Misal::Auto\_Yaz()'

//Sehv: 'Static\_Misal::Auto\_Yaz()' cagrilarken

//. ve ya -> istifade edin

Static\_Misal::Static\_Yaz(); return 0;

}

### const Funksiyalari

Bir obyektdan bir malumat1 almaq üçün istifada olunan üzv funksiyalar1n1n, yaz1larkan sahvan da olsa, üzv dayi§kanlarini dayi§dirmasi arzuolunmazd1r. Daxilinda bu cür funksiyalar1n tayin edilmasi zaman1 bu istak parametr siyah1s1ndan sonra const (sabit) açar sözü yaz1laraq bildirilir. Tayin edilarkan sonunda const (sabit) açar sözü olan funksiyalar ***malumat funksiyasi*** adland1r1l1r. Bu funksiyalar üzv dayi§kanlarinin qiymatlarini dayi§dira bilmadiklari kimi, funksiyas1 olmayan digar funksiyalar1 da çag1ra bilmazlar.

class sinif\_adn

{ ...

int const\_funksiyasn(parametr\_siyahnsn) const;

...

}

int sinif\_adn::const\_funksiyasn(parametr\_siyahnsn) const

{ ...

return qiymat;

}

x = 8 \*\* 8 \*\*

y = 300

Static\_Misal, Static\_Message funksiyasi y = 300

Static\_Misal, Static\_Message funksiyasi

y = 300

Static\_Misal, Static\_Message funksiyasi

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

### iç-iça Tayinlar

Sinfin tayin edilmasi zaman1 yeni bir sinif va ya tip da tayin etmak olar. Tayin edilan bu yeni tip, tayin olundugu sinfin ad1 ila barabar istifada edilir. Bu da tayin

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

olundugu sinfin ad1ndan sonra görma (:: scope) operatoru va ad1n yaz1lmas1 ila yerina yetirilir.

//Massiv xaricinde teyin olunan Uzv obyekti

//Her iki teyinetmeye diqqet edin public:

Massiv(char \*, int, int);

};

Massiv::Massiv(char\* n, int t, int s)

{ for(int i = 0; i < 10; i++)

{ A[i].X = t;

A[i].Y = s;

B[i].Ad = n;

B[i].X = s;

}

}

main()

{ clrscr(); Massiv::Uzv K; K.Yaz();

//K Massiv daxilinde teyin edilen obyektdir.

Uzv H; H.Yaz();

//H Massiv xaricinde teyin edilen obyektdir.

//Burada ::Uzv H seklinde istifade edile biler.

return 0;

}

sinif\_adn::tip\_adn sinif\_adn::enum\_qiymati

Proqram ç1x1§1

Massiv::Uzv 0 0

::Uzv XxXxX 0

//ARRAYNST.CPP

#include <conio.h>

#include <stdio.h> struct Uzv

{ int X; char \*Ad;

Uzv(char\* ad = "XxXxX", int a = 0)

{ Ad = ad; X = a; }

void Yaz()

{ printf("::Uzv %s %d\n", Ad, X); }

};

class Massiv

{ public:

struct Uzv

{ int X, Y;

Uzv(int a = 0, int b = 0)

{ X = a; Y = b; }

void Yaz()

{ printf("Massiv::Uzv %d %d\n", X, Y); }

};

Uzv A[10]; //Massiv daxilinde teyin olunan Uzv obyekti

::Uzv B[10];

### Obyekt Göstaricilari

Bir obyektin göstarici kimi tayin olunmas1 C-da

oldugu kimidir:

Massiv \*A; Noqte \*C; Noqte \*\*D;

Bu cür tayinlarda obyekt deyil, sadaca, obyekti göstaran bir göstarici vard1r. Buna göra da tayin zaman1 layihalandiricinin, proqram bloku tamamland1qdan sonra da yoxedicinin icra olunacag1n1 gözlamak olmaz. Bu istak proqramç1 tarafindan tayin olunur.

main()

{ Massiv massiv(20, 10); Massiv \*massivPtr = &massiv;

...

...

return 0;

} //massiv obyekti özü yox olur.

8gar obyekt göstaricilari üçün yadda§da dinamik olaraq bir yer ayr1larsa, bunun üçün new va ayr1lm1§ bu

Massiv \*massivPtr;

massivPtr = new Massiv(30, 40);

...

...

delete massivPtr;

...

...

int PaketSayi = 30;

int PaketUzunlugu = 6\*;

massivPtr = new Massiv(PaketSatisi \*PaketUzunlugu, 0);

...

...

delete massivPtr; delete massiv;

return 0;

}

new operatoru yer ay1rd1g1 obyektin layihalandiricisini, delete operatoru da bu obyektin yoxedicisini avtomatik çal1§d1r1r. Buna göra da new operatoru istifada edilarkan obyektin layihalandiricilarindan biri new operatorundan sonra yaz1l1r. delete opertorunun istifada edilmasinda isa sadaca obyekt göstaricisi verilir. Burada new ila yer ayr1lmam1§ obyektlari delete ila, new ila yer ayr1lm1§ olsalar bela, eyni obyekti bir dafadan art1q silmak olmaz.

obyektin silinmasi üçün da delete operatorunun istifada edilmasi laz1md1r.

Massiv massiv(20, 30);

…

delete massiv;

Massiv\* massivPtr = new Massiv(20, 30);

...

delete massivPtr; delete massivPtr;

main()

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

{ Massiv \*massiv = Massiv(10, 20);

Yuxar1da göstarilanlarin har biri sahva yol aça bilar.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

### Obyekt Massivi

Massiv \*C = new Massiv[10];

//50 elementden ibaret Massivlerin massivi

\*(A -> f + 5) = 1.00;

//A-nin gosterdiyi Massivin 5-ci elementi

\*(B -> f + 5) = 2.00;

//B-nin gosterdiyi Massivin 5-ci elementi

\*((C + 8) -> f + 5) = 3.00;

//C-nin gosterdiyi massivlerin 8-ci sirasindaki

//Massivin 5-ci elementi

\*(C[8].f + 5) = 4.00;

//C-nin gosterdiyi massivlerin 8-ci sirasindaki

//Massivin 5-ci elementi delete A;

//Bir Massiv obyekti silinir. (20 elementli) delete B;

//Bir Massiv obyekti silinir. (50 elementli) delete []C;

//10 Massiv obyekti silinir. (Her biri 50 elementli)

return 0;

}

**Standart** (default) layihalandiriciya malik olmaq

§arti ila bir sinfin obyektlarindan ibarat olan massiv alda

etmak mümkündür.

Burada delete [ ]C; yerina delete [10]C; kimi C massivinin ölçüsü da yaz1la bilar. Lakin bu amr massivi tamamila sildiyi üçün mötarizalarin daxilinda verilan massivin ölçüsünü nazara almayacaqd1r. Va komplyator bunu Siza “Array size for ‘delete’ ignored” malumat1 ila bildiracakdir. Bu sahvdan qurtulmaq üçün ”#pragma warn dsz-“ direktivindan istifada eda bilarsiniz.

//MASARRAY.CPP

#include <conio.h>

#include <stdio.h>

class Massiv

{ public:

int Olcu; float \*f;

Massiv(int n = 50);

~Massiv();

};

Massiv::Massiv(int n)

{ static say = 0;

f = new float(Olcu = n); printf("%d\tolcu %d\n", ++say, Olcu);

}

Massiv::~Massiv()

{ printf("Yoxedici\tolcu = %d\n", Olcu); delete f;

}

main()

{ clrscr();

Massiv \*A = new Massiv(20);

//20 elementlik bir Massiv gostericisi Massiv \*B = new Massiv;

//50 elementlik bir Massiv gostericisi

Burada delete [ ]C; yerina delete C; amri istifada edilarsa, C-nin ilk elementi silinacakdir ki, bu da xo§agalmaz hallar1n yaranmas1na sabab olacaqd1r. Buna

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

göra da delete amrindan bu §akilda istifada etmak

main()

{ clrscr(); int i;

//Gostericiler ucun yer ayirma Massiv \*\*A = new Massiv\*[MasOlcu];

//Her bir gostericinin gosterdiyi obyekti yaratmaq ve menimsetmek for(i = 0; i < MasOlcu; i++)

A[i] = new Massiv(i \* 2 + 10);

//Istifade edilmesi

\*(A[2] -> f + 5) = 1.0;

//Her bir obyekti bir-bir silme emeliyyati for(i = MasOlcu - 1; i >= 0; i--) delete A[i];

//Gosterici sahelerini silme emeliyyati delete []A;

return 0;

}

maqsadauygun deyildir.

8gar istifada etmak istadiyiniz obyektin standart

layihalandiricisi yoxdursa va ya yarad1lacaq obyektlar bir-birindan farqli xüsusuyyatlar da§1yacaqlarsa, bu halda obyekt göstaricilari massivindan istifada etmak maqsadauygundur.

//MASPTRAR.CPP

#include <conio.h>

#include <stdio.h>

class Massiv

{ public:

int Olcu; float \*f;

Massiv(int n = 50);

~Massiv();

};

Massiv::Massiv(int n)

{ static say = 0;

f = new float(Olcu = n); printf("%d\tolcu %d\n", ++say, Olcu);

}

Massiv::~Massiv()

{ printf("Yoxedici\tolcu = %d\n", Olcu); delete f;

}

#define MasOlcu 10

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

**V F8SiL 5.2 Siniflarin Töradilmasi**

//MESAJ.CPP

#include <stdio.h>

#include <string.h>

class Mesaj

{ char \*mesaj[5];

public: Mesaj();

~Mesaj();

int Deyisdir(int, char\* = ""); int Xeber(int);

};

Mesaj::Mesaj()

{ int i;

for(i = 0; i < 5; i++)

{ mesaj[i] = new char[1]; strcpy(mesaj[i], "");

}

}

Mesaj::~Mesaj()

{ int i;

for(i = 0; i<5; i++) delete mesaj[i];

}

int Mesaj::Deyisdir(int i, char \*mes)

{ if(i < 0 || i > 5) return -1;

delete mesaj[--i];

mesaj[i] = new char[strlen(mes) + 1];

## OBYEKT TÖR8TM8K

### 5.1. Töratma 8maliyyati

Bir sinfa asaslanaraq onun yerina yetirdiyi i§in mahiyyatini dayi§dirmak va ya inki§af etdirmak kimi icra olunan amaliyyatlara **töratma** (derivation) deyilir. Töratma amaliyyat1n1n yerina yetirilmasi üçün tayin olunmu§ bir sinif mövcud olmal1d1r. Töratma amaliyyat1nda istifada olunan bu mövcud sinfa **baza sinfi** (base class) deyilir. Töratma naticasinda meydana galan obyekta isa **töranmi§ sinif** (derivated class) ad1 verilir. Ba§qa sözla baza sinfina **valideyn** (parent), töranmi§ sinfa isa **U§aq** (child) ad1 verilir.

A§ag1da göstarilmi§ iki metodla sinif töratmak olar:

1. Bir sinfin üzv funksiyalar1ndan birinin

vazifalarini ba§qa bir formada yerina yetiracak

§aklinda dayi§dirmakla;

1. Obyektin vazifalarini art1raraq yeni üzv funksiyalar1 alava etmak, yani inki§af etdirmakla.

Obyekt töradarkan bu iki metodun har ikisi ayr1- ayr1l1qda va ya barabar istifada oluna bilar.

Yoxlama xeberdarligi

Daxili sehv: Xeberdarliq mesajinin nomresi sehvdir Cixis xeberdarligi

Yoxlama xeberdarligi

Bu siniflarin tatbiqi naticasinda a§ag1dak1 proqram1 yazaraq növbati naticani alda etmak olar:

strcpy(mesaj[i], mes); return 0;

}

int Mesaj::Xeber(int n)

{ if(n < 0 || n > 5)

{ printf("\nDaxili sehv: Xeberdarliq mesajinin nomresi sehvdir\n"); return -1;

}

printf("%s\n", mesaj[n-1]); return 0;

}

indi isa xabardarl1q masaj1n1n nömrasi tak isa, proqram Enter düymasini s1xana qadar gözlasin, cüt isa gözlamadan icras1n1 tamamlas1n. Xeber hissasi istisna olmaqla bu strukturun digar funksiyalar1 eyni qals1n. Bu halda Mesaj sinfini baza kimi qabul edib Sehv sinfini töratmak daha maqsadauygundur.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

//MMESAJ.CPP

#include <conio.h>

#include "mesaj.cpp" Mesaj Xeberd;

main()

{ clrscr();

Xeberd.Deyisdir(1, "\nYoxlama xeberdarligi"); Xeberd.Deyisdir(2, "\nCixis xeberdarligi"); Xeberd.Xeber(1);

Xeberd.Xeber(21); Xeberd.Xeber(2); Xeberd.Xeber(1); return 0;

}

Proqram ç1x1§1

//SEHV.CPP

#include "mesaj.cpp"

#include <conio.h>

#include <stdlib.h>

struct Sehv : Mesaj

{ Sehv() { }

int Xeber(int);

};

int Sehv::Xeber(int N)

{ int Err = Mesaj::Xeber(N);

if(N % 2)

{ printf("\nDavam etmek ucun ENTER duymesini sixin."); while(getch() != 13);

printf("\n");

}

else

{ printf("\nProqram icrasinn davam etdire bilmir.\n"); exit(N);

}

return Err;

}

Yeni sinfin tayin edilmasi zaman1 baza sinfinin asas sinif ad1ndan sonra yaz1ld1g1na diqqat edin. class Sehv : Mesaj tayini Sehv sinfinin Mesaj sinfindan törandiyini göstarir. Digar bir halda agar laz1m galarsa, yeni sinfin layihalandirici va yoxedici funksiyalar1 digar üzv funksiyalar1ndan farqli olaraq töranmi§ olduqlar1 sinfin layihalandirici va ya yoxedici funksiyalar1n1 avtomatik olaraq istifadaya verirlar.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Sehv sinfi üçün Xeber funksiyas1 yenidan yaz1larkan Mesaj sinfinin Xeber funksiyas1na ehtiyac1 oldugu zaman, bu ehtiyac Mesaj sinfinin Xeber funksiyas1n1n Mesaj::Xeber §aklinda çag1r1lmas1 ila ödanilmi§ olur. Bu çag1r1§1 yerina yetirmak vacib deyildir. Eyni formada “u§aq” sinfinin “valideyn” sinfindan olan bir funksiyan1 çag1rmas1 asas1nda çag1r1lan funksiyan1n ad1 öz funksiyalar1n1n ad1 ila üst-üsta dü§ürsa, bu funksiyan1 asas sinfin “valideyn”larindan olmaq §arti ila

:: (görma) operatorundan istifada edarak çag1rmaq mümkündür. Mesaj::Deyisdir() kimi.

indi da bundan avvalki misal1 Sehv sinfi ila yenidan yazaq:

Proqram ç1x1§1

//MSEHV.CPP

#include <conio.h>

#include "sehv.cpp" Sehv Xeberd;

main()

{ clrscr();

Xeberd.Deyisdir(1, "\nYaxlama xeberdarligi"); Xeberd.Deyisdir(2, "\nCixis xeberdarligi"); Xeberd.Xeber(1);

Xeberd.Xeber(21); Xeberd.Xeber(2); Xeberd.Xeber(1); return 0;

}

Yaxlama xeberdarligi

Davam etmek ucun ENTER duymesini sixin. Daxili sehv: Xeberdarliq mesajinin nomresi sehvdir Davam etmek ucun ENTER duymesini sixin.

Cixis xeberdarligi

Proqramin icrasi tamamlandi.

Bu misalda Sehv sinfi üçün Deyisdir funksiyas1n1n varm1§ kimi istifada edilmasina diqqat edin. Haqiqatda isa Sehv sinfi Deyisdir vazifasini Mesaj sinfinin tatbiq

etdiyi metoddan istifada edarak yerina yetirir. Xeber vazifasi oldugu zaman isa Sehv bunu öz metodlar1na göra yerina yetirir. Sehv vazifasini yerina yetirarkan Mesaj-dan da faydalana bilir.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

### Müraciat Haqlari va Nüfuz Etma

Bir sinfin üzvlarina (dayi§kan va metodlar1na) müraciat eda bilan xarici agentlar üç sinfa bölünür. Bir töradilmi§ sinif, baza sinfi daxilindaki **qorunmu§** (protected) va tabii ki, **ümumi** (public) kimi tayin edilmi§ üzvlara müraciat hüququna malikdir.

Töradilmi§ sinif baza qabul edilarak yeni bir obyektin töradilmasi talab olundugunda, son töradilan sinif ilk töradilan sinfa bu sinfin müayyan etdiyi hüquqlar ila müraciat eda bilacakdir. Lakin bu halda asas bazan1n üzvlarina müraciat hüququ neca olacaqd1r?

Burada töradilan sinfin, özündan töranan siniflara avvalki siniflardan qalan miras1 neca tahvil veracayini müayyan etmak laz1md1r. Bu o sinfin nüfuz etma qabiliyyatini müayyan edir.

class A { ... };

class B : public A { ... };

class C : private A { ... };

class D : A { ... };

class E : A { ... };

Burada A tayin edilmi§ baza sinfidir. B, C, D va E isa A-dan töranmi§ siniflardir. B-nin tayin edilmasi zaman1 A-n1n avvalindaki public ifadasi B-nin A-dan tahvil ald1g1 miras1 eyni hüquqlarla (A-n1n B-ya verdiyi hüquqlarla) özündan sonrak1lara tahvil vermasi manas1na galir.

C da A-dan töranmasina ragman onun tayin edilmasi zaman1 private ifadasindan istifada edildiyi üçün C A-dan ald1g1 bütün xüsusiyyatlari özündan töranan siniflara private kimi tahvil veracakdir. Burada A-n1n public xüsusiyyatlarinin bela B-dan törananlar üçün istifada edilmasinin qadagan olmas1na diqqat edin.

|  |  |
| --- | --- |
| Nüfuz etma public | |
| public | public |
| protected | protected |
| private | private |
| Nüfuz etma private | |
| public | private |
| protected | private |
| private | private |

D va E siniflarinin tayin edilmasi a§ag1dak1 kimidir:

class D : public A { ... };

class E : private A { ... };

Buna sabab, avvalcadan qeyd edildiyi kimi class vasitasila edilan tayinlarda müraciat hüququ müayyan edilmadiyi zaman private, struct vasitasila edilan

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

tayinlarda isa public oldugunun avtomatik qabul edilmasidir.

Toremis(int e) : Baza(e)

{ Z = X + Y; }

void Goster()

{ printf("Toremis X = %d\tY = %d\tZ = %d\n", X, Y, Z); }

};

main()

{ clrscr();

Baza A(1); Toremis B(2); Baza \*C;

printf("A.Goster\n\t"); A.Goster(); printf("B.Goster\n\t"); B.Goster();

printf("C = &A\nC->Goster\n\t"); C = &A;

C -> Goster();

printf("C = &B\nC->Goster\n\t"); C = &B;

C -> Goster();

return 0;

}

### Dinamik Yüklama

Baza sinfi va töradilmi§ sinif aras1ndak1 ahamiyyatli olan alaqalardan biri da baza sinfinin göstaricisina töradilmi§ sinfin ünvan1n1n manimsadila bilmasidir. Bu **dinamik yüklama** (dinamic binding) qaydas1n1n meydana galmasini tamin edan bir hadisadir. Lakin bu manimsatma dinamik yüklamanin meydana galmasi üçün kifayat deyildir. Bunu bir misal ila izah edak:

Proqram ç1x1§1

A.Goster

Baza X = 1 B.Goster

Toremis X = 2

C = &A

C->Goster

Y = 2

Y = 4

Z = 6

//CLASSPOI.H

#include <stdio.h>

#include <conio.h>

class Baza

{ protected: int X, Y;

public: Baza(int C)

{ X = C; Y = 2 \* C; }

void Goster()

{ printf("Baza X = %d\tY = %d\n", X, Y); }

};

class Toremis:public Baza

{ public:

int Z;

sahib olan sinifdan töranmi§ siniflarin eyni adl1 üzvlari da xayali xüsusiyyat da§1y1rlar.

Baza X = 1

Y = 2

C = &B

C->Goster

Baza X = 2 Y = 4

//CLASSPOI.H duzeldilmis hissesi

class Baza

{ protected: int X, Y;

public: Baza(int C)

{ X = C; Y = 2 \* C; }

virtual void Goster()

{ printf("Baza X = %d\tY = %d\n", X, Y); }

};

Bu misalda töradilan sinfin daxilinda baza sinfinin üzvü olan Goster() funksiyas1 yenidan tayin olunmu§dur. Bu tayin dogrudur (B obyektinin Goster() xüsusiyyatinin olmas1na diqqat edin). Bela ki, baza sinfinin göstaricisina töradilmi§ sinif manimsadildikdan sonra göstaricidan Goster() xüsusiyyatini vermasi talab olunarsa, bu ancaq töradilmi§ sinfin obyektinin malumatlar1 ila baza sinfinin imkanlar1n1n ortaya qoyulmas1 ola bilar. Bu hal komplyatorun komplyasiya zaman1 davran1§lar1ndan qaynaqlan1r.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Bunun qar§1s1n1 almag1n an yax§1 yolu baza sinfi tayin edilarkan töradilan siniflarin dayi§diracaklari metodlar1 xayali (virtual) tayin etmakdir. Bu sababdan da bu cür metodlar1n prototip tayin etmalarina virtual açarsözü ila ba§lamaq laz1md1r.

Buna göra da laz1m olan samaraliliyi alda etmak üçün yuxar1dak1 misalda Baza sinfinin üzv funksiyas1 olan void Goster() funksiyas1 virtual sözü ila ba§layaraq virtual void Goster() kimi tayin edilmalidir. Bu cür tayin etma yerina yetirildikdan sonra töradilan siniflar üçün virtual tayin etmanin hayata keçirilmasinin va ya keçirilmamasinin heç bir ahamiyyati yoxdur. Xayali üzva

Proqram ç1x1§1 (düzali§dan sonra)

A.Goster

Baza X = 1 B.Goster

Toremis X = 2

C = &A

C->Goster

Baza X = 1

C = &B

C->Goster

Baza X = 2

Y = 2

Y = 4

Z = 6

Y = 2

Y = 4

Bu i§ prinsipini bu cür §arh etmak olar: virtual açarsözü olmadan edilan tayinlarda obyektin yaln1z malumat sahalari (dayi§kanlari) yadda§da saxlan1l1r. Metodlar üçün isa komplyator qarar verir. Lakin metod

xayali tayin edilarsa, bu, metodun yadda§dak1 yerina aid malumatda (ba§lang1c ünvan1) obyektin digar malumat sahalari ila birlikda yadda§da saxlan1l1r. Bela bir metod çag1r1ld1g1 zaman da yadda§dak1 bu yer ila alaqadar olan malumat istifada edilarak haqq1nda söhbat aç1lan obyektin metoduna müraciat etmasi tamin olunur. Xüsusila da bir sinfin metodlar1 xaricinda yoxedicilarinin da xayali olaraq tayin edilmasina imkan verilir. Layihalandiricilar isa xayali olaraq tayin edila bilmaz.

//EYRI.CPP

class Eyri

{ protected: int \_X, \_Y; public: Eyri();

Eyri(const Eyri&); Eyri(int, int);

virtual char\* Ad() const; int Yer\_X() const;

int Yer\_Y() const;

virtual void Yer\_X(int); virtual void Yer\_Y(int);

virtual int Uzunluq() const;

};

Eyri::Eyri()

{ \_X = \_Y = 0; }

Eyri::Eyri(const Eyri& eyri)

{ \_X = eyri.\_X;

\_Y = eyri.\_Y;

}

Eyri::Eyri(int x, int y)

{ \_X = x;

\_Y = y;

}

### Qaydali Funksiyalar

Obyektlarla proqramla§d1rma zaman1 asasan bütün obyektlar ümumi §akilda deyil, eyni amaliyyatlar üçün nazarda tutulanlar tasniflandirilarak baza obyektindan töradilir. Ümumi xüsusiyyatlar da mümkün olmad1qca baza obyekti daxilinda camla§arak yena bu obyekt üçün yaz1l1r. Bundan sonra asas obyektlar yaz1laraq proqram ortaya ç1x1r.

Masalan, ayrilarla alaqadar bir proqramda çevra radiusu, ellips radiusu, qövs kimi ayrilarin olmas1 va bu ayrilarin fazadak1 yerlarinin va uzunluqlar1n1n hesablanmas1 talab oluna bilar.

Bu halda bütün obyektlari bir Eyri sinfindan töratmak mümkündür. 8maliyyatlar1 yerina yetirmak üçün Eyri sinfinin ayrilarin yeri va ölçülari ila alaqadar üzv funksiyalar1 olmal1d1r.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

|  |  |  |
| --- | --- | --- |
| Eyri | | |
| Qovs | Cevre\_Radiusu | Ellips\_Radiusu |

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Lakin Uzunluq funksiyas1n1 yazmaq laz1m galdiyi zaman Eyrin-in uzunlugunun na oldugu va ya neca hesablanacag1 müayyan olmad1g1 üçün Uzunluq funksiyas1 ancaq sonradan yaz1lmaq §arti ila müvaqqati bir funksiya kimi yaz1la bilar.

char\* Eyri::Ad() const

{ return "EYRI"; }

int Eyri::Yer\_X() const

{ return \_X; }

int Eyri::Yer\_Y() const

{ return \_Y; }

void Eyri::Yer\_X(int x)

{ \_X = x; }

void Eyri::Yer\_Y(int y)

{ \_Y = y; }

EYRI.CPP davam1

int Eyri::Uzunluq() const

{ return 0; }

Bu cür tayinlarin üç asas manfi cahati vard1r:

1. Sonradan istifada edilmasina baxmayaraq tayin olunan bu funksiyalar, proqram daxilinda istifada olunmayacaqlar1na baxmayaraq proqram1n böyümasina sabab olacaqlar;
2. Proqramç1lar istifada etmadiklari bu proqram parçalar1n1 yazmal1 olacaqlar;
3. Galacakda yaz1lmas1 unuduldugu zaman sahv

naticalar veracakdir.

Bu hallardan qurtulman1n yolu isa, sinif daxilinda bu cür sonradan yaz1lmas1 talab olunan funksiyalar1 tayin etdikdan sonra “= 0“ ifadasini yazmaqd1r. Bu yaz1l1§ bela, bir funksiyan1n olmas1n1n vacibliyini, ancaq onun bu marhalada yaz1lmayacag1n1 va sonrak1 marhalalarda bu sinifdan töranan siniflarin hamin funksiyan1 tayin edarak funksiyas1nda yazacaqlar1n1 bildirir. Bu funksiyalar sonradan töranan siniflar daxilinda yaz1lacaqlar1 üçün onlar **xayali** (virtual) tayin edilmalidirlar. Bu funksiyalar1 xayali funksiyalardan farqlandirmak üçün onlara xayali **kor funksiya** (pure virtual) va ya qaydal1 funksiya ad1 verilir.

Daxilinda bu cür funksiyalar olan siniflardan obyekt yarad1la bilmaz. Bu siniflar ümumi maqsadlar üçün istifada edilir. Bu cür siniflara **mücarrad** (abstract) sinif ad1 verilir. Yuxar1dak1 misalda Eyri sinfi abstarakt sinifdir. Proqram daxilinda Eyri ad1nda bir obyekt ola bilmaz. Ancaq Eyri-dan töranmi§ haqiqi siniflarin (Xett, Cevre\_Qovsu kimi) obyektlari ola bilar.

Proqram1n strukturu ayrinin müxtalif amaliyyatlar1 üçün bir obyektda ehtiyac duyuldugu zaman mücarrad sinfin obyektlarinin yerina göstaricilari va ya taqdimatlar1

ola bilar. Tabii ki, bu halda göstarici va ya taqdimatlara

haqiqi obyektlar manimsadilmalidir.

Masalan,

main()

{ Eyri A1;

...

return 0;

### Misallar

#### Curve

Qaydal1 funksiyalar üçün verilan misal1 tamamlayaq.

}

//EYRI2.CPP

class Eyri

{ protected: int \_X, \_Y; public: Eyri();

Eyri(const Eyri&); Eyri(int, int);

virtual char\* Ad() const = 0; int Yer\_X() const;

int Yer\_Y() const;

virtual void Yer\_X(int); virtual void Yer\_Y(int);

virtual int Uzunluq() const = 0;

};

Eyri::Eyri()

{ \_X = \_Y = 0; }

Eyri::Eyri(const Eyri& eyri)

{ \_X = eyri.\_X;

\_Y = eyri.\_Y;

}

kimi bir proqram, mücarrad sinif obyektindan istifada etdiyi üçün yaz1la bilmadiyi halda

main()

{ Eyri\* Eptr = new Line(100, 100, 300, 600);

...

return 0;

}

va ya

main()

{ Eyri\*& Eptr = new Line(100, 100, 300, 600);

...

delete Eref; return 0;

}

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

proqramlar1 yaz1laraq istifada oluna bilarlar.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Eyri::Eyri(int x, int y)

{ \_X = x;

\_Y = y;

}

int Eyri::Yer\_X() const

{ return \_X; }

int Eyri::Yer\_Y() const

{ return \_Y; }

void Eyri::Yer\_X(int x)

{ \_X = x; }

void Eyri::Yer\_Y(int y)

{ \_Y = y; }

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include <math.h>

#include <stdlib.h>

class Cevre\_Qovsu : public Eyri

{ protected: int \_Radius;

int \_BaslangicBucagi; int \_SonBucagi;

public: Cevre\_Qovsu();

Cevre\_Qovsu(const Cevre\_Qovsu&); Cevre\_Qovsu(int, int, int = 1, int = 0, int = 360); Cevre\_Qovsu(int);

virtual char\* Ad() const; virtual int Uzunluq() const;

};

Cevre\_Qovsu::Cevre\_Qovsu()

{ \_Radius = 1;

\_BaslangicBucagi = 0;

\_SonBucagi = 360;

}

Cevre\_Qovsu::Cevre\_Qovsu(const Cevre\_Qovsu& qovs) : Eyri(qovs)

{ \_Radius = qovs.\_Radius;

\_BaslangicBucagi = qovs.\_BaslangicBucagi;

\_SonBucagi = qovs.\_SonBucagi;

}

Cevre\_Qovsu::Cevre\_Qovsu(int x, int y, int r, int a, int b) : Eyri(x, y)

{ \_Radius = r;

\_BaslangicBucagi = a;

\_SonBucagi = b;

}

Cevre\_Qovsu::Cevre\_Qovsu(int r)

{ \_Radius = r;

\_BaslangicBucagi = 0;

\_SonBucagi = 360;

}

char\* Cevre\_Qovsu::Ad() const

{ return "Cevre Qovsu"; }

int Cevre\_Qovsu::Uzunluq() const

{ return (int) 1.0 \* \_Radius \* 2 \* M\_PI \* abs(\_SonBucagi - \_BaslangicBucagi)/360.0;

}

#define SQR(a) ((float)(a) \* (float)(a)) class Xett : public Eyri

{ protected: int \_SonX; int \_SonY;

public:

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Xett();

Xett(const Xett&); Xett(int, int, int, int); Xett(int, int);

virtual char\* Ad() const; virtual int Uzunluq() const;

};

Xett::Xett()

{ \_SonX = 1;

\_SonY = 1;

}

Xett::Xett(const Xett& xett) : Eyri(xett)

{ \_SonX = xett.\_SonX;

\_SonY = xett.\_SonY;

}

Xett::Xett(int x1, int y1, int x2, int y2) : Eyri(x1, y1)

{ \_SonX = x2;

\_SonY = y2;

}

Xett::Xett(int x, int y)

{ \_SonX = x;

\_SonY = y;

}

char\* Xett::Ad() const

{ return "Xett"; }

int Xett::Uzunluq() const

{ return sqrt(SQR(\_SonX - Yer\_X()) + SQR(\_SonY - Yer\_Y())); }

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include <conio.h>

#include <stdio.h>

Eyri \*EyriMassivi[20]; void Hesab()

{ int i;

printf("%3s %-22s %5s %5s %5s\n%30s %5s\n"

" \n", "No","Eyri Tipi","Yer","","Olcu","X","Y");

for(i = 0; i < 20; i++) if(EyriMassivi[i])

{ printf("%2d %-22s %5d %5d %5d\n",

i, EyriMassivi[i]->Ad(),EyriMassivi[i]->Yer\_X(), EyriMassivi[i]->Yer\_Y(),EyriMassivi[i]->Uzunluq());

}

}

main()

{ clrscr(); randomize(); int i = 0;

for(i = 0; i < 20; i++) if(random(2))

EyriMassivi[i] = new Xett(random(100), random(100),

random(200), random(300));

else

{ int baslangicbucagi = random(180);

int sonbucagi = baslangicbucagi + random(180);

EyriMassivi[i] = new Cevre\_Qovsu(random(300), random(300),

random(100), baslangicbucagi, sonbucagi);

}

Hesab();

for(i = 0; i < 20; i++) if(EyriMassivi[i])

Graphics::Graphics()

{ }

void Graphics::Enter()

{ int grDrv; int grMode;

char\* grPath = ""; int grErr = grOk;

do

{ grDrv = DETECT;

grMode = 0;

initgraph(&grDrv, &grMode, grPath); if((grErr = graphresult()) != grOk)

{ printf("Error %s\n", grapherrormsg(grErr)); grPath = getBGIPath();

if(grPath == NULL) break;

}

}

while(grErr != grOk);

if(grErr != grOk) abort();

}

Graphics::~Graphics()

{ }

void Graphics::Leave()

{ MessageOnExit(); closegraph();

}

void Graphics::MessageOnExit()

{ settextjustify(BOTTOM\_TEXT, LEFT\_TEXT);

outtextxy(0, getmaxy(), "Pres Any Key to EXIT Graph Mode"); getch();

}

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

delete EyriMassivi[i];

return 0;

}

#### LineDemo

Xattin qrafik ekranda tayin olunmas1 va uc nöqtasinin harakat etdirilarak canland1r1lmas1 (animasiyas1) üçün yaz1lan bir proqrama edilan alavalar ila iki xatdan ibarat olan qrup tayini va eyni proqram1n tatbiqi ila alaqadar misallar1 gözdan keçirak:

//LINE.CPP

#include <graphics.h>

#include <stdio.h>

#include <stdlib.h>

#include <conio.h>

#include <dos.h>

class Graphics

{ public:

Graphics();

virtual ~Graphics();

virtual void Enter(); virtual void Leave();

protected:

virtual char\* getBGIPath(); virtual void MessageOnExit();

};

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

char\* Graphics::getBGIPath()

{ static char Path[67];

printf("Enter BGI path or \'.\'for end :"); scanf("%s", Path);

if(Path[0] == '.') return NULL; return Path;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* class Line

{ int X1, Y1, X2, Y2;

int Status;

public: Line();

Line(const Line&); Line(int, int); Line(int, int, int, int);

int StartX() const; int EndX() const; int StartY() const; int EndY() const;

virtual void MoveRel(int, int); virtual void Show();

virtual void Hide(); virtual int isVisible();

virtual void StartPointRel(int, int); virtual void EndPointRel(int, int);

protected:

virtual void Draw(); virtual void Clear();

};

Line::Line()

{ X1 = Y1 = Y2 = 0; X2 = 1;

Status = 0;

}

Line::Line(const Line& L)

{ X1 = L.X1; Y1 = L.Y1; X2 = L.X2; Y2 = L.Y2;

Status = L.Status;

}

Line::Line(int x, int y)

{ X1 = Y1 = 0; X2 = x, Y2 = y;

Status = 0;

}

Line::Line(int a, int b, int c, int d)

{ X1 = a; Y1 = b; X2 = c; Y2 = d;

Status = 0;

}

int Line::StartX() const

{ return X1; }

int Line::EndX() const

{ return X2; }

int Line::StartY() const

{ return Y1; }

int Line::EndY() const

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

{ return Y2; }

int Line::isVisible()

{ return Status; }

void Line::Show()

{ Status = 1; Draw();

}

void Line::Hide()

{ Status = 0; Clear();

}

void Line::MoveRel(int Dx, int Dy)

{ StartPointRel(Dx, Dy); EndPointRel(Dx, Dy);

}

void Line::StartPointRel(int dx, int dy)

{ int S = isVisible(); if(S) Hide();

X1 += dx; Y1 += dy;

if(S) Show();

}

void Line::EndPointRel(int dx, int dy)

{ int S = isVisible(); if(S) Hide();

X2 += dx; Y2 += dy;

if(S) Show();

}

void Line::Draw()

{ int C = getcolor(); setcolor(RED);

line(X1, Y1, X2, Y2);

setcolor(C);

}

void Line::Clear()

{ int C = getcolor(); setcolor(getbkcolor()); line(X1, Y1, X2, Y2);

setcolor(C);

}

void DemoStartPoint(Line& L)

{ int i;

for(i = 0; i < 23; i++)

{ L.StartPointRel(10, 0); delay(100);

}

}

class LineGroup : public Line

{ Line& L1; Line& L2;

public:

LineGroup(const LineGroup&); LineGroup(Line& a, Line& b);

void MoveRel(int, int); void Show();

void Hide(); int isVisible();

virtual void StartPointRel(int, int); virtual void EndPointRel(int, int);

private:

int StartX() const; int EndX() const; int StartY() const;

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

int EndY() const;

};

LineGroup::LineGroup(const LineGroup& LG)

: L1(LG.L1), L2(LG.L2)

{ }

LineGroup::LineGroup(Line& a, Line& b)

: L1(a), L2(b)

{ }

void LineGroup::MoveRel(int dx, int dy)

{ L1.MoveRel(dx, dy); L2.MoveRel(dx, dy);

}

void LineGroup::Show()

{ L1.Show();

L2.Show();

}

void LineGroup::Hide()

{ L1.Hide();

L2.Hide();

}

int LineGroup::isVisible()

{ return L1.isVisible(); }

void LineGroup::StartPointRel(int dx, int dy)

{ L1.StartPointRel(dx, dy); L2.StartPointRel(dx, dy);

}

void LineGroup::EndPointRel(int dx, int dy)

{ L1.EndPointRel(dx, dy); L2.EndPointRel(dx, dy);

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* void LineDemo()

{ cleardevice();

Line A(300, 300);

A.Show();

Line B(100, 200, 300, 300);

B.Show();

getch();

A.Hide();

getch();

A.Show();

getch();

DemoStartPoint(A); DemoStartPoint(B);

///////////// LineGroup //////////////

LineGroup Group(A, B); Group.Show();

getch();

Group.Hide();

getch();

Group.Show();

getch();

DemoStartPoint(Group); getch();

}

};

Graphics::Graphics()

{ }

void Graphics::Enter()

{ int grDrv; int grMode;

char\* grPath = ""; int grErr = grOk;

do

{ grDrv = DETECT;

grMode = 0;

initgraph(&grDrv, &grMode, grPath); if((grErr = graphresult()) != grOk)

{ printf("Error %s\n", grapherrormsg(grErr)); grPath = getBGIPath();

if(grPath == NULL) break;

}

}

while(grErr != grOk);

if(grErr != grOk) abort();

}

Graphics::~Graphics()

{ }

void Graphics::Leave()

{ MessageOnExit(); closegraph();

}

void Graphics::MessageOnExit()

{ settextjustify(BOTTOM\_TEXT, LEFT\_TEXT);

outtextxy(0, getmaxy(), "Pres Any Key to EXIT Graph Mode");

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

main()

{ clrscr();

Graphics graphicsMedia; graphicsMedia.Enter(); LineDemo();

graphicsMedia.Leave(); return 0;

}

indi da Line sinfi asas1nda Box (qutu) sinfini tayin edib eyni maqsadla istifada edak. Bunun üçün LINE.CPP proqram1na qal1n (bold) sitilda yaz1lm1§ satirlari a§ag1dak1 kimi alava edak:

**//BOX.CPP**

#include <graphics.h>

#include <stdio.h>

#include <stdlib.h>

#include <conio.h>

#include <dos.h>

class Graphics

{ public:

Graphics();

virtual ~Graphics();

virtual void Enter(); virtual void Leave();

protected:

virtual char\* getBGIPath(); virtual void MessageOnExit();

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

getch();

}

char\* Graphics::getBGIPath()

{ static char Path[67];

printf("Enter BGI path or \'.\'for end :"); scanf("%s", Path);

if(Path[0] == '.') return NULL; return Path;

}

class Line

{ int X1, Y1, X2, Y2;

int Status;

public: Line();

Line(const Line&); Line(int, int); Line(int, int, int, int);

int StartX() const; int EndX() const; int StartY() const; int EndY() const;

virtual void MoveRel(int, int); virtual void Show();

virtual void Hide(); virtual int isVisible();

virtual void StartPointRel(int, int); virtual void EndPointRel(int, int);

protected:

virtual void Draw(); virtual void Clear();

};

Line::Line()

{ X1 = Y1 = Y2 = 0; X2 = 1;

Status = 0;

}

Line::Line(const Line& L)

{ X1 = L.X1; Y1 = L.Y1; X2 = L.X2; Y2 = L.Y2;

Status = L.Status;

}

Line::Line(int x, int y)

{ X1 = Y1 = 0; X2 = x, Y2 = y;

Status = 0;

}

Line::Line(int a, int b, int c, int d)

{ X1 = a; Y1 = b; X2 = c; Y2 = d;

Status = 0;

}

int Line::StartX() const

{ return X1; }

int Line::EndX() const

{ return X2; }

int Line::StartY() const

{ return Y1; }

int Line::EndY() const

{ return Y2; }

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

int Line::isVisible()

{ return Status; }

void Line::Show()

{ Status = 1; Draw();

}

void Line::Hide()

{ Status = 0; Clear();

}

void Line::MoveRel(int Dx, int Dy)

{ StartPointRel(Dx, Dy); EndPointRel(Dx, Dy);

}

void Line::StartPointRel(int dx, int dy)

{ int S = isVisible(); if(S) Hide();

X1 += dx; Y1 += dy;

if(S) Show();

}

void Line::EndPointRel(int dx, int dy)

{ int S = isVisible(); if(S) Hide();

X2 += dx; Y2 += dy;

if(S) Show();

}

void Line::Draw()

{ int C = getcolor(); setcolor(RED); line(X1, Y1, X2, Y2);

setcolor(C);

}

void Line::Clear()

{ int C = getcolor(); setcolor(getbkcolor()); line(X1, Y1, X2, Y2);

setcolor(C);

}

void DemoStartPoint(Line& L)

{ int i;

for(i = 0; i < 23; i++)

{ L.StartPointRel(10, 0); delay(100);

}

}

class LineGroup : public Line

{ Line& L1; Line& L2;

public:

LineGroup(const LineGroup&); LineGroup(Line& a, Line& b);

void MoveRel(int, int); void Show();

void Hide(); int isVisible();

virtual void StartPointRel(int, int); virtual void EndPointRel(int, int);

private:

int StartX() const; int EndX() const; int StartY() const; int EndY() const;

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

};

LineGroup::LineGroup(const LineGroup& LG)

: L1(LG.L1), L2(LG.L2)

{ }

LineGroup::LineGroup(Line& a, Line& b)

: L1(a), L2(b)

{ }

void LineGroup::MoveRel(int dx, int dy)

{ L1.MoveRel(dx, dy); L2.MoveRel(dx, dy);

}

void LineGroup::Show()

{ L1.Show();

L2.Show();

}

void LineGroup::Hide()

{ L1.Hide();

L2.Hide();

}

int LineGroup::isVisible()

{ return L1.isVisible(); }

void LineGroup::StartPointRel(int dx, int dy)

{ L1.StartPointRel(dx, dy); L2.StartPointRel(dx, dy);

}

void LineGroup::EndPointRel(int dx, int dy)

{ L1.EndPointRel(dx, dy); L2.EndPointRel(dx, dy);

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* void LineDemo()

{ cleardevice();

Line A(300, 300);

A.Show();

Line B(100, 200, 300, 300);

B.Show();

getch();

A.Hide();

getch();

A.Show();

getch();

DemoStartPoint(A); DemoStartPoint(B);

///////////// LineGroup //////////////

LineGroup Group(A, B); Group.Show();

getch();

Group.Hide();

getch();

Group.Show();

getch();

DemoStartPoint(Group); getch();

}

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**class Box : public Line**

**{ public:**

**Box();**

**Box(const Box&); Box(int, int); Box(int, int, int, int);**

**protected:**

**virtual void Draw(); virtual void Clear();**

**};**

**Box::Box()**

**{ }**

**Box::Box(const Box& box) : Line(box)**

**{ }**

**Box::Box(int x, int y) : Line(x, y)**

**{ }**

**Box::Box(int a, int b, int c, int d) : Line(a, b, c, d)**

**{ }**

**void Box::Draw()**

**{ int C = getcolor(); setcolor(GREEN);**

**rectangle(StartX(), StartY(), EndX(), EndY()); setcolor(C);**

**}**

**void Box::Clear()**

**{ int C = getcolor(); setcolor(getbkcolor());**

**rectangle(StartX(), StartY(), EndX(), EndY()); setcolor(C);**

**}**

**void BoxDemo()**

**{ cleardevice();**

**Box Abox(300, 300);**

**Abox.Show();**

**Box Bbox(100, 200, 300, 300);**

**Bbox.Show();**

**getch();**

**Abox.Hide();**

**getch();**

**Abox.Show();**

**getch();**

**DemoStartPoint(Abox); DemoStartPoint(Bbox);**

**///////////// BoxGroup //////////////**

**LineGroup Group(Abox, Bbox); Group.Show();**

**getch();**

**Group.Hide();**

**getch();**

**Group.Show();**

**getch();**

**DemoStartPoint(Group); getch();**

**}**

main()

{ clrscr();

class Eyri

Graphics graphicsMedia; graphicsMedia.Enter(); LineDemo(); **BoxDemo();**

graphicsMedia.Leave(); return 0;

}

{ protected:

int \_X, \_Y;

public: Eyri();

Eyri(const Eyri&); Eyri(int, int);

vitual char\* Adlar() const = 0; int Yer\_X() const;

int Yer\_Y() const;

### C++ Metod Çagiri§ Sistemi

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

indi da xayali funksiyalar1n i§lama prinsipini §arh edak. Bunun üçün C-dan bildiyimiz kimi bir funksiya yaz1ld1g1 zaman, yadda§da komplyator tarafindan müayyan edilib çag1r1larkan onun mövqeyina avtomatik keçid ba§ verir. Lakin xayali funksiyalar üçün bu mümkün deyildir.

Eyri misal1nda oldugu kimi tayin olunmu§ baza sinfinin göstaricilarina Xett manimsadildiyi zaman çag1r1lan Uzunluq funksiyas1 xattin uzunlugunu hesablayarkan, Cevre\_Qovsu manimsadilib uzunlugu soru§uldugu zaman, bu dafa da çevra qövsü üçün hesablamalar1 apararaq geri göndarir. Bu cür siniflararas1 müqayisani aparmaq üçün siniflarin har biri üçün xayali funksiya cadvallari (virtual function table) yarad1l1r.

virtual void Yer\_X(int); virtual void Yer\_Y(int);

virtual int Uzunluq() const = 0;

};

tayinina uygun olaraq a§ag1dak1 cadval haz1rlan1r.

|  |  |  |
| --- | --- | --- |
| No | Adn | Ünvann |
| 1 | Adlar | NULL |
| 2 | Yer\_X | Eyri::Yer\_X |
| 3 | Yer\_Y | Eyri::Yer\_Y |
| 4 | Uzunluq | NULL |

Cadvaldaki NULL qiymatlari qaydal1 funksiyalar1 göstarir.

Xett sinfi tayin edilarkan ox§ar bir cadval onun üçün da haz1rlan1r. Xett sinfi Eyri sinfindan törandiyi üçün bu cadvala avvalca Eyri sinfinin qiymatlari köçürülür. Sonra isa Xett sinfi üçün yaz1lan xayali funksiyalar bu

cadvaldan tap1laraq ünvan qiymatlari dayi§dirilir. Töranan sinif üçün da yeni tayin edilan xayali funksiyalar varsa, bunlar da cadvalin sonuna satir alava edilarak daxil edilir.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

|  |  |  |
| --- | --- | --- |
| No | Adn | Ünvann |
| 1 | Ad | Xett::Ad |
| 2 | Yer\_X | Eyri::Yer\_X |
| 3 | Yer\_Y | Eyri::Yer\_Y |
| 4 | Uzunluq | Xett::Uzunluq |

Buna ox§ar amaliyyatlar töranan bütün siniflar üçün yerina yetirilir. Bu siniflardan birinin obyekti yarad1ld1g1 zaman bu siniflar virtual funksiyaya malik olduqlar1 üçün hans1 cadvalin istifada edilacayi da daxil olmaqla eyni zamanda cadval göstaricisini da saxlay1rlar. Masalan, Xett va Cevre\_Qovsu obyektlarini yaradaq:

Xett D; Cevre\_Qovsu C;

D�[VTPXett][\_X][\_Y][Son\_X][Son\_Y]

C�[VTPCevre\_Qovsu][\_X][\_Y][\_Radius][\_BaslangicBucagi][

\_SonBucagi]

VTP  VirtualTablePointer

Göründüyü kimi har bir tayina xayali funksiyan1n cadval göstaricisi (VTP) daxildir. Bu malumat har bir obyektin aid oldugu sinfin funksiya cadvalini göstarir.

Xayali funksiyalar çag1r1larkan da ela bu cadvaldan faydalanaraq çag1rma amaliyyatlar1 yerina yetirilir.

Masalan, D obyekti üçün Uzunluq funksiyas1 çag1r1ld1g1 zaman, D-nin VTP qiymati ila Xett sinfinin cadvalina va bu cadvalin 4-cü satri vasitasila da Xett::Uzunluq funksiyas1na müraciat edilacakdir. Bu cür birba§a olmayan müraciatlar C proqramlar1na nisbatan süratin a§ag1 dü§masina sabab olur.

### Mövcud Olandan Töranan Siniflar

Malum oldugu kimi obyektyönlü proqramla§d1rmada töradilmi§ sinfin yaln1z bir baza sinfina malik olma macburiyyati yoxdur. Bir sinif bir neça sinifdan töraya bilar. Bu halda töranan sinif törandiyi sinfin bütün üzvlarina müraciat edib onlar1 istifada eda bilar. Bu hal bütün siniflarin xüsusiyyatlarinin tak bir sinifda camlanmasini tamin etdiyi üçün daha güclü siniflarin yaranmas1na sabab olur. Bu cür tayinlar çox bazal1l1q adland1r1l1r.

C++-da bu cür tayinlarda baza siniflarinin virtual, olub olmamas1ndan as1l1 olmayaraq eyni metodlardan (eyni ad va parametr siyah1s1na malik olan funksiyalardan) ibarat olmas1 zaman1 töranan sinif daxilinda bu metodlar1n yenidan tayin edilmasina ehtiyac

virtual void Yaz()

{ printf("Bu B sinfinin Yazilmasidir.\n"); }

virtual void Goster()

{ printf("Bu B sinfinin Gosterilmesidir.\n"); }

};

class C : public A, public B

{ public: C()

{ }

void C\_Xususi()

{ printf("Bu C sinfinin C\_Xususiyyetidir.\n"); }

void Yaz()

{ printf("Bu C sinfinin Yazilmasidir.\n"); }

void Goster()

{ printf("Bu C sinfinin Gosterilmesidir.\n"); }

};

main()

{ clrscr(); C obyekt;

obyekt.A\_Xususi(); obyekt.B\_Xususi(); obyekt.C\_Xususi();

obyekt.Yaz(); obyekt.Goster();

printf("\n"); A \*APtr;

APtr = &obyekt;

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

vard1r. Bu tayin ila baza siniflarindan biri va ya ham1s1 çag1r1la bilacayi kimi, metod yenidan da yaz1la bilar.

Çox bazal1l1q xüsusiyyatinin üstünlüklarindan biri

da töranan sinfin obyektinin baza siniflarindan har hans1 birinin göstaricisina manimsadila bilmasidir. Bu halda töranan sinfin obyekti manimsadildiyi sinif kimi davranaraq özüna aid qabiliyyatlarini nümayi§ etdirir.

//MULTINT2.CPP

#include <conio.h>

#include <stdio.h>

class A

{ public: A()

{ }

void A\_Xususi()

{ printf("Bu A sinfinin A\_Xususiyyetidir.\n"); }

void Yaz()

{ printf("Bu A sinfinin Yazilmasidir.\n"); }

virtual void Goster()

{ printf("Bu A sinfinin Gosterilmesidir.\n"); }

};

class B

{ public: B()

{ }

void B\_Xususi()

{ printf("Bu B sinfinin B\_Xususiyyetidir.\n"); }

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

APtr->A\_Xususi(); APtr->Yaz();

APtr->Goster(); printf("\n");

B \*BPtr;

BPtr = &obyekt; BPtr->B\_Xususi(); BPtr->Yaz();

BPtr->Goster();

return 0;

}

Proqram ç1x1§1

Bu A sinfinin A\_Xususiyyetidir. Bu B sinfinin B\_Xususiyyetidir. Bu C sinfinin C\_Xususiyyetidir. Bu C sinfinin Yazilmasidir.

Bu C sinfinin Gosterilmesidir.

Bu A sinfinin A\_Xususiyyetidir. Bu A sinfinin Yazilmasidir.

Bu C sinfinin Gosterilmesidir.

Bu B sinfinin B\_Xususiyyetidir. Bu C sinfinin Yazilmasidir.

Bu C sinfinin Gosterilmesidir.



#define IMP\_SORT(Tip)

void imp\_sort\_##Tip(Tip Mas[], unsigned int Olcu)

{

int Nezaret = 1; int i;

Tip c;

while (Nezaret)

{

Nezaret = 0;

for( i = 1; i < Olcu; i++ ) if(Mas[i-1]>Mas[i])

{ c = Mas[i-1];

Mas[i-1] = Mas[i]; Mas[i] = c; Nezaret = 1;

}

}

}

IMP\_SORT(float); IMP\_SORT(double); IMP\_SORT(int);

IMP\_SORT(long);

#define SORT(Tip, Mas, Olcu) imp\_sort\_##Tip(Mas, Olcu)

#include <conio.h>

#include <stdio.h>

double D[5] = {3.4, 7, -2.5, 4.1, 0.5};

long L[5] = {6, 2, 90, 34, 45};

int i;

main()

{ clrscr();

SORT(double, D, 5);

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

# VI F8SiL

## $ABLONLAR HAZIRLAMAQ

### $ablonlar

Bir proqram kodunun müxtalif hallar üçün takrar yaz1lmas1 yerina qalib kimi haz1rlan1b istifada edilmasina

**§ablonlama** (template) ad1 verilir. Masalan, float tipli elementlardan ibarat massiv ila double tipli va hatta int, char va ya long tipli elementlardan ibarat massivlarin kiçikdan böyüya s1ralanmas1 alqoritmas1 aras1nda heç bir farq yoxdur. Bu halda har bir tip üçün ayr1 bir s1ralama

funksiyas1n1n yaz1lmas1 art1q bir i§dir. Yaln1z bir proqram kodu ila proqram yazmaq ham proqramç1n1n yükünü azald1r, ham da galacakda edilacak dayi§ikliklarda eyni maqsad üçün bir çox proqram kodunun dayi§dirilmasi macburiyyatini aradan qald1r1r.

Bu bax1mdan C dilinda bu cür kodla§d1rmalar makrolar1n kömayi ila apar1l1r va bu hal **ümumi** (generic) tayinlar adland1r1l1r. Bir misala baxaq:

//GENSORT.C

funksilar1n xaricinda (qlobal olaraq)

SORT(long, L, 5);

printf("\nlong\t\tdouble\n"); for(i = 0; i < 5; i++)

printf("%ld\t\t%lf\n", L[i], D[i]);

return 0;

}

IMP\_SORT(char); satri yaz1lmal1d1r.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Misalda IMP\_SORT makrosu ila s1ralama funksiyalar1n1n ümumi bir §ablonu ç1xard1lm1§, tayin olunma satrinin alt1nda isa yeni bir makrodan istifada edarak mövcud tiplar üçün s1ralama funksiyalar1 alda edilmi§dir. Bu funksiyalar1n çag1r1lmas1 üçün da SORT adl1 digar bir makro yaz1lm1§d1r.

Ümumila§dirilmi§ bu makro tayinlarinin manfi

cahatlari isa a§ag1dak1lard1r:

* S1ralanacaq bir massivin elementlarinin tipi parametr siyah1s1na birba§a yaz1lmal1d1r. 8ks halda sahv ba§ verir;
* 8gar massivin elementlarinin tipi dayi§dirilacaksa, bu massivi s1ralamaq üçün istifada edilan SORT funksiyalar1 da dayi§dirilmalidir;
* S1ralanacaq har massivin elementlari IMP\_SORT ila uygun tipda tayin edilmalidir. Masalan, char tipli bir massiv s1ralanmadan avval bütün

### $ablon Funksiyalar

indi da C++-da §ablon xüsusiyyata malik eyni proqram1 yazaq.

//TMPSORT.CPP

template <class Tip>

void SORT(Tip Mas[], unsigned int Olcu)

{

int Nezaret = 1; int i;

Tip c;

while(Nezaret)

{

Nezaret = 0;

for( i = 1; i < Olcu; i++ ) if(Mas[i-1]>Mas[i])

{

c = Mas[i-1]; Mas[i-1] = Mas[i]; Mas[i] = c; Nezaret = 1;

}

}

}

#include <conio.h>

#include <stdio.h>

main()

template ifadasi C++1n acar sözüdür. template ila tayin edilan funksiyalarda avvalcadan proqram kodunun dayi§masina sabab olan tiplar tayin edilarak onlara simvolik bir ad verilir. Bu tayin template açar sözü ila barabar a§ag1dak1 simvolik adlar alava edilarak ba§lan1r.

{

clrscr();

double D[5] = {3.4, 7, -2.5, 4.1, 0.5};

long L[5] = {6, 2, 90, 34, 45};

SORT(D, 5u);

SORT(L, 5u);

int i; printf("\nlong\t\tdouble\n"); for(i = 0; i < 5; i++)

printf("%ld\t\t%lf\n", L[i], D[i]);

return 0;

}

template<class simvolik\_ad1 [, class simvolik\_ad2 ...]>

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Proqram ç1x1§1

|  |  |
| --- | --- |
| long | double |
| 2 | -2.500000 |
| 6 | 0.500000 |
| 34 | 3.400000 |
| 45 | 4.100000 |
| 90 | 7.000000 |

Bununla mövcud olan bütün tiplar üçün SORT

funksiyas1 tayin edilir.

Burada SORT funksiyas1n1n çag1r1lmas1na nazar yetirsaniz, s1ralanacaq massivin elementlarinin tiplarina aid heç bir malumat görmayacaksiniz. Çünki art1q komplyator SORT funksiyas1n1n çag1r1lmas1 zaman1 birinci parametrin tipini istifada edarak hans1 funksiyan1 çag1racag1na özü qarar verir.

Bu tayindan sonra funksiya, simvolik adlardan da istifada edilarak C++ qaydalar1na göra yaz1l1r.

template ila alda edilan funksiyalarla istifadaçinin özünün tayin etdiyi obyektlarin istifada edilmasi da mümkündür.

//TMPMAX.CPP

#include <conio.h>

#include <iostream.h>

template <class T> T Max(T a, T b)

{ return a > b ? a: b; }

class A

{ int N, M;

public: A()

{ N = 0; M = 1; }

A(int a, int b)

{ N = a; M = b; }

A(const A& K)

{ N = K.N; M = K.M; }

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

sinifi tayin edilarkan operator > funksiyas1 da tayin edilmi§dir.

friend int operator>(const A& a, const A& b)

{ return a.N % a.M > b.N % b.M; }

friend ostream& operator<<(ostream& Stream, const A& a)

{ Stream<<a.N % a.M; Stream<<"("<<a.N<<" mod "<<a.M<<")";

return Stream;

}

};

main()

{ clrscr();

A Bir(34, 9);

A Iki(456, 13);

cout<<"Bir = "<<Bir<<endl; cout<<"Iki = "<<Iki<<endl;

cout<<"En boyuk qiymet ->"<<Max(Bir, Iki)<<endl;

return 0;

}

### $ablon Obyektlar

Bazi obyektlar icra olundugu zaman istifada etdiklari dayi§kanlarin tiplari müxtalif hallarda dayi§a bilar. Bu obyektlar üçün da funksiyalardak1na ox§ar

§ablonlar haz1rlay1b istifada etmak olar. Obyekt tayinlari iki va daha çox hissadan (üzv va dost funksiya tayinlarindan) ta§kil olundugu üçün funksiya tayinindan nisbatan farqlanir.

Proqram ç1x1§1

Bir = 7(34 mod 9)

Iki = 1(456 mod 13)

En boyuk qiymet ->7(34 mod 9)

//TMPARRAY.CPP

#include <conio.h>

#include <iostream.h>

#include <stdlib.h>

template <class Tip> class Array

{ Tip \*Mas; unsigned int Olcu;

public: Array(unsigned int); Array(const Array&);

~Array()

{ if (Mas) delete Mas; }

Tip& operator[](unsigned int);

virtual void PrintHeader(ostream& Stream) const

{ Stream<<'{'; }

Burada template ila tayin edilan funksiyan1n simvolik adla verilan obyektdan na gözladiyina diqqat etmak laz1md1r. Max funksiyas1 obyektin öz sinfindan digar obyekt ila müqayisa edilmasini va “böyükdürmü?” sual1n1n cavabland1r1lmas1n1 gözlayir. Buna göra da A

Burda sinif tayini funksiyalarda oldugu kimi template ifadasi ila ba§lam1§d1r. Simvolik tip ad1na asaslanaraq tayinlar edilmi§dir. Buradak1 istifada qaydas1 da funksiyalarda oldugu kimidir. Lakin inline ila kodla§d1r1lmayan funksiyalar üçün sonradan kodla§d1r1larkan template ifadasini sinif tayininda oldugu kimi kodla§d1r1lacaq funksiyalar1n avvalina yazmaq laz1md1r. Burada diqqat edilacak hal simvolik tip adlar1n1n eyni olmas1 deyil, sinif tayinindaki kimi tip ad1n1n bu tayin daxilinda da eyni ard1c1ll1qla yaz1lmas1d1r.

virtual void PrintSeperator(ostream& Stream) const

{ Stream<<','; }

virtual void PrintTrailer(ostream& Stream) const

{ Stream<<'}'; }

virtual void Print(ostream&) const;

};

TMPARRAY.CPP davam1

Burada nazari calb edan, template ifadasindan ba§qa, tayin edilan üzv funksiyas1n1n hans1 obyekta aid oldugunu göstaran görma (scope) operatorunun yaz1lmas1ndak1 dayi§iklikdir. Normal halda Array::

§aklinda olmas1 laz1m galan tayin Array::<Tip>:: §aklinda edilmi§dir. Buna sabab Array-in sinif tayini deyil, müxtalif Array sinfinin §ablonu olmas1d1r. Ba§qa sözla tayin edilmakda olan üzv funksiya Array<Tip> kimi tayin olunmu§ bir §ablonun üzvüdür.

Digar tarafdan tayin edilmakda olan layihalandiricinin ad1n1n sadaca Array olduguna diqqat edin.

TMPARRAY.CPP davam1

template <class Tip> Array<Tip>::Array(const Array<int> &A)

{ Mas = new Tip[Olcu = A.Olcu]; if(!Mas) abort();

int i;

for(i = 0; i < Olcu; i++) Mas[i] = A.Mas[i];

}

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Tayin edilmakda olan üzv funksiyan1n parametrlari köçürma layihalandiricisinda oldugu kimi §ablondan alda edilacak bir obyekti göstaracaksa, bu Array kimi deyil, Array<Tip> §aklinda yaz1lmal1d1r. Bu, §ablondan alda edilacak sinfin üzv va ya dost funksiyas1n1n daxilina

template <class Tip> Array<Tip>::Array(unsigned int B)

{ Mas = new Tip[Olcu = B]; if(!Mas) abort();

}

eyni sinifdan bir obyektin parametr kimi girmasi demakdir.

TMPARRAY.CPP davam1

template <class Tip>

Tip& Array<Tip>::operator[](unsigned int I)

{ static Tip Komekci;

if(I < Olcu) return Mas[I]; return Komekci;

}

template <class Tip>

void Array<Tip>::Print(ostream& Stream) const

{ PrintHeader(Stream);

int i;

for(i = 0; i < Olcu - 1; i++)

{ Stream<<Mas[i]; PrintSeperator(Stream);

}

Stream<<Mas[Olcu - 1]; PrintTrailer(Stream);

}

Array<unsigned long int> MassivULI;

§aklinda olmal1d1r.

TMPARRAY.CPP davam1

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

main()

{ clrscr(); Array<int>I(5);

I[0] = 1; I[1] = 3; I[2] = 5; I[3] = 7; I[4] = 9;

cout<<endl; I.Print(cout); cout<<endl;

return 0;

}

�ablon tayin edildikdan sonra onun vasitasila obyektlar tayin edilarkan art1q simvolik tip adlar1n1n avazina haqiqi adlar tayin etma ila barabar yaz1lmal1d1r.

Masalan, double va unsigned long int elementli massivlar üçün

Array<double> MassivD;

### Axin Nadir?

# VII F8SiL

## AXINLAR

|  |  |  |  |
| --- | --- | --- | --- |
| **Ax1n** | **Simvolik fayl** | **DOS fayl1** | **istifada maqsadi** |
| cerr | stderr | con | sahv masajn |
| clog |  | con | çap |

Har hans1 bir ax1ndan verilanlari oxumaq üçün >> (saga sürü§dürma), ax1na verilanlari yazmaq üçün << (sola sürü§dürma) operatorundan istifada edilir.

indi bucag1n qiymatini daraca ila daxil edan va radyan ila ekrana ç1xaran bir C va C++ proqram1 yazaraq

**Axin** (Stream) – malumatlar1n ard1c1l formada ax1n1n1 tamin edan, malumatlar1 istifada etmazdan avval va sonra onlar1n saxlanmas1n1 ard1c1l formada nizamlayan mexanizmdir (sinif va ya obyekt). Ax1nlar giri§ va ya ç1x1§ maqsadila istifada edilir. Ax1nlar1n asas ahamiyyati istar standart tipda olsun, istarsa da proqramç1 tarafindan tayin edilmi§ olsun, har tipda olan dayi§kanin (obyektin) ax1na yaz1l1b oxuna bilmasidir. C- da yaz1lan proqramlarda oldugu kimi yazma va oxuma amaliyyatlar1nda format satrina ehtiyac yoxdur.

### Standart Axinlar

C++-da tayin olunmu§ standart ax1nlar

a§ag1dak1lard1r:

bunlar1 müqayisa edak.

//BUCAQ.C

#include <stdio.h>

#include <math.h>

#include <conio.h> double d, r;

main()

{ clrscr();

printf("Bucaq (derece) : ");

scanf("%lg", &d);

r = d / 180.0 \* M\_PI; printf("\n%lg%c = %lg rad\n",

d, 248, r);

return 0;

}

//BUCAQ.CPP

#include <iostream.h>

#include <math.h>

#include <conio.h> double d, r;

main()

{ clrscr();

cout <<"Bucaq (derece) : "; cin >>d;

r = d / 180.0 \* M\_PI;

cout <<d<<(char)248<<" = "

<<r<<" rad"<<endl;

return 0;

}

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Bu iki proqram1n müqayisasindan göründüyü kimi

|  |  |  |  |
| --- | --- | --- | --- |
| **Ax1n** | **Simvolik fayl** | **DOS fayl1** | **istifada maqsadi** |
| cin | stdin | con | standart giri$ |
| cout | stdout | con | standart çnxn$ |

printf(“mesaj”); ifadasi cout<<”mesaj”; ifadasina çevrilmi§,

cout printf ila eyni funksiyan1 yerina yetirmi§dir. printf("%d, %lf\n", 5, 1.4); kimi bir ifada da cout<<5<<”,”<<1.4<<”\n”; §aklina çevrilarak format tayinedicilari göstarilmami§dir. Belalikla, format tayinedicisinin istadiyi tip ila uygun galan qiymatin tiplarinin uygunsuzlugundan meydana gala bilacak sahvlar aradan qald1r1lm1§d1r. Bununla barabar növbati paraqraflarda göracayimiz kimi formatl1 ç1x1§ üçün C-nin imkanlar1ndan da istifada oluna bilar.

Giri§ (oxuma) amrlarinda isa scanf(“%lg”, &d); kimi bir ifadanin yerina cin>>d; ifadasi yaz1laraq format ifadasi va & ünvan operatorundan istifada edilmami§dir.

Ax1nlar1n istifada edilmasi zaman1 ax1na iki va daha art1q arqumentin daxil edilmasi, ya da oxunmas1 laz1m galarsa, arqumentlar aras1nda uygun istiqamatlandirma i§aralari qoyaraq amaliyyatlar1 yerina yetirmak mümkündür.

int X, Y; double Z; unsigned int U;

cin >>X>>Z>>U>>Y;

cout <<X<<’\*’<<Y;

cout <<’=’<<(X\*Y)<<’\n’;

cin >> U; cin >> Y;

cout <<X<<’\*’<<Y<<’=’<<(X\*Y)<<’\n’;

eyni amaliyyatlar1 yerina yetiran iki proqram hissasidir.

<< va >> operatorlar1n1 ax1nlarla istifada edarkan ard1c1ll1qlar1na diqqat etmak laz1md1r. Bela ki,

cout <<X = Y<<’\n’;

ifadasinda amaliyyatlar1n prioritet s1ras1 nazara al1narsa,

(cout <<X) = (Y<<’\n’);

§aklinda icra olunur. Bu da complyasiya sahvi verir. Çünki (cout<<X) amaliyyat1 naticasinda X-in qiymati ekrana ç1xar1laraq cout qiymati al1n1r. ikinci halda ifada cout = (Y<<’\n’); hal1n1 al1r. Bu ifadanin sag tarafi da hesabland1qdan sonra (agar, tabii ki, hesablana bilarsa) cout-a bir qiymat manimsadilmaya cahd edilir. Bu cür manimsatma operatoru tayin olunmad1g1 üçün sahv ba§ verir. Bela hallarda ifada mötarizalarin kömayi ila aç1k

§akilda yaz1lmal1d1r.

cout <<(X = Y)<<’\n’;

va ya

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

cin >>X; cin >> Z;

Yena da saga va ya sola sürü§dürma amaliyyatlar1n1n ax1nlarla birlikda istifada edilmasi

sahvlara yol aça bilacayi üçün mötarizalardan istifada etmak maqsadauygundur.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

int X = 3;

cout <<X<<1<<’\n’; cout <<(X<<1)<<’\n’;

Proqram naticasinin ekran görünü§ü

31

6

olacaqd1r. ilk satir 3 ila 1-in ard1c1l yaz1lmas1n1, ikinci satir isa 3-ün bir bit sola sürü§dürülmasi va naticanin ekrana ç1xar1lmas1n1 tamin edir.

### Axinlara Nizamlanmi§ Malumat Yazilmasi

#### Geni§lik Nazarati

Ekrana ç1xar1lacaq malumat1n ax1n üzarinda müayyan sayda simvol uzunlugunu doldurmas1 talab olunarsa, bu ax1n üçün width() üzv funksiyas1ndan istifada edilir. Masalan, ekrana ç1xar1lacaq istanilan bir adadin 12 simvol uzunlugunda olmas1 üçün

cout.width(12); cout<<56;

ifadasindan istifada edilir. width ifadasi sadaca özündan sonrak1 malumat sahasinin yaz1lmas1na tasir edir. Daha sonrak1 malumatlar1n ekrana ç1xar1lmas1nda manas1 olmaz. 8gar laz1m galarsa, digar malumatlar1 da ekrana

<1xarmazdan avval eyni ifadadan istifada edilmalidir.

Lakin malumat verilan geni§liya s1xmazsa, bu geni§lik nazara al1nm1r va ehtiyac oldugu qadar saha istifada edilarak ekrana ç1xar1l1r.

8vvalcadan verilmi§ geni§lik qiymatini müayyanla§dirmak üçün parametrsiz width() funksiyas1 istifada edilarak bundan sorak1 yazma (ekrana ç1xarma) amaliyyat1nda malumat üçün ayr1lacaq sahanin geni§liyini tayin etmak olar.

int Genislik = cout.width();

width() funksiyalar1n1n hans1n1n istifada edilmasindan as1l1 olmayaraq naticada ax1n üçün daha avvalki add1mlarda tayin olunan yazma geni§liyinin qiymati geri göndarilacakdir. Har yazma amaliyyat1ndan sonra yazma geni§liyi s1f1r olacaqd1r. Yazma geni§liyinin s1f1r olmas1 ax1na ehtiyac oldugu qadar yazma sahasini istifada etma imkan1n1 verir.

int width();

int width(int);

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

#### Yerla§ma Nazarati

Malumat yaz1larkan, onun geni§liyi verilan geni§likdan az olarsa, malumatlar saga s1x1§d1r1laraq yaz1l1r. Sol taraflari isa bo§ burax1l1r.

8gar malumatlar1 yaln1z saga s1x1§d1r1lm1§ deyil, istaya uygun olaraq saga va ya sola s1x1§d1r1laraq yazmaq talab olunarsa, ax1nlar1n mansub olduqlar1 ios::adjustfield parametri ios::left va ya ios::right kimi göstarilmalidir. Bu bax1mdan ios::adjustfield parametrina manimsatmak üçün setf() funksiyas1ndan istifada etmak olar. Masalan,

cout.setf(ios::left, ios::adjustfield); cout.widht(14);

cout<<56;

cout.setf(ios::right, ios::adjustfield); cout.width(14);

cout<<56;

Burada ios::right malumat1n verilan geni§lik daxilinda saga, ios::left isa sola s1x1§d1r1lmas1n1 tamin edir.

#### Bo§luq Nazarati

Malumatlar1n müayyan bir geni§likdaki sahada saga va ya sola s1x1§d1r1laraq yaz1lmas1 naticasinda sol va ya sag tarafda istifada olunmam1§ sahalar qal1r. Bu sahalar çox zaman bo§ saxlansa da, bazan doldurula da bilar. Bunun üçün fill() üzv funksiyas1ndan istifada edilir. Masalan, yaz1lacaq qiymatlarin saga s1x1§d1r1lm1§ olmas1 va solda qalan bo§ sahalarin da s1f1rla doldurulmas1 talab olunarsa,

cout.setf(ios::right, ios::adjustfield); cout.fill(‘0’);

cout.width(14); cout<<78<<”\n”;

kimi bir proqram hissasi yaz1la bilar.

fill() funksiyas1n1n tasiri yeni bir fill() funksiyas1n1n istifada edilmasina kimidir. Buna göra da doldurulma macburiyyatinin aradan qald1r1lmas1 üçün fill(32); va ya fill(‘ ‘); satirlari yaz1lmal1d1r.

fill() funksiyas1 har zaman avvalki hal1nda istifada edilmayan sahalara doldurulacaq simvolu qaytarir. 8gar doldurma simvolu dayi§dirilmazsa va sadaca hans1 simvolun oldugunu müayyanla§dirmak laz1m galarsa, parametrsiz fill() funksiyas1ndan istifada etmak olar.

char EvvelkiSimvol = cout.fill();

cout.fill(‘!’);

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

...

cout.fill(EvvelkiSimvol);

Har iki fill() funksiyas1n1n prototipi a§ag1dak1 kimidir:

char fill(); char fill(char);

#### Tam 8dadlarin 8saslarina Nazarat

Tam adadlar 3 formada: **onluq** (decimal), **sakkizlik** (octal) va **onaltifiq** (hexadecimal) say sistemlarinda yaz1la bilarlar. Tam adadlar ax1nlara bu say sistemlarindan birinda yaz1la bilar. 8sasan onluq say sisteminda, laz1m galdiyi zaman da digar say sistemlarinda yaz1la bilar. Bunun üçün setf() üzv funksiyas1ndan istifada edilir. setf() funksiyas1 ila ax1nlar üçün tayin olunmu§ ios::basefield parametrina ios::dec, ios::oct va ya ios::hex qiymatlarindan biri alava edilmalidir.

int x = 36;

cout.setf(ios::dec, ios::basefield); cout<<x<<”\n”;

cout.setf(ios::oct, ios::basefield); cout<<x<<”\n”;

cout.setf(ios::hex, ios::basefield); cout<<x<<”\n”;

naticada

36

44

24

qiymatlari ax1nda (cout oldugu üçün ekranda) görünacakdir.

#### Haqiqi 8dadara Nazarat

Haqiqi adadlar yaz1larkan onluq nöqtadan sonra neça raqamin yaz1lacag1n1n tayin olunmas1, laz1ms1z raqamalarin qar1§1ql1q va nizams1z bir görünü§ yaratmas1n1n qar§1s1n1 al1r. Bu precision() üzv funksiyas1 ila tayin edilir. Bu funksiya ila nöqtadan sonra görünacak raqamlarin say1 tayin edilir. Geri qaytarma qiymati kimi isa avvalki asas qiymat qaytar1l1r.

double X = 7.7881881; cout.precision(3); cout<<X<<”\n”

naticada

7.788

va ya

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

double X = 7.7881881; cout.precision(2); cout<<X<<”\n”

naticada

7.79

kimi görünacakdir.

Buradan göründüyü kimi at1lan raqamlar yuvarlaqla§d1r1l1r.

Haqiqi adadlarin yaz1lmas1nda, asas qiymat xaricinda ahamiyyatli olan qiymatin nizamlanm1§ olmas1d1r. Haqiqi adadlar iki müxtalif formada göstarila bilar: ±x.xxxxE±xxxx §aklinda mühandislik görünü§ü (scientific) va ±xxxx.xxxx §aklindaki normal görünü§ (fixed).

Ax1nlar üçün nizamlama amaliyyatlar1n1n yerina yetirilmasi ancaq onlar üçün tayin edilmi§ ios::floatfield parametrina ios::fixed va ya ios::scientific qiymatlarindan birinin yaz1lmas1 ila mümkündür. ios::fixed normal görünü§, ios::scientific isa mühandislik görünü§ü üçündür.

double X = 567.8990; cout.setf(ios::fixed, ios::floatfield); cout<<X<<”\n”; cout.setf(ios::scientific, ios::floatfield);

cout<<X<<”\n”

Naticada ax1ndak1 görünü§

567.899

5.67899e+2

§akilnda olur.

### Axinlardan Nizamlanmi§ Malumat Oxunmasi

Ax1nlardan malumatlar1n oxunmas1 üçün yena da avvalki paraqrafda §arh edilan nizamlanm1§ malumat yazma funksiyalar1ndan istifada eda bilarsiniz. Bunlardan yaln1z tam adadlarin asaslar1n1n göstarilmasi ahamiyyatlidir. Digarlarinin heç bir tasiri yoxdur.

Tam adadi ax1ndan oxumaq üçün bu adadin avvalinda 0o, 0O, 0x, 0X kimi i§aralar yoxdursa, daxil edilmi§ adadlar onluq adad kimi qabul edilir. 8dadlarin avvalinda 0o va ya 0O yaz1larsa, onlar1n sakkizlik, 0x va ya 0X yaz1larsa, onlar1n onalt1l1q say siteminda yaz1ld1g1 qabul edilir. Bu halda oxunacaq har hans1 bir adadin asas1 göstarilarsa, ax1nda olan adad da eyni asasda olmal1d1r. 8ks halda sahv qabul edilir.

Misal 1:

Proqram giri§ va ç1x1§1

45

37

Misal 3:

#include <conio.h>

#include <iostream.h>

main()

{ clrscr();

int Integer;

cin>>Integer; cout<<”\n”<<Integer<<”\n”;

return 0;

}

Proqram giri§ va ç1x1§1

45

45

#include <conio.h>

#include <iostream.h>

main()

{ clrscr();

int Integer;

cin.setf(ios::hex, ios::adjustfield); cin>>Integer; cout<<”\n”<<Integer<<”\n”;

return 0;

}

Misal 2:

Proqram giri§ va ç1x1§1 1

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

45

69

Proqram giri§ va ç1x1§1 2

0x45

69

#include <conio.h>

#include <iostream.h>

main()

{ clrscr();

int Integer;

cin.setf(ios::oct, ios::basefield); cin>>Integer; cout<<”\n”<<Integer<<”\n”;

return 0;

}

Proqram giri§ va ç1x1§1 3

0o45 <sahfdir 0

### Sahvlara Nazarat

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Har hans1 bir ax1n ila yerina yetirilan amaliyyatlarda sahvin ba§ vermasi zaman1 ax1n özü-özlüyünda bir i§ görmür. Yaln1z sahvin ba§ verdiyi haqq1nda malumat1 özünda saxlay1r. Daha sonra özündan soru§ulan bazi malumatlara bu malumata asaslanaraq cavab verir.

Bu suallar a§ag1dak1lard1r:

|  |  |
| --- | --- |
| Funksiya | (TRUE) Snfnrdan farqli qaytarma qiymatinin sababi |
| int bad() | Sahv ba$ vermi$sa. |
| int fail() | 8maliyyat sahv sababindan yarnmçnq qalmn$sa. |
| int good() | icra olunan amaliyyatlardan sonra heç bir sahv  olmamn$sa. |
| int eof() | Oxuma maqsadli axnndan fayl sonu simvolu oxunmu$sa. |

Sahvin sababi malum olub düzaldildikdan sonra amaliyyatlar1n davam etdirilmasi talab olunarsa, meydana galan sahvin sonrak1 marhalalarda da sahv kimi qiymatlandirilmamasi üçün ax1na aid sahv malumat silinmalidir. Bunun üçün void clear(); funksiyas1ndan istifada olunur.

### Fayl Üzarindaki Axinlar

Fayllar1n ax1n kimi istifada edilmasi üçün bazi funksiyalar tayin edilmi§dir. Bu funksiyalar1 istifada

etmak üçün fstream.h ba§l1q fayl1 proqram koduna alava

edilmalidir.

#### Fayla Yazma

Malumatlar1n fayllara yaz1lmas1 üçün ofstream() sinfindan istifada edilir. Bu sinfin 4 müxtalif layihalandiricisi vard1r.

ofstream();

Bu layihalandirici yaln1z yazma maqsadli ax1n tayin edir. Bu ax1n1n hans1 fayl oldugu open() üzv funksiyas1 ila müayyan edilir. open() funksiyas1n1n strukturu

void open(char\* *fayl\_adt*, int *tip*);

§aklindadir. *fayl\_adt* aç1lacaq fayl1n amaliyyat sisteminin xüsusiyyatlarina uygun olaraq verilmi§ ad1d1r. *tip* isa yazma maqsadli fayllar üçün ios::out va ya ios::app olmal1d1r. ios:app agar fayl mövcuddursa, yeni malumatlar1n fayl1n sonuna alava edilmasini, ios::out isa malumatlar1n fayl1n ba§lang1c1ndan etibaran daxil edilmasini tamin edir. Bu halda fayldak1 mövcud olan malumatlar silinir.

Ax1nlar aç1lma rejimindan as1l1 olmayaraq close()

üzv funksiyas1 ila baglanmal1d1r.

if (Output.bad())

{ cerr<<"Fayl acilmir.\n"; return 1;

}

Output<<"Axina yazma\n"; Output<<8<<' '<<7.78<<"\n";

Output.close(); return 0;

}

MISAL2.DAT fayl1na yaz1lan malumatlar

Axina yazma 8 7.78

#include <fstream.h>

main()

{ ofstream Output;

Output.open("misal1.dat", ios::out); if (Output.bad())

{ cerr<<"Fayl acilmir.\n"; return 1;

}

Output<<"Axina yazma\n"; Output<<8<<' '<<7.78<<"\n";

Output.close(); return 0;

}

MISAL1.DAT fayl1na yaz1lan malumatlar

Axina yazma 8 7.78

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

ofstream(char\* *fayl\_adt*);

Bu layihalandirici da *fayl\_adt* ila verilan fayl1n ax1nla alaqalandirilarak istifada edilmasini tamin edir.

#include <fstream.h>

main()

{ ofstream Output("misal2.dat");

ofstream(int *handle*);

Bu layihalandirici daha avval aç1lm1§ bir fayl1n ax1n kimi aç1l1b istifada edilmasi üçün tayin edilmi§ bir layihalandiricidir. Bu layihalandirici ila barabar istifada edilan *handle* ila verilan fayl bu tayindan sonra sadaca ax1n kimi istifada edilmalidir. 8ks halda fayl daxilinda sahv malumatlara rast gala bilarsiniz.

ofstream(int *handle*, char\* *buffer*, int *uzunluq*);

Bundan avval §arh edilmi§ layihalandirici kimi eyni maqsadlar üçün istifada edilan bu layihalandirici *buffer* ila verilan *uzunluq* uzunlugundak1 aral1q yadda§ vasitasi ila fayla yazma amaliyyat1n1 yerina yetirir. Yani aral1q yadda§ dolana qadar malumatlar1 bu yadda§a yaz1r. Yadda§ dolduqdan sonra bu malumatlar1n ham1s1n1 fayla yaz1r. Sonra aral1q yadda§a yeni malumatlar1 avvaldan yazmaga ba§lay1r.

//OFSTR3.CPP

#include <io.h>

#include <fcntl.h>

#include <fstream.h>

main()

{ int OutputHandle = open("misal3.dat", O\_RDWR); write(OutputHandle, "Kohne idareetme\n", 12); ofstream Output(OutputHandle);

Output<<"Axina yazma\n"; Output<<8<<' '<<7.78<<"\n";

Output.close(); return 0;

}

#### Fayldan Oxuma

Fayylar1n ax1n olaraq tayin edilmasi üçün ifstream() sinfindan istifada olunur. Bu sinfin da ofstream() sinfinda oldugu kimi dörd müxtalif layihalandiricisi vard1r.

ifstream();

Bu layihalandirici, yaln1z oxuma maqsadli ax1n tayin edir. Bu ax1n1n hans1 fayl oldugu open() üzv funksiyas1 ila tayin edilir. open() funksiyas1n1n yaz1l1§1 bundan avvalki paraqraflarda oldugu kimidir. Yaln1z tip kimi tayin olunmu§ ikinci parametr oxuma maqsadli fayllar üçün ios::in olmal1d1r. Ax1n neca aç1lmas1ndan as1l1 olmayaraq close() üzv funksiyas1 ila baglanmal1d1r.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

MISAL3.DAT fayl1na yaz1lan malumatlar

Kohne idareetme Axina yazma

8 7.78

//IFSTR1.CPP

#include <conio.h>

#include <fstream.h>

main()

{ clrscr(); ifstream Input;

int X, Y; double D, E; char \*Setir;

Input.open("input.dat", ios::in); if (Input.rdstate())

{ cerr<<"Fayli acma xetasi.\n";

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

INPUT.DAT fayl1

return 1;

}

Input>>X>>D; Input>>Y>>E; Input>>Setir; Input.close();

cout<<endl<<Setir<<endl;

cout<<"Tam ededler\t"<<X<<'\t'<<Y<<endl; cout<<"Heqiqi ededler\t"<<D<<'\t'<<E<<endl;

return 0;

}

3 67.8

90 12e34

Misal

Proqram ç1x1§1

Misal

Tam ededler 3

Heqiqi ededler 67.8

90

1.2e+35

ifstream(char \* *fayltn\_adt*);

Bu layihalandirici *fayltn\_adt* ila verilan fayl1 ax1nla alaqalandirarak istifada edilmasini tamin edir. Bu halda *fayltn\_adt* adl1 fayl mövcud olmal1d1r.

ifstream(int *handle*);

Bu layihalandirici avvalcadan oxuma maqsadi ila aç1lm1§ fayl1n ax1n kimi istifada edilmasi üçün tayin olunmu§ layihalandiricidir. Bu layihalandirici ila birlikda istifada edilan *handle* ila verilmi§ fayl bu tayindan sonra yaln1z ax1n kimi istifada edilmalidir. 8ks halda fayl daxilinda sahv malumatlara rast gala bilarsiniz.

ifstream(int *handle*, char \**buffer*, int *uzunluq*);

ifstream(int *handle*); layihalandiricisi ila eyni maqsadlar üçün istifadasi mümkün olan bu layihalandirici *buffer* ila verilan *uzunluq* uzunlugundak1 aral1q yadda§ vasitasila fayldan oxuma amaliyyat1n1 yerina yetirir. 8vvalca aral1q yadda§ dolana qadar fayldan bu yadda§a malumat oxunur. Sonra oxuma amaliyyat1 bu yadda§ üzarindan icra olunur. Aral1q yadda§dak1 malumatlar1n ham1s1 oxunduqdan sonra bu yadda§a fayldan yeni malumatlar oxunaraq amaliyyat davam etdirilir.

### 7.7 Obyektlar va Axinlar

Proqramç1n1n özünün tayin edacayi obyektlari ax1nlara << operatoru ila yaz1b va >> operatoru ila oxuya bilmasi üçün bu obyektlar (siniflar) üçün uygun operatorlar1 da tayin etmalidir.

Point::Point(int x, int y)

{ \_X = x;

\_Y = y;

}

int Point::X() const

{ return \_X; }

int Point::Y() const

{ return \_Y; }

void Point::X(int x)

{ \_X = x; }

void Point::Y(int y)

{ \_Y = y; }

void Point::MoveRel(int dx, int dy)

{ \_X += dx;

\_Y += dy;

if(\_X < 0 || \_X > getmaxx())

\_X = getmaxx() / 2;

if(\_Y < 0 || \_Y > getmaxy())

\_Y = getmaxy() / 2;

}

void Point::Draw() const

{ line(\_X - 2, \_Y - 2, \_X + 2, \_Y + 2);

line(\_X - 2, \_Y + 2, \_X + 2, \_Y - 2);

}

ostream& operator<<(ostream& stream, const Point& point)

{ stream<<point.X()<<' '<<point.Y(); return stream;

}

istream& operator<<(istream& stream, Point& point)

{ int x, y;

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Bu operator funksiyalar1 ümumi §akilda a§ag1dak1 kimi olmal1d1r:

ostream& operator<<

(ostream& *Stream*, const *YeniClass*& *Obyekt*);

istream& operator>>

(istream& *Stream*, const *YeniClass*& *Obyekt*);

//POINT.CPP

#include <iostream.h> class Point

{ int \_X, \_Y;

public:

Point(int x = 0, int y = 0);

int X() const; int Y() const;

void X(int); void Y(int);

void MoveRel(int, int); virtual void Draw() const;

};

ostream& operator<<(ostream&, const Point&); istream& operator>>(istream&, Point&);

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include <graphics.h>

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

stream>>x>>y; point.X(x);

point.Y(y);

return stream;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* class PointArray

{ Point \*\_Array; int \_Size;

public: PointArray(int);

~PointArray(); int Size() const;

void Read(); void Write();

void Load(char\*); void Save(char\*);

void Draw() const; void Animate() const;

protected:

void Create(int); void Destroy();

};

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include <fstream.h>

#include <stdio.h>

#include <stdlib.h>

#include <conio.h>

PointArray::PointArray(int size)

{ Create(size); }

PointArray::~PointArray()

{ Destroy(); }

int PointArray::Size() const

{ return \_Size; }

void PointArray::Read()

{ cout<<"\nNoqtelerin X ve Y qiymetlerini\n"; cout<<"aralarina bosluq qoyaraq daxil edin\n";

int i;

for(i = 0; i < \_Size; i++)

{ cout<<i+1<<">"; cin>>\_Array[i];

}

}

void PointArray::Write()

{ cout<<"Noqtenin qiymeleri\n";

int i;

for(i = 0; i < \_Size; i++) cout<<i+1<<'\t'<<\_Array[i]<<endl;

}

void PointArray::Load(char\* fname)

{ ifstream Input(fname); if(Input.rdstate())

return;

Destroy(); int size; Input>>size;

Create(size);

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

int i;

for(i = 0; i < \_Size; i++) Input>> \_Array[i];

}

void PointArray::Save(char\* fname)

{ ofstream Output(fname); if(Output.rdstate()) return;

int i; Output<<\_Size<<endl; for(i = 0; i < \_Size; i++)

Output<<\_Array[i]<<endl;

}

void PointArray::Draw() const

{ int i;

for(i = 0; i < \_Size; i++)

\_Array[i].Draw();

}

void PointArray::Animate() const

{ randomize(); int i = 0;

setwritemode(XOR\_PUT); Draw();

outtextxy(0, getmaxy() - 12, "Dayandirmaq ucun bir duymeye sixin");

while(!kbhit())

{ if(i >= \_Size) i = 0;

\_Array[i].Draw();

\_Array[i].MoveRel(random(3) - 1, random(3) - 1);

\_Array[i].Draw(); i++;

}

getch();

}

void PointArray::Create(int size)

{ \_Array = new Point [\_Size = size]; if(!\_Array)

abort();

}

void PointArray::Destroy()

{ delete []\_Array; }

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* main()

{ clrscr();

PointArray Noqteler(15); Noqteler.Read();

clrscr(); Noqteler.Write();

cout<<"\ndavam etmek ucun bir duymeye sixin\n"; getch();

Noqteler.Save("Noqte.ntk"); Noqteler.Load("Noqte.ntk");

cout<<"Fayldan oxunan qiymetler\n"; Noqteler.Write();

cout<<"\ndavam etmek ucun bir duymeye sixin\n"; getch();

int D = DETECT, M = 0;

initgraph(&D, &M, "c:\\borlandc\\bgi"); if(graphresult() != grOk)

abort(); Noqteler.Animate();

closegraph();

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

cout<<"Animate funksiyasindan sonraki qiymetler\n"; Noqteler.Write();

cout<<"\ndavam etmek ucun bir duymeye sixin\n"; getch();

return 0;

}



# VIII F8SiL

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

## CLASS KiTABXANASI

### Container Class Kitabxanasi

Container Class kitabxanas1 proqramç1lar tarafindan çox istifada edilan müayyan tayinlar, malumatlar strukturu, alqoritmlarin yaz1lmas1 va takmilla§dirilmasinda, problem üzarinda çox vaxt sarf etma zaman1 ahamiyyatli rol oynay1r. Bu tayinlar va malumatlar strukturu C++-da avvalcadan haz1rlanaraq proqramç1lara taqdim edilmi§dir. Borland C++ va Turbo C++-da bu strukturlar1n istifada edilmasi ila alaqadar

§arh vermaya ehtiyac vard1r.

Bu strukturlar1n ba§l1q fayllar1 C:\BORLANDC\CLASSLIB\INCLUDE qovlugunda saxlan1l1r. Ba§l1q fayllar1n1n proqrama daxil edilmasi üçün onlar1n ad1 ila birlikda bu satir da (yol) yaz1lmal1d1r. Bu satri OPTIONS menyusundan DIRECTORIES amrini seçarak, aç1lacaq dialoq pancarasinda INCLUDE parametrina uygun galan satra daxil etmak laz1md1r.

Bundan ba§qa class kitabxanalar1n1n LINK

marhalasinda ola bilmasi üçün kitabxanalar1n daxilinda

oldugu \BORLANDC\CLASSLIB\LIB yolu INCLUDE yoluna ox§ar tayin edilmalidir. Bu yolu OPTIONS menyusundan DIRECTORIES amrini seçarak, aç1lacaq dialoq pancarasinda LIBRARIES parametrina uygun galan satra yazmaq laz1md1r. Normal olaraq istifada edilacak kitabxanan1n ad1 PROJECT daxilinda göstarilmali va ya LINK marhalasinda kitabxana ad1 olaraq verilmalidir. Lakin Borland C++ paketinda OPTIONS menyusundan LINKER alt menyusunu, sonra da LIBRARIES amri, aç1lacaq dialoq pancarasinda “Container Class Library” sahasindaki STATIC va ya DINAMIC parametrlarindan biri seçilmalidir.

### Tayin Olunmu§ Siniflar

Tayin olunmu§ üç sinif vard1r:

1. Verilanlari saxlayan siniflar:
   1. bir malumatdan ibarat olan siniflar;
   2. iki malumat aras1nda alaqa quran siniflar;
   3. çox malumatdan ibarat olan siniflar.
2. Sürücülarla i§layan siniflar;
3. Kömakçi siniflar.

Siniflarin növlari a§ag1dak1lard1r:

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

### Tayinlar va Tiplar

Sortable

BaseDate

Date

BaseTime

Time

Deque

Queue

Association

Stack

Object

Error

Stringt

Container

Collection

Btree

DoubleList

List

#### Tip va Sinif Kodlari

Kitabxanan1n kodla§d1r1lmas1nda istifada edilan

tayinlar va tiplar a§ag1dak1lard1r:

##### classType

typedef unsigned int classType;

Siniflari bir-birindan ay1rmaq üçün qiymat olaraq

har sinfa uygun galan bir tam adad verilmi§dir.

|  |  |
| --- | --- |
| HasTable | |
|  |  |
| Bag | |
|  |  |
| Set | |
|  |  |
| Dictionary | |

Bunlar a§ag1dak1lard1r:

AbstractArray

Array

SortedArray

|  |  |  |
| --- | --- | --- |
| **Sinif Ad1** | **Simvolik Sabit** | **adadi Qiymati** |
| Array | arrayClass | 16 |
| Association | associationClass | 15 |
| Bag | bagClass | 12 |
| Btree | btreeClass | 22 |
| Collection | collectionClass | 10 |
| Container | containerClass | 6 |
| Date | dateClass | 21 |
| Deque | dequeClass | 9 |
| Dictionary | dictionaryClass | 14 |
| DoubleList | doubleListClass | 19 |
| DoubleListElement | doubleListElementClass | 5 |
| Error | errorClass | 1 |
| HashTable | hashTableClass | 11 |
| List | listClass | 18 |
| ListElement | listElementClass | 4 |

##### sizeType

|  |  |  |
| --- | --- | --- |
| **Sinif Ad1** | **Simvolik Sabit** | **adadi Qiymati** |
| Object | objectClass | 0 |
| PriorityQueue | priorityQueueClass | 23 |
| Queue | queueClass | 8 |
| Set | setClass | 13 |
| Sortable | sortableClass | 2 |
| SortedArray | sortedArrayClass | 17 |
| Stack | stackClass | 7 |
| String | stringClass | 3 |
| Time | timeClass | 20 |

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

0 ila lastLibClass (qiymati 255) aras1ndak1 qiymatlar Borland tarafindan istifada edilmak üçün nazarda tutulmu§dur. firstUserClass (qiymati 256) ila

lastClass (qiymati 65568) aras1ndak1 qiymatlar isa proqramç1lar1n öz siniflarini tayin edarkan istifada eda bilmalari üçün nazarda tutulmu§dur.

Yuxar1dak1 cadvaldan da göründüyü kimi simvolik sinif qiymatlari sinfin ad1n1n ilk harfinin kiçik yaz1lmas1 va sonuna Class sözünün alava edilmasi ila göstarilmi§dir.

##### hashValueType

typedef unsigned int hashValueType;

Yadda§dak1 obyektlara müraciat etmak maqsadila obyektlarin müayyan bir qiymata göra siniflandirilmasi va bu siniflandirmaya göra axtar1lmas1 üçün onlar1n verdiyi bir adadi qiymatdir.

typedef unsigned int sizeType;

Yadda§da saxlan1lan obyektlarin say1n1 tayin etmak üçün istifada edilir.

##### iterFuncType

typedef void (\*iterFuncType)(class Object&, void\*);

forEach üzv funksiyas1n1n tayininda istifada edilir.

##### condFuncType

typedef int countType;

Yadda§dak1 elementlari saymaq üçün istifada olunan bir tip tayinidir.

Burada göstarilan bütün tiplar va sabitlar clstypes.h

ba§l1q fayl1 daxilinda tayin edilmi§di.

#### Sahv Kodlarinin Tayini

Burada §arh edilan obyekt siniflarinin tatbiqinda proqram1n icras1n1n davam etmasina angal olan sahvlar meydana galdiyi zaman proqramdan exit() amri ila ç1x1l1r.

Meydana galan sahvin sababina göra exit() funksiyas1n1n (eyni zamanda proqram1n) ç1x1§ kodlar1 va adadi qiymatlari a§ag1da göstarilmi§dir:

|  |  |
| --- | --- |
| **Sinif Ad1** | **Simvolik Sabit** |
| BtreeIterator | BTREE.H |
| Collection | COLLECT.H |
| Container | CONTAIN.H |
| ContainerIterator | CONTAIN.H |
| Date | DATE.H |
| Deque | DEQUE.H |
| Dictionary | DICT.H |
| DoubleList | DBLLIST.H |
| DoubleListElement | DLSTELEM.H |
| Error | ERROR.H |
| HashTable | HASHTBL.H |
| HashTableIterator | HASHTBL.H |
| List | LIST.H |
| ListElement | LSTELEM.H |
| ListIterator | LIST.H |
| Object | OBJECT.H |
| PriorityQueue | PRIORTYQ.H |
| Queue | QUEUE.H |
| Set | SET.H |
| Sortable | SORTABLE.H |
| SortedArray | SORTARRY.H |
| Stack | STACK.H |
| String | STRING.H |
| Time | TIME.H |

|  |  |  |
| --- | --- | --- |
| EEXPAND | 2 |  |
| ENOMEM | 3 | Yadda$da kifayat qadar yer olmadngn  zaman saha ayrnlmadngn hallarda |
| ENOTSTOP | 4 | SortedArray sinfinin snralana bilmamasi halnnda |
| ENOTASSOC | 5 | Dictionary sinfinin har hansn bir obyektina Association sinfindan olmayan bir obyektin yerla$dirilamsina cahd edilmasi halnnda |

Bu sahv kodlar1 “clsdef.h” ba§l1q fayl1 daxilinda tayin edilmi§dir.

#### Ba§fiq Fayllari va Tayin Edilmi§ Siniflar

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Siniflarin tayin edildiklari ba§l1q fayllar1 a§ag1dak1lard1r:

|  |  |
| --- | --- |
| **Sinif Ad1** | **Simvolik Sabit** |
| AbstractArray | ABSTARRY.H |
| Array | ARRAY.H |
| Association | ASSOC.H |
| Bag | BAG.H |
| BaseDate | LDATE.H |
| BaseTime | LTIME.H |
| Btree | BTREE.H |

### Siniflar

#### Object

Object **mücarrad sinif** (abstract class) olmaqla barabar digar bütün container class kitabxanas1n1n obyektlarinin töranmasinda istifada edilan ilk baza

sinfidir. Bu sinif töranan digar siniflarin sada xüsusiyyatlarini tayin edir.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Bu sinif daxilinda tayin edilan üzv funksiyalar1n1n prototiplari va vazifalari a§ag1dak1lard1r:

virtual classType isA() const = 0;

Bu funksiya bir sinfi tayin edan va sadaca o sinfa aid olan obyektlarin da§1ya bilacayi xüsusi bir kodu qiymat kimi geri qaytar1r. Bu funksiya obyektlarin eyni sinifdan olub olmad1qlar1n1 müayyanla§dirmak üçün istifada edilir. Yeni töradilan Oject bazal1 har obyekt üçün yaz1lmas1 vacibdir.

virtual char\* nameOf() const = 0;

Bu funksiya obyektin xüsusi ad1 ila geri qay1d1r. Sinfin obyektlarinin ad1 yoxdursa, onun ad1 ila geri qay1d1r. Yeni töradilan Object bazal1 har obyekt üçün yaz1lmas1 vacibdir.

virtual hashValueType hashValue() const = 0;

Malumatlar strukturunun bir qrup daxilinda saxlan1lmas1 müxtalif formalarda ola bilar. Saxlan1lan malumata müraciat müddati bu forman1n tayin edilmasinda çox ahamiyyatlidir. Bunun üçün bir çox malumatlar strukturu va müraciat mexanizmlari tayin

edilmi§dir. Bu mexanizmlardan biri da hashing adland1r1lan qrupla§d1rma üsuludur. Bu üsulda malumatlar bir neça qrupa ayr1l1r va har hans1 bir malumata bu qrupdan yaln1z birinin daxilinda olma haqq1 verilir. Bir malumata müraciat etmak istadiyiniz zaman o malumat1n hans1 qrup daxilinda ola bilacayini bildikda, onu yaln1z o qrup daxilinda axtarars1n1z. Bu müraciat müddatini çox q1sald1r.

Masalan, malumatlar strukturunda ada göra axtar1§ apar1ld1g1 zaman malumatlar ad1n ilk harfina göra qrupla§d1r1larsa, bütün malumatlar1 32 müxtalif qrupa ay1rmaq laz1md1r. Bela bir malumatlar strukturunda “Kanan Seyidzada” adl1 malumata müraciat etmak istayarsinizsa, sadaca, “K” qrupuna baxmag1n1z kifayatdir. Bu misalda “Kanan Seyidzada” adl1 bir malumat1n qrup müraciat kodu (hash value) “K”-dir.

Bu üsul container class kitabxanas1 daxilinda HashTable sinfi tarafindan tatbiq edilir. Bu sinfin proqramç1dan gözladiyi yaln1z bir §ey isa, HashTable va ya bu sinifdan töranan siniflar daxilinda istifada edilacak malumat siniflarinin obyektlarinin saxlad1qlar1 malumat1 tamsil edan qrup nömrasi vermalaridir. Object sinfindan töranan har obyektin bu qiymati vera bilmasi üçün hashValue() üzv funksiyas1ndan istifada edilir. Bu funksiya obyektin daxilinda olmas1 laz1m galan qrupun nömrasi ila geri qay1d1r. Object sinfindan töranan har obyekt üçün bu üzv funksiyas1n1n yaz1lmas1 vacibdir.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

virtual int isEqual(const Object& *Test*) const = 0;

Object sinfindan töranan har sinif üçün tayin edilmasi vacib olan bir digar üzv funksiya isa isEqual funksiyas1d1r. Bu funksiya bir parametri olan Object sinfinin *Test* obyekti ila asas obyektin eyni qiymatlara malik olub olmad1g1na nazarat edir. Bu funksiya iki Object obyektinin barabar olub olmad1g1na nazarat etmak üçün istifada olunur. Bunun üçün obyektlarin ilk növbada eyni sinifdan olmas1 vacibdir. Bu nazarat operator == vasitasila olunur. Ancaq eyni sinfa daxil olan iki obyekt müqayisa edilarsa, isEqual sinfi çag1r1l1r. Bu da *Test* parametrinin isEqual sinfinin aid oldugu sinif ila eyni sinifdan olmas1 manas1na galir. Bu halda *Test* tip çevrilmasi yerina yetirilarak üzv dayi§kanlari müqayisa edilmalidir. Bu müqayisa naticasinda iki obyektin eyni qiymata malik olmas1na qarar verilarsa 1, müxtalif qiymatlara malik olduqlar1na dair qarar verilarsa, 0 qiymati geri qaytar1l1r.

virtual int isSortable() const;

Bu funksiya obyektin (sinfin) s1ralan1b s1ralanmayacag1n1 tayin edan funksiyad1r. 8gar bir sinfin obyektlari s1ralana bilarsa 1, aks halda 0 qiymati bu funksiyadan geri qaytar1l1r. Bu qayda ila obyektlari iki hissaya ay1rmaq mümkündür: s1ralana bilan obyektlar va

s1ralanmayan obyektlar. Bunlardan s1ralanmayan obyektlar Object, s1ralana bilan obyektlar isa Sortable sinfindan töranan siniflardan alda edilir. Object sinfi bu funksiya üçün 0, Sortable sinfi isa 1 qiymatini geri qaytar1r. Bu sababdan, bu funksiya har sinif üçün yenidan yaz1lmal1d1r.

virtual int isAssociation() const;

isSortable funksiyas1na ox§ar olaraq, isAssociation funksiyas1 bir obyektin iki obyekt aras1ndak1 alaqani (Association) göstaran obyekt olub olmad1g1n1, yani bu obyektin strukturunda bir-biri ila alaqalandirilmi§ iki müxtalif obyektin olub olmad1g1n1 bildirir. Normal olaraq alaqali malumatlardan ibarat obyektlar Association obyektindan törandiyi üçün digar siniflar üçün bu funksiya 0 qiymatini geri qaytar1r.

virtual void forEach(iterFuncType *Func*, void\* *Data*) const;

Bu funksiya da yenidan yaz1lmas1 laz1m olmayan bir funksiyad1r. forEach ilk parametri olan *Func* funksiyas1n1 çag1r1r. Çag1r1lan *Func* funksiyas1n1n iki parametrindan birincisi obyektin özü, ikincisi isa forEach funksiyas1n1n ikinci parametri olan *Data* göstaricisidir.

Bu funksiya haqiqatan da bir çox obyektin daxil oldugu malumatlar strukturu üzarinda bu struktura daxil

olan bütün obyektlara eyni funksiyan1n (*Func*) tatbiq edilmasini, *Data* göstaricisi isa bu funksiyan1n ehtiyac duydugu qiymatlarin funksiyaya müraciat etmasini tamin edir.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

virtual Object& firstThat(condFuncType *Func*, void\* *Data*) const;

firstThat funksiyas1 da forEach funksiyas1na ox§ar olaraq *Func* nazarat funksiyas1n1 çag1r1r. Obyektin özünü birinci, *Data* göstaricisini da ikinci parametr kimi bu funksiyaya göndarir. *Func* funksiyas1ndan geri qaytar1lan qiymat s1f1rdan farqli olarsa, obyekti (\*this qiymatini), s1f1r olarsa, NOOBJECT qiymatini geri qaytar1r.

Bu funksiya bir obyektin obyekt qrupu daxilinda müayyan §artlara uygun olaraq malumatlar strukturu daxilindaki yerla§masina göra ilk obyekt olub olmad1g1n1 anlamaq üçün istifada edilir.

virtual Object& lastThat(condFuncType *Func*, void\* *Data*) const;

Bu funksiya firstThat funksiyas1n1n i§lama prinsipina uygun olaraq icra olunur. Yaln1z obyektin malumatlar qrupu daxilinda yerla§masina göra müayyan olunmu§ §artlari tamin edan an son obyekt olub olmad1g1n1 anlamaq üçün istifada edilir.

virtual void printOn(ostream& *Stream*) const = 0;

Bu funksiya tayin olunan har yeni sinif üçün yaz1lmas1 vacib olan bir funksiyad1r. *Stream* ila tayin olunan ax1na obyektin ifada etdiyi malumatlar1n yaz1lmas1n1 tamin edir.

friend ostream& operator<<(ostream& *Stream*, const Object&

*obyekt*);

Tayin olunan bu operator funksiyas1 ila Object sinfindan töranan bütün siniflarin obyektlarinin ostream va bundan töranan ax1nlar üzarina yaz1la bilacayini göstarir. Bu operator funksiyas1

*obyekt*.printOn(*Stream*);

§aklinda tasira sabab olur.

int operator ==(const Object& *Test1*, const Object& *Test2*); int operator !=(const Object& *Test1*, const Object& *Test2*);

Bu iki operator funksiyas1 *Test1* va *Test2* Object obyektlarinin müqayisasini tamin edir. Bu funksiyalardan birincisi barabar olmalar1 hal1nda, digari isa barabar olmamalar1 hal1nda s1f1rdan farqli bir qiymat ila geri qay1d1r. Bir misala baxaq.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

//IKILIK.CPP

#include <conio.h>

#include <object.h>

#include <clstypes.h>

#define ikilikClass firstUserClass class Ikilik : public Object

{ int x, y;

public:

Ikilik(int = 0, int = 0);

int X() const { return x; } void X(int);

int Y() const { return y; } void Y(int);

classType isA() const; char\* nameOf() const;

hashValueType hashValue() const; int isEqual(const Object&) const; void printOn(ostream&) const;

};

Ikilik::Ikilik(int a, int b)

{ x = a; y = b;

}

void Ikilik::X(int a)

{ x = a; }

void Ikilik::Y(int b)

{ y = b; }

classType Ikilik::isA() const

{ return ikilikClass; }

char\* Ikilik::nameOf() const

{ return "Ikilik"; }

hashValueType Ikilik::hashValue() const

{ return 0; }

int Ikilik::isEqual(const Object& Test) const

{ return ((Ikilik&)Test).x == x && ((Ikilik&)Test).y == y; }

void Ikilik::printOn(ostream& Stream) const

{ Stream<<nameOf()<<"("<<x<<","<<y<<")\n"; }

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* struct Miqdar{

int Dx, Dy;

};

void Surustur(Object& \_Obyekt, void\* \_Data)

{ Ikilik& Obyekt = (Ikilik&)\_Obyekt;

struct Miqdar \*Data = (struct Miqdar\*)\_Data;

Obyekt.X(Obyekt.X() + Data->Dx); Obyekt.Y(Obyekt.Y() + Data->Dy);

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* main()

{ clrscr();

Ikilik A(30, 10);

Ikilik B(40, 50);

struct Miqdar SurusturmeMiqdari = { 10, 40 }; cout<<A<<"\n\tA obyekti B obyektine ";

if(A == B) cout<<"beraberdir\n"; else cout<<"beraber deyildir\n";

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

#### Error

A.forEach(Surustur, &SurusturmeMiqdari); cout<<A<<"\n\tA obyekti B obyektine ";

if(A == B) cout<<"beraberdir\n"; else cout<<"beraber deyildir\n";

return 0;

}

Error sinfi Class kitabxanas1 daxilinda meydana galan sahv vaziyyatlarini haqiqi qiymatlardan ay1rmaq üçün istifada edilan kömakçi sinifdir. Masalan, bir Object massivinin bo§ olan, hala obyekt yerla§dirilmami§ elementlarina Error sinfinin obyektlari manimsadilarak bu hal müayyanla§mi§ olur. Ümumiyyatla bu amaliyyat üçün istifada olunan NOOBJECT bu sinfin bir obyektidir. NOOBJECT obyektinin tayin edilmasi üçün bu sinif töradilmi§dir. Error sinfi Object sinfinin xüsusiyyatlarini tahvil almaqla yana§1 bazilarini da dayi§dirmi§dir.

Error sinfi ila barabar yenidan yaz1lan üzv funksiyalar1n davran1§lar1 a§ag1dak1 kimidir:

|  |  |
| --- | --- |
| **Üzv funksiya** | **Qiymati** |
| isA() | errorClass |
| nameOff() | “Error” |

#### Sortable

Sortable öz aralar1nda s1ralana bilan obyektlarin siniflarinin tayin edilmasi üçün nazarda tutulmu§ bir sinifdir. Sortable sinfi çox istifada olunmur. Sadaca s1ralana bilan obyektlarin töradilmasi üçün haz1rlanm1§ baza sinfidir.

Törandiyi obyekt sinfindan farqli davranan üzv funksiyalar1 va geri qaytard1g1 qiymatlar a§ag1dak1lard1r:

|  |  |
| --- | --- |
| **Üzv funksiya** | **Qiymati** |
| isA() | sortableClass |
| isSortable() | 1 |

Burada isSortable() funksiyas1n1n 1 qiymatini qaytard1g1na diqqat edin. Bundan sonra Sortable sinfindan töradilacak siniflar s1ralana bilan obyektlarin siniflari olacaqlar1 üçün bu siniflari töradarkan isSortable() funksiyas1n1 yenidan yazmaq macburiyyatinda qalmayacag1q. Lakin bu dafa obyektlarin hans1n1n kiçik, hans1n1n böyük oldugunu müayyanla§dirmak üçün isLessThan funksiyas1na ehtiyac duyacag1q.

virtual int isLessThan(const Object& *Test*) const;

Bu funksiya isEqual funksiyas1na ox§ar icra olunur.

isLessThan funksiyas1 tayin edilmakda olan asas sinif ila

eyni sinifdan olan *Test* obyektinin müqayisasini apararaq asas obyektin *Test* obyektindan kiçik olmas1 (s1ralamada daha avval yerla§masi) hal1nda s1f1rdan farqli bir qiymat ila, böyük va ya barabar olmas1 hal1nda isa s1f1r qiymati ila geri qaytar1l1r.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Sortable sinfi üçün isLessThan funksiyas1na asaslanaraq <=, <, >, >= operatorlar1 yenidan tayin edilmi§dir. Bu operatorlar1 Sortable sinfi va bu sinifdan töranan digar siniflar üçün istifada etmak mümkündür.

#### String

String sinfi proqramlarda çox istifada edilan harf- raqam ifadalarinin C++ ila Class kitabxanas1 daxilinda yenidan tayin edilmasi üçündür. String Sortable sinfindan töranmi§ bir sinifdir.

iki müxtalif layihalandiricidan ibaratdir:

String(const char \**S*);

Bu layihalandirici *S* ila verilan harf-raqam ifadasinin bir nüsxasini obyekt daxilinda saxlay1r.

String(String& *S*);

Bu layihalandirici isa avvalcadan tayin edilmi§ bir

String obyektinin saxlad1g1 harf-raqam qiymatinin tayin

edilacak yeni harf-raqam sinfina köçürülmasini tamin edir.

~String();

Bu sinfin sahib oldugu yoxedici funksiya layihalandirici tarafindan harf-raqam sahasi üçün ayr1lan yadda§1n bo§ald1lmas1n1 tamin edir.

operator const char \*() const;

Bu sinif eyni zamanda bir da (char\*) tip çevirma operatoruna malikdir. Bu operatorun kömayi ila String sinfinin obyektlari asanl1qla simvol göstaricisi olaraq istifada oluna bilarlar. Bu operator daha çox String sinfinin obyektlarinin harf-raqam qiymatini öyranmak üçündür.

String sinfi bir ax1na yaz1larsa, aslinda saxlad1g1 harf-raqam qiymatini ax1na yaz1r. String sinfinin obyektlarinin barabarliyi isa saxlad1qlar1 harf-raqam qiymatlarinin barabarliyi ila tayin olunmu§dur.

Tayin olunmu§ digar iki funksiya va qaytard1g1 qiymatlar a§ag1dak1lard1r:

|  |  |
| --- | --- |
| **Üzv funksiya** | **Qiymati** |
| isA() | stringClass |
| nameOf() | “String” |

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Birinci icran1n naticasi:

Adiniz :Etibar Soyadiniz :Seyidov

Etibar uzunlugu 6

Seyidov uzunlugu 7 Etibar==Seyidov? Yox Etibar<Seyidov? He

Yeni kimlik--> Kenan Seyidov

//DNSTRING.CPP

#include <conio.h>

#include <strng.h>

#include <iostream.h>

#include <string.h>

main()

{ clrscr();

char Buffer[80];

cout<<endl<<"Adiniz :"; cin>>Buffer;

String Ad(Buffer);

cout<<"Soyadiniz :"; cin>>Buffer;

String Soyad(Buffer);

cout<<endl<<Ad<<"\tuzunlugu "<<strlen(Ad)<<endl; cout<<Soyad<<"\tuzunlugu "<<strlen(Soyad)<<endl;

cout<<Ad<<"=="<<Soyad<<"? "

<<((Ad==Soyad)?"He":"Yox")<<endl;

cout<<Ad<<"<"<<Soyad<<"? "

<<((Ad<Soyad)?"He":"Yox")<<endl;

Ad="Kenan";

cout<<"\nYeni kimlik-->\t"<<Ad<<" "<<Soyad<<endl;

return 0;

}

ikinci icran1n naticasi:

Adiniz :Memmed Soyadiniz :Eliyev

Memmed uzunlugu 6

Eliyev uzunlugu 6 Memmed==Eliyev? He Memmed<Eliyev? Yox

Yeni kimlik--> Kenan Eliyev

#### BaseDate

BaseDate tarix malumatlar1n1n bir obyekt kimi saxlan1lmas1 üçün Sortable sinfindan töranmi§ birba§a istifada edilmayan, tarixla alaqadar töradila bilacak siniflara baza yaratmas1 üçün haz1rlanm1§ bir sinifdir.

Qorunmu§ (protected) kimi tayin edilmi§ üç layihandiricisi vard1r.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

BaseDate(unsigned int *Ay*, unsigned int *Gun*, unsigned int *Il*);

istanilan il (*Il*), ay (*Ay*) va günün (*Gun*) tarix olaraq yeni yarad1lan obyektda saxlanmas1n1 tamin edir. Sahv qiymatin daxil edilmasi proqram1n q1r1lmas1na sabab olur.

BaseDate();

Obyektin yarad1ld1g1 cari tarixi saxlay1r.

BaseDate(BaseDate& *date*);

8vvalcadan tayin olunmu§ tarix obyektina asaslanaraq yeni yarad1lan tarix obyektina avvalki obyektin göstardiyi tarixin qiymatini manimsadir.

Obyektin göstardiyi tarixin müayyan edilmasi üçün ümumi (public) kimi a§ag1dak1 üzv funksiyalar1 tayin edilmi§dir:

unsigned Day() const;

Obyektin göstardiyi tarixin gününü 1-dan 31-a qadar bir adadla ifada edir.

unsigned Month() const;

Obyektin göstardiyi tarixin ay1n1 1-dan 12-ya qadar bir adadla ifada edir.

unsigned Year() const;

Obyektin göstardiyi tarixin ilini ifada edir.

Mövcud bir tarix obyektinin göstardiyi tarixin dayi§dirilmasi talab olunarsa, bu dafa yena ümumi (public) kimi tayin olunmu§ a§ag1dak1 üç funksiya istifada edilir:

void SetDay(unsigned char *Gun*);

Bu üzv funksiyas1 tarixa manimsadilan yeni günün 1-dan 31-a qadar qiymat almas1 §artila, tarixin yaln1z gününü dayi§dirir.

void SetMonth(unsigned char *Ay*);

Bu üzv funksiyas1 tarixa manimsadilan yeni ay1n 1- dan 12-ya qadar qiymati olmaqla, tarixin yaln1z ay1n1 dayi§dirir.

void SetYear(unsigned char *Il*);

Bu üzv funksiyas1 da *Il* ila göstarilan ili tarixin ilina manimsadir.

//DNDATE.CPP

#include "tarix.h"

#include <conio.h>

#include <ldate.h>

#include <iostream.h>

Date Today;

Date BirthdayOfKenan(10, 18, 1997);

Date BirthdayOfKamran(10, 27, 1996);

Tarix Bugun;

Tarix KenaninDogumGunu(18, 10, 1997);

Tarix KamraninDogumGunu(27, 10, 1996);

main()

{ clrscr();

cout<<endl<<"Date"<<endl; cout<<"Bugun"<<Today<<endl; cout<<"Kenan"<<BirthdayOfKenan<<endl; cout<<"Kamran"<<BirthdayOfKamran<<endl; if(BirthdayOfKenan == BirthdayOfKamran) cout<<"Kenan ile Kamran eyni gunde dogulmusdur.\n"; else if(BirthdayOfKenan < BirthdayOfKamran)

cout<<"Kenan Kamrandan daha evvel dogulmusdur.\n"; else cout<<"Kenan Kamrandan daha sonra dogulmusdur.\n";

cout<<endl<<"Tarix"<<endl; cout<<"Bugun"<<Bugun<<endl; cout<<"Kenan"<<KenaninDogumGunu<<endl; cout<<"Kamran"<<KamraninDogumGunu<<endl; if(KenaninDogumGunu == KamraninDogumGunu) cout<<"Kenan ile Kamran eyni gunde dogulmusdur.\n"; else if(KenaninDogumGunu < KamraninDogumGunu)

cout<<"Kenan Kamrandan daha evvel dogulmusdur.\n"; else cout<<"Kenan Kamrandan daha sonra dogulmusdur.\n";

return 0;

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Bundan ba§qa BaseDate sinfi iki tarixin qiymatini müqayisa edir. 8gar iki tarixin il, ay va gün qiymatlari bir-birina barabardirsa, har iki tarix barabar say1l1r. S1ralama bax1m1ndan tarixlar müqayisa edilarsa, *Il* qiymati böyük olan tarix daha böyükdür. *Il* qiymatlari da eyni oldugu halda *Ay*, *Ay* qiymatlari eyni oldugu halda isa *Gun* qiymatlarina bax1laraq *Gun* qiymati daha böyük olan tarixin böyük oldugu qabul edilir.

#### Date

BaseDate sinfindan töranan Date sinfinin BaseDate sinfindan yegana farqi printOn funksiyas1n1n yenidan yaz1lmas1d1r. Bu da Date sinfinin çox istifada olunmas1n1 tamin edir. Bu funksiya Date sinfinin obyektlarinin “ay, gün, il” format1 ila ba§lamas1n1 tamin edir. Masalan, “13- 01-1970” tarixi “January, 01, 1970” olaraq göstarilan ax1na yaz1l1r. Ay ad1 göründüyü kimi ingilis dilindadir.

Date sinfinin BaseDate sinfinda oldugu kimi üç layihalandiricisi vard1r. Bu layihalandiricilar BaseDate ila eyni funksiyalar1 yerina yetirirlar. Yaln1z bu layihalandiricilar public saviyyasinda tayin edilmi§dirlar.

|  |  |
| --- | --- |
| **Üzv funksiya** | **Qiymati** |
| isA() | dateClass |
| nameOf() | “Date” |

}

//TARIX.H

#ifndef TARIX\_H

#define TARIX\_H

#include <ldate.h>

#define tarixClass 10000 class Tarix:public BaseDate

{

public: Tarix();

Tarix(unsigned int, unsigned int, unsigned int);

virtual classType isA() const; virtual char\* nameOf() const;

virtual void printOn(ostream&) const;

protected:

int day() const;

unsigned long julday() const;

};

#endif

//TARIX.CPP

#include "tarix.h"

#include <math.h>

#include <iomanip.h>

Tarix::Tarix()

{}

Tarix::Tarix(unsigned int Gun, unsigned int Ay, unsigned int Il)

:BaseDate(Ay, Gun, Il)

{ }

classType Tarix::isA() const

{ return tarixClass; }

char\* Tarix::nameOf() const

{ return "Tarix"; }

#define MaxBufferLen 30

void Tarix::printOn(ostream& Stream) const

{ static char \*AyAdlari[] = {"Yanvar", "Fevral", "Mart", "Aprel",

"May", "Iyun", "Iyul", "Avqust", "Sentyabr", "Oktyabr", "Noyabr", "Dekabr"};

static char \*GunAdlari[] = {"Bazarertesi", "Cersembe axsami", "Cersembe",

"Cume axsami", "Cume", "Sembe", "Bazar"};

char buffer[MaxBufferLen];

ostream ostr(MaxBufferLen, buffer); ostr<<setw(2)<<Day()<<" "<<AyAdlari[Month() - 1]<<" "

<<setw(4)<<Year()<<" "<<GunAdlari[day()]<<ends;

Stream<<buffer;

}

#define IGREG (15 + 31L \* (10 + 12L \* 1582))

unsigned long Tarix::julday() const

{ unsigned long jul;

int ja, jy, jm, iyyy, mm;

iyyy = Year(); mm = Month();

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Bu sinif zaman1, saat, daqiqa, saniya va saniyanin yüzda biri martabasinda günlük saxlay1r. Saatlar 0-23 aras1nda, daqiqa va saniyalar 0-59 aras1nda, saniyanin yüzda biri isa 0-99 aras1nda qiymatlar ala bilar.

if(mm > 2)

{ jy = iyyy;

jm = mm + 1;

}

else

{ jy = iyyy - 1; jm = mm + 13;

}

jul = (unsigned long) (floor(365.25 \* jy) + floor(30.6001 \* jm) + Day() + 1720995L);

if(Day() + 31L \* (Month() + 12L \* iyyy) >= IGREG)

{ ja = 0.01 \* jy;

jul += 2 - ja + (int)(0.25 \* ja);

}

return jul;

}

int Tarix::day() const

{ return (int)((julday() + 1) % 7); }

Bu sinfin üç protected saviyyasinda layihalandiricisi vard1r.

BaseTime(unsigned char *saat*, unsigned char *deqiqe* = 0, unsigned char *saniye* = 0, unsigned char *yuzdebir* = 0);

Günün istanilan bir an1n1

saat:daqiqa:saniya.yuzdabir

olaraq tayin edir. Sahv qiymatin daxil edilmasi proqram1n q1r1lmas1na sabab olur.

BaseTime();

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

#### BaseTime

BaseDate sinfina ox§ar mantiqla haz1rlanan BaseTime, Sortable sinfindan töranmi§ va tayin olunacaq zaman ila alaqadar siniflarin töradilmasina baza yaradan bir sinifdir. Sortable üzvlarini istifada edarkan eynila isA(), nameOf() va printOn() funksiyalar1n1n yenidan yaz1lmas1 talab olunur.

Obyektin yarad1lma vaxt1n1 obyektda saxlay1r.

BaseTime(BaseTime& *date*);

8vvalcadan tayin edilmi§ zaman obyektina asaslanaraq yeni yarad1lan zaman obyektina avvalki obyektin göstardiyi zaman1n qiymatini manimsadir.

BaseDate obyektinin göstardiyi tarixin öyranilmasi üçün ümumi (public) olaraq a§ag1dak1 üzv funksiyalar tayin edilmi§dir:

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

unsigned hour() const; unsigned minute() const; unsigned second() const; unsigned hundredths() const;

hour saat, minute daqiqa, second saniya, hundredths isa saniyanin yüzda birini müayyanla§dirmak üçün istifada edilir.

Mövcud bir zaman obyektinin göstardiyi zaman1n dayi§dirilmasi talab olunarsa, ümumi (public) kimi tayin edilmi§ a§ag1dak1 funksiyalar istifada edila bilar:

unsigned setHour(unsigned char *Saat*) const; unsigned setMinute(unsigned char *Deqiqe*) const; unsigned setSecond(unsigned char *Saniye*) const;

unsigned setHundredths(unsigned char *Yuzdebir*) const;

Bu funksiyalar ard1c1l olaraq zaman1n, saat, daqiqa, saniya va saniyanin yüzda birinin qiymatlarinin bir- birindan ayr1l1qda dayi§dirilmasina imkan verirlar.

BaseTime sinfi zaman malumatlar1n1n barabarliyini, har iki zaman obyektinin saat, daqiqa, saniya va saniyanin yüzda birinin qiymatlarinin barabar olmas1 kimi tayin edilmi§dir. Eyni zamanda kiçiklik alaqasini da tayin edir.

#### Time

Time isa BaseTime sinfindan töranmi§ sada bir zaman sinfidir. istifada olunmas1 bax1m1ndan ahamiyyatlidir.

Time sinfinin an çox istifada edilan funksiyalar1 a§ag1dak1lard1r:

|  |  |
| --- | --- |
| **Üzv funksiya** | **Qiymati** |
| isA() | timeClass |
| nameOf() | “Time” |

Bu sinfin da BaseTime sinfindakilara uygun olan, lakin public saviyyasinda üç layihalandirici funksiyas1 vard1r:

Time(unsigned char *saat*, unsigned char *deqiqe* = 0, unsigned char *saniye* = 0, unsigned *yuzdebir* = 0);

Günün istanilan bir an1n1n

saat:daqiqa:saniya.yuzdabir

format1nda saxlan1lmas1n1 tamin edir. Sahv qiymatin daxil edilmasi proqram1n q1r1lmas1na sabab olur.

Time();

Obyektin yarad1ld1g1 zaman1 obyektda saxlay1r.

\_setcursortype(\_NOCURSOR); int Davam = 1;

while(Davam)

{ cout<<Time()<<'\r';

if(kbhit()) Davam = getch() != 27;

}

\_setcursortype(\_SOLIDCURSOR); Time Son;

cout<<"\n\nProqramin baslama saati = "<<Baslangic; cout<<"\nProqramin bitmesi saati = "<<Son; cout<<endl;

return 0;

}

Time(BaseTime& *date*);

8vvalcadan tayin edilmi§ zaman obyektina asaslanaraq yeni yarad1lan zaman obyektina avvalki obyektin göstardiyi zaman1n qiymatini manimsadir.

printOn(ostream& *Stream*) const;

funksiyas1 zaman1 ikiraqamli adadlar §aklinda

SS:DD:ss.YY ZM

SS saat, DD daqiqa, ss saniya, YY saniyanin yüzda biri va ZM günortadan avval/günortadan sonran1 müayyanla§dirarak Stream ax1n1na yaz1r. ZM günortadan avval saat 0-12 aras1ndak1 qiymatlar üçün am, günortadan sonra saat 12-24 aras1nda isa pm §aklinda göstarilir.

Proqram ç1x1§1

Proqramdan cixmaq ucun ESC duymesini sixin... 9:47:27.60 pm

Proqramin baslama saati = 9:47:27.20 pm Proqramin bitmesi saati = 9:47:27.60 pm

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

#### Association

//TIME.CPP

#include <conio.h>

#include <ltime.h>

main(void)

{ clrscr();

Time Baslangic;

cout<<"Proqramdan cixmaq ucun ESC duymesini sixin...\n";

Obyekt sinfindan töranmi§ Association sinfi struktur olaraq tarkibinda iki Object sinfindan töranmi§ obyekti (taqdimat kimi) saxlayan bir növ xüsusi obyekt sinfidir. Association sinfi tarkibinda taqdimat

malumatlar1n1 saxlad1g1 bu iki sinif aras1nda alaqanin olmas1n1 tayin edir. Bu siniflardan birincisi **açar** (key), ikincisi isa **qiymat** (value) adland1r1l1r. Association sinfi daha çox bir qrup kimi daxilinda müayyan açara uygun galan qiymati tapmaq üçün Dictionary sinfinin tamalini yaratmaq üçün layihalandirilmi§dir.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

|  |  |
| --- | --- |
| **Üzv funksiya** | **Qiymati** |
| isA() | associationClass |
| nameOf() | “Time” |

Bu sinfin iki layihalandirici funksiyas1 vard1r:

Association(Object& *key*, Object& *value*);

Parametr siyah1s1ndak1 *key* (açar) ila *value* (qiymat) aras1nda alaqa oldugunu bilan Association obyektini tayin edir.

Association(const Association& *association*);

Bu layihalandirici mövcud *association* obyektina asaslanaraq eyni alaqali Association obyekti tayin edir.

Yenidan tayin edilmi§ digar üzv funksiyalar va onlar1n vazifalari a§ag1dak1lard1r:

hashValueType hashValue() const;

Association sinifli bir obyektin hashValue qiymati, obyektin alaqalarini tayin etdiyi obyektlardan key (açar) olaraq ayr1lm1§ obyektin hashValue qiymatidir. Value (qiymat) obyekti nazara al1nmaz. Key (açar) obyekti bu alaqa naticasinda meydana galan obyekti takba§1na tamsil edir. Association sinfindan iki obyektin qar§1la§d1r1lmas1 zaman1 eyni cür davran1§ gözlanilir.

int isEqual(const object& *Test*) const;

Bu funksiya Association sinfina daxil olan iki obyekti müqayisa edir va onlar barabar olarsa, s1f1rdan farqli bir qiymat, farqli olarsa, s1f1r qiymatini geri qaytar1r. Association sinifli iki obyektin barabarliyi iki Association obyektinin da key (açar) qiymatlarinin barabar olmas1 kimi tayin edilmi§dir.

void printOn(ostream& *Stream*) const;

Association sinfinin obyektlarinin ax1n üzarina yaz1lmas1 normal hallarda

sinif\_adn { key\_object, value\_object }

format1 ila olur. sinif\_adn isa nameOf() ila öyranildiyi üçün Association olur. Tabii ki, bu format1 Association sinfindan töradilacak yeni siniflarin printOn funksiyas1n1 yenidan yazaraq dayi§dirmak mümkündür.

Association sinfi ila tayin edilmi§ iki yeni funksiya isa Association obyektinin aralar1nda alaqa qurdugu key (açar) va value (qiymat) obyektlarina müraciat edilmasini tamin edir.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Bu iki funksiya a§ag1dak1 prototiplari ila tayin edilir:

Object& key() const; Object& value() const;

### Malumatlar Sturukturu Siniflari

Mövcud proqramla§d1rma dillari malumatlar qrupu olan massivlari (ProLog va LISP siyah1 strukturlar1n1) dastaklayir. Bu bax1mdan massivlar (cadvallar da adland1r1l1r) proqramla§d1rma dili üçün çox ahamiyyatli bir mövzunu ta§kil edir. Malumatlar sturukturu isa sadaca, massiv va ya siyah1dan ibarat deyildir. Xüsusi maqsadlar üçün oldugu kimi, ümumi maqsadlar üçün da müxtalif malumatlar qrupu tayin etmak olar.

indi da Container Class kitabxanas1nda tayin olunmu§ malumatlar strukturlar1n1 gözdan keçirak.

#### Container

Çox istifada olunmayan Container sinfi asasan mantiqi olaraq müxtalif tipli (hams1 eyni da ola bilar)

malumatlar1 tarkibinda saxlayan va bu malumatlar1n yadda§da saxlan1lmas1n1 nizamlayan bir sinifdir. Bu sinif digar yadda§ siniflarinin töradilmasi üçün baza ta§kil etdiyi üçün çox ahamiyyatlidir.

Bu sinif da Object sinfindan törandiyi üçün onun xüsusiyyatlarini miras alm1§ va dayi§dirmi§dir.

virtual classType isA() const = 0; virtual char\* nameOf() const = 0;

virtual hashValueType hashValue() const = 0;

Bu üzv funksiyalar1 Object sinfinda oldugu kimi eyni manan1 ifada edirlar. Lakin Container sinfi daxilinda kodla§d1r1lmayaraq ondan töradilacak siniflar üçün kodla§d1r1lmalar1 vacibdir.

virtual int isEqual(const Object& Test) const;

Bu üzv funksiyas1 isa iki Container obyektinin barabarliyini yoxlay1r. Barabar olmalar1 üçün har iki Container obyektinin eyni sayda elementdan ibarat olmas1 va eyni mövqedaki element obyektlarinin barabar olmas1 §artdir.

Container sinfi ila birlikda tayin olunan yeni üzv funksiyalar1 isa a§ag1dak1lard1r:

int isEmty() const;

Container obyekti daxilinda heç bir obyekt olmazsa, s1f1r qiymati ila geri qay1d1r. Bu funksiya Container obyektinin sanki bo§ olub olmad1g1n1 müayyanla§dirir.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

countType getItemsInContainer() const;

Bu funksiyas1 isa Container obyektina daxil olan obyektlarin say1n1 protected saviyyasinda tayin etdiyi

countType itemsInContainer;

dayi§kaninda saxlay1r. Container obyektina har dafa yeni obyekt alava edildiyi zaman bu qiymat bir vahid art1r. Xaric edilan har obyekt üçün da bir vahid azal1r.

virtual void printOn(ostream& Stream) const; virtual void printHeader(ostream& Stream) const;

virtual void printSeperator(ostream& Stream) const; virtual void printTrailer(ostream& Stream) const;

Container sinfi obyektinin ax1na yaz1lmas1n1 tamin edan asas funksiya printOn() funksiyas1d1r. printOn() printHeader() funksiyas1n1 çag1raraq amaliyyata ba§lay1r. printHeader() ax1na göndarilacak malumatlar1n avvalina ba§l1q alava edilmasi maqsadila istifada olunur. Malumatlar1n avvalina alava olunan bu ba§l1q Container sinfi üçün obyektin ad1 (nameOf() ila öyranilir) va “{“ i§arasidir. printOn() funksiyas1 daha sonra Container daxilindaki obyektlari ard1c1l olaraq yazmaga ba§lay1r.

Yazd1g1 obyektlar aras1nda isa printSeperator() funksiyas1n1 çag1r1r. printSeperator() isa yaz1lan obyektlarin bir-birindan ayr1lmas1n1 tamin edir. Bunun tasiri Container sinfi üçün “,\n” masaj1n1n yaz1lmas1

§aklinda görünür. Container daxilindaki bütün obyektlar yaz1ld1qdan sonra da printTrailer() funksiyas1 çag1r1laraq yazma amaliyyat1 tamamlan1r. Bu funksiya da Container sinfi üçün “}“ simvolunun yaz1lmas1 §aklinda tasir göstarir.

virtual void forEach(iterFunctionType *iterfunc*, void \**param*);

Bu funksiya Object sinfinin tayin edilmasindan

farqlanir. forEach funksiyas1 *iterfunc* parametri ila verilan

void *iterfunc*(Object& *obyekt*, void \**param*);

§aklinda funksiyan1n, Container sinfina daxil olan bütün obyektlara ard1c1l olaraq tatbiq edilmasini tamin edir. *iterfunc* funksiyas1n1n ikinci parametri olan *param* ila forEach funksiyas1n1n ikinci parametri olan *param* eyni qiymatlardir.

virtual Object& firstThat(contFuncType *condfunc*, void \**param*) const;

virtual Object& lastThat(contFuncType *condfunc*, void \**param*) const;

Bu funksiyalar

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

int *confunc*(Object& obyekt, void\* param);

§aklinda tayin edilan va birinci parametr ila göstarilan bir müqayisa funksiyas1na Container obyekti daxilinda olan obyektlari ard1c1l olaraq tatbiq edir. *condfunc* funksiyas1 obyektlarin müayyan olunmu§ §arti ödayib ödamadiklarini yoxlay1r. Uygun §arti ödayan obyektlar üçün s1f1rdan farqli qiymati, §arti ödamayan obyektlar üçün isa s1f1r qiymatini qaytar1r.

Bu funksiyalardan firstThan *condfunc* funksiyas1ndan s1f1rdan farqli bir qiymat ila geri qaytar1lan ilk obyekti, lastThan isa son obyekti geri qaytarma qiymati kimi qaytar1r.

virtual ContainerIterator& initIterator() const = 0;

Bu funksiya isa yenilayicilar bölmasinda §arh edilacak bir yenilayici (iterator) Container sinfinin obyekti üçün haz1rlayaraq geri qaytarma qiymati kimi qaytar1r.

#### Stack

8sas malumatlar strukturu olan Stack, Last In First Out – LIFO (son giran ilk ç1xar) qaydas1 ila i§layan bir Container sinfidir. Stack strukturunda yadda§a yeni yerla§dirilacak obyekt digar obyektlarin üzarina

yerla§dirilir. Yadda§dan bir obyektin geri qaytar1lmas1 talab olundugu zaman da üstdaki obyekt geri qaytar1l1r. Araya bir obyekt alava etmak va ya aradan bir obyekti

<1xarmaq mümkün deyildir.

Stack sinfi Container sinfindan töranmi§dir.

Yenidan tayin olunduqlar1 funksiyalar

a§ag1dak1lard1r:

|  |  |
| --- | --- |
| **Üzv funksiya** | **Qiymati** |
| isA() | stackClass |
| nameOf() | “Stack” |

Stack strukturuna yeni alava edilan üzv funksiyalar isa a§ag1dak1lard1r:

void push(Object& *obyekt*);

Bu funksiya *obyekt* parametri ila verilan Object sinfindan töranmi§ bir obyektin Stack daxilina yerla§dirilmasini tamin edir. Yerla§dirma tabii olaraq Stack strukturundak1 bütün obyektlarin üzarina olacaqd1r.

Object& pop();

Bu funksiya Stack yadda§1n1n an üstdaki obyekti geri qaytarma qiymati kimi qaytararkan bu obyektin yadda§dan ç1xar1lmas1na da sabab olur.

Object& top();

cout<<"Yiginda "<<Yigin.getItemsInContainer()

<<" element var.\n"; while(!Yigin.isEmpty())

{ Object& Obyekt = Yigin.pop(); cout<<Obyekt<<endl;

delete &Obyekt;

}

cout<<endl;

return 0;

}

Bu funksiya Stack yadda§1n1n an üstdaki obyektinin öyranilmasini tamin edir va obyekti yadda§dan ç1xartm1r.

virtual ContainerIterator& initIterator() const;

Bu funksiya Stack yadda§1 üzarinda bir yenilayici tayin etmak üçün istifada olunur. Yenilayici Stack daxilindaki obyektlari son daxil olandan ilk daxil olana qadar yenila§dirir.

Proqram ç1x1§1

Verilenlerin girisini tamamlamaq ucun ‘.’ daxil edin...

> Baki

> Gence

> Quba

> Sumqaynt

> .

List {

.,

Sumqayit, Quba, Gence, Baki }

Yiginda 5 element var.

.

Sumqayit Quba Gence Baki

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

//DNSTACK.CPP

#include <conio.h>

#include <stack.h>

#include <strng.h>

main()

{ clrscr();

Stack Yigin; char oxu[80];

cout<<"Verilenlerin girisini tamamlamaq ucun \'.\' daxil edin...\n"; do

{ cout<<"> "; cin>>ws>>oxu;

Yigin.push(\*new String(oxu));

}

while(oxu[0] != '.');

cout<<Yigin<<endl;

#### Deque

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Container sinfindan töranmi§ Deque sinfi Stack sinfina ox§ar formada i§layir. Deque sinfinda obyekt yadda§a onun sol (alt) va ya sag (üst) ucundan ba§layaraq yerla§dirila bilar. Deque yadda§1ndan al1nacaq obyekt onun sol va ya sag ucundak1 obyekt ola bilar. Ortadak1 bir obyekti Deque yadda§1ndan ç1xartmaq mümkün deyildir.

Deque sinfinin Container sinfindan al1b dayi§dirdiyi funksiyalar a§ag1dak1lard1r:

|  |  |
| --- | --- |
| **Üzv funksiya** | **Qiymati** |
| isA() | dequeClass |
| nameOf() | “Deque” |

virtual ContainerIterator& initIterator() const;

Bu funksiya Deque yadda§1 üzarinda bir yenilayici tayin etmak üçün istifada edilir. Yenilayici Deque daxilindaki obyektlari Deque yadda§1n1n sol ucundan sag ucuna dogru yenilayir. 8gar obyektlarin tars ard1c1ll1qda yenilanmasi talab olunarsa, bunun üçün

ContainerIterator& initReverseIterator() const;

üzv funksiyas1ndan istifada etmak olar. Bu funksiya virtual deyildir. Sadaca Deque üçün tayin edilmi§dir.

Deque üçün tayin edilmi§ digar funksiyalar isa a§ag1dak1lard1r:

void putLeft(Object& *obyekt*); void putRight(Object& *obyekt*);

Bu funksiyalar obyektlarin Deque yadda§1na alava edilmalarini tamin edir. Bunlardan putLeft() obyektlari yadda§1n sol ucuna, putRight() isa sag ucuna yerla§dirir.

Object& peekLeft(); Object& peekRight();

Bu funksiyalar isa Deque yadda§1n1n sag va sol uclar1ndak1 obyektlarin müayyanla§dirilmasini tamin edir. peekLeft() sol ucundak1, peekRight() isa sag ucundak1 obyektlari müayyanla§dirir.

Object& getLeft(); Object& getRight();

Bu funksiyalar uygun olaraq peekLeft() va peekRight() kimi icra olunmalar1 ila barabar, uclardak1 obyektlari müayyanla§dirmakla yana§1 bu obyektlari Deque yadda§1ndan ç1xar1rlar.

//DNDEQUE.CPP

#include <conio.h>

#include <deque.h>

> Ramil

> Kamil

> .

DoubleList { Kamil, Ramil, Amil, Ismayil, Kenan,

. }

Yiginda 6 element var.

.

Kenan Ismayil Amil Ramil Kamil

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Proqram ç1x1§1

#include <strng.h>

#include <ctype.h>

main()

{ clrscr();

Deque Yigin; char oxu[80];

cout<<"Verilenlerin girisini tamamlamaq ucun \'.\' daxil edin...\n"; do

{ cout<<"> "; cin>>ws>>oxu; if(isupper(oxu[0]))

Yigin.putLeft(\*new String(oxu)); else Yigin.putRight(\*new String(oxu));

}

while(oxu[0] != '.');

cout<<Yigin<<endl;

cout<<"Yiginda "<<Yigin.getItemsInContainer()

<<" element var.\n"; while(!Yigin.isEmpty())

{ Object& Obyekt = Yigin.getRight(); cout<<Obyekt<<endl;

delete &Obyekt;

}

cout<<endl;

return 0;

}

Verilenlerin girisini tamamlamaq ucun ‘.’ daxil edin...

> Kenan

> Ismayil

> Amil

#### Queue

Deque yadda§1n1n mahdudla§d1r1lm1§ bir tatbiqi olan Queue sinfinda obyektlar yadda§a bir ucdan yerla§dirilib digar ucdan ç1xar1la bilar. Ba§qa sözla, Queue yadda§1n1n i§lama formas1 “ilk giran ilk ç1xar” (Firs In First Out – FIFO) §akilndadir. Queue bir yadda§ növbasidir.

Queue sinfinin Container sinfindan al1b dayi§dirdiyi funksiyalar a§ag1dak1lard1r:

**Üzv funksiya Qiymati**

isA() queueClass

nameOf() “Queue”

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Bu sinif üçün tayin olunmu§ üzv funksiyalar

a§ag1dak1lard1r:

void put(Object& *obyekt*);

üzv funksiyas1 obyekti növbanin sonuna alava edir.

Object& get();

üzv funksiyas1 isa növbanin avvalindaki obyekti növbadan ç1xararaq geri qaytarma qiymati olaraq qaytar1r.

Object& peekLeft(); Object& peekRight();

isa Deque sinfinda oldugu kimi Queue sinfinda növbanin ba§lang1c1nda va sonundak1 obyektlarin yaln1z müayyanla§dirilmasi üçün istifada edilir. Bunlardan peekLeft() növbanin sonundak1, peekRight() funksiyas1 isa növbanin avvalindaki obyekti göstarir.

#### PriorityQueue

PriorityQueue Container sinfindan töranmi§ xüsusi bir növba tipidir. PriorityQueue növbasi “an böyük ilk

<1xar” (“Greatest In First Out” – GIFO) va ya “an kiçik ilk

<1xar” (“Smallest In First Out”) – SIFO) qaydalar1ndan biri ila i§layir. Yani növbaya daxil olan obyektlar prioriteta göra s1ralanm1§ olur. Növbadan bir qiymatin al1nmas1 laz1m galdiyi zaman prioritetina göra birinci galan obyekt al1n1r.

PriorityQueue növbasi daxilindaki obyektlarin yerla§acaklari yer, isLessThan funksiyas1 vasitasila müayyanla§dirilir. Kiçik olan obyekt birincidir. Bir-birina barabar olan obyektlar isa “birinci galan ilk prioritetlidir” qaydas1 ila yerla§dirilir. Buna göra da PriorityQueue növbasi daxilinda ancaq Sortable sinfindan töranmi§ siniflar a§ag1dak1lard1r.

|  |  |
| --- | --- |
| **Üzv funksiya** | **Qiymati** |
| isA() | priorityQueueClass |
| nameOf() | “PriorityQueue” |

Object& get();

Növbanin ba§lang1c1ndak1 birinci olan obyekti geri qaytarma qiymati olaraq qaytararkan bu obyekti eyni zamanda növbadan ç1xar1r.

Object& peekLeft();

get() funksiyas1na ox§ar §akilda çal1§an bu funksiya isa sadaca prioritetina göra birinci olan obyektin

müayyanla§dirilmasi üçün istifada edilir. Obyekti növbadan ç1xarm1r.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

void detachLeft();

Bu funksiya isa növba daxilinda birinci olan obyektin sadaca növbadan ç1xar1lmas1n1 tamin edir. Obyektin müayyanla§dirilmasi üçün istifada edilmaz.

void put(Object& *obyekt*);

Bu funksiya parametr siyah1s1nda verilan *obyekt* obyektini, növba siyah1s1nda prioritetina uygun yerda yerla§dirir.

#### Collection

Collection sinfi Conatainer sinfinda tatbiq edilan obyektlarin müayyan qaydaya göra yadda§a yerla§dirilib, yena müayyan bir qayda ila yadda§dan xaric edilmasini nazara alaraq obyektlarin yadda§ daxilinda axtar1l1b tap1lmas1, istanilan zaman ç1xar1lmas1, istanilan yera qoyulmas1 asas1nda i§layir. Collection Container kimi istifada olunan bir sinif deyildir. Ox§ar siniflara baza yarad1r.

virtual classType isA() const = 0; virtual char\* nameOf() const = 0;

virtual hashValueType hashValue() const = 0; virtual containerIterator& initIterator() const = 0;

Container sinfindan al1nan bu üzv funksiyalar yenidan yaz1laraq Collection sinfindan töranacak siniflara verilir.

Collection üçün yeni tayin edilan üzv funksiyalar isa a§ag1dak1lard1r:

virtual void add(Object& *obyekt*) = 0;

Bu üzv funksiyas1 yeni bir obyekti kolleksiyaya daxil etmak üçün istifada edilir.

virtual void detach(const Object& *obyekt*, int *tip* = 0) = 0;

Bu üzv funksiyas1 isa birinci parametr ila verilan obyektin kolleksiyadan ç1xar1lmas1n1 tamin edir. Verilmamasi hal1nda s1f1r qiymatini alan *tip* parametrinin s1f1r qiymati üçün obyekt sadaca kolleksiyadan ç1xar1ld1g1 zaman, s1f1rdan farqli qiymatlar üçün isa kolleksiyadan

<1xar1ld1qdan sonra yadda§dan da silinacakdir.

void destroy(const Object& *obyekt*);

destroy funksiyas1 detach funksiyas1n1 *tip* parametri

1 olmas1 ila çag1r1r. Yani *obyekt* obyektina barabar

obyekti kolleksiyada taparaq oradan ç1xar1r va yadda§dan silir.

malumat1na va sonrak1 elementa müraciat eda bilacak

malumata malikdir.

virtual Object& findMember(const Object& *obyekt*) const;

findMember üzv funksiyas1 kolleksiya daxilinda *obyekt* obyektina barabar olan obyekti axtar1r va tapd1g1 ilk obyekti geri qaytar1r. Axtar1lan obyekt tap1lmazsa, NOOBJECT qiymatini qaytar1r.

M�’lumat

Sonrak1



Kök

M�’lumat

Sonrak1

M�’lumat

Sonrak1

virtual int hashMember(const Object& *obyekt*) const;

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Bu üzv funksiya isa verilan obyektin Collection daxilinda tap1l1b tap1lmad1g1na nazarat edir. Uygun obyekt Collection daxilinda tap1larsa, s1f1rdan farqli bir qiymati ila tap1lmazsa, s1f1r qiymatini qaytar1r.

#### List

Malumatlar strukturlar1 aras1nda an ahamiyyatlilarindan biri olan list strukturu, asas malumat1 göstaran **taqdimat** (referans) va bu malumatdan sonra galan siyah1 elementini göstaran bir göstaricidan ibaratdir. Bu siyah1 elementlarinin bir- birinin ard1nca alava edilmasini zancir formas1nda siyah1 strukturlar1 alda edilir. Siyah1 daxilindaki har element öz

Stack, Queue, Referans va Set kimi digar malumatlar strukturlar1 da siyah1 strukturundan istifada edarak, sadaca bu struktura malumatlar1n daxil/xaric edilmasi ila alaqadar mahdudiyyatlar gatirmi§dir. Bununla da siyah1 strukturlar1 proqramla§d1rma bax1m1ndan ahamiyyatli yer tuturlar.

List sinfinin Collection sinfindan miras ald1g1 va yenidan tayin etdiyi funksiyalar a§ag1dak1lard1r:

|  |  |
| --- | --- |
| **Üzv funksiya** | **Qiymati** |
| isA() | listClass |
| nameOf() | “List” |
| hashValue() | 0 |

Object& peekHead() const;

Bu üzv funksiyas1 siyah1ya an son alava edilan obyekti geri qaytarma qiymati olaraq qaytar1r. Siyah1 bo§ olarsa, bu funksiya NOOBJECT qiymatini qaytar1r.

Bunun xaricindaki üzv funksiyalar Collection, Container siniflarinda tayin olunan funksiyalard1r. Va bu siniflarin gördüyü amaliyyatlar1 yerina yetirirlar.

#### DoubleList

Collection sinfindan töranan DoubleList sinfi List sinfindan farqli olaraq malumat1 göstaran malumat taqdimat1 va sonrak1 siyah1 elementini göstaran sonrak1 malumat taqdimat1 ila barabar avvalki malumat elementini göstaran avvalki malumat taqdimat1na da malikdir.

Kök



8vv�lki

Sonrak1

8vv�lki

Sonrak1

Bo�

Bo�

istiqamatda da iralilamak mümkün oldugundan bazi alqoritmlar üçün daha uygun bir strukturdur.

Bu sinif daxilinda avvalki va sonrak1 hallar ila alaqadar eyni amaliyyat1 yerina yetiran simmetrik iki amr har zaman mövcuddur.

void addAtHead(Object& *obyekt*); void addAtTail(Object& *obyekt*);

Bu funksiyalar Collection üçün tayin edilan add amri kimi icra olunur. Aralar1ndak1 asas farq, addAtHeader obyekti siyah1n1n avvalina alava etdiyi halda, addAtTail isa sonuna alava edir.

void destroyFromHead(const Object& *obyekt*);

Bu funksiya siyah1n1 avvaldan ax1ra qadar gözdan keçirarak *obyekt* obyektina barabar olan tapd1g1 ilk obyekti siyah1dan ç1xar1r va yadda§dan silir.

void destroyFromTail(const Object& *obyekt*);

M�’lumat M�’lumat

Son

Bu funksiya isa siyah1n1 ax1rdan avvala qadar gözdan keçirarak tapd1g1 *obyekt* obyektina barabar olan ilk obyekti siyah1dan ç1xar1r va yadda§dan silir.

Bu struktur biristiqamatli siyah1 strukturlar1na nisbatan daha çox yadda§ talab edir, lakin har iki

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

void detachFromHead(const Object& *obyekt*, int *tip* = 0); void detachFromTail(const Object& *obyekt*, int *tip* = 0);

Bu funksiyalar siyah1n1 gözdan keçirarak (detachFromHead avvaldan ax1ra qadar, detachFormTail ax1rdan avvala qadar) qar§1lar1na ç1xd1g1 ilk obyekti sadaca siyah1dan ç1xar1rlar. *tip* qiymatinin 0-dan farqli olmas1 obyektin yadda§dan silinmasi amaliyyat1n1n da eyni zamanda icra edilmasini, yani bu amrlarin destroyFromHead va ya destroyFromTail funksiyalar1na uygun icra edilmasini tamin edir.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Object peekAtHead() const;

siyah1n1n avvalindaki,

Object peekAtTail() const;

isa siyah1n1n sonundak1 obyektin müayyanla§dirilmasi üçün istifada edilir.

virtual ContainerIterator& initIterator() const;

Bu funksiya siyah1 malumatlar1n1n siyah1n1n avvalindan ax1r1na dogru,

virtual ContainerIterator& initReverseIterator() const;

isa siyah1n1n ax1r1ndan avvalina dogru yenilanmasini

tamin edir.

Collection üçün tayin edilan bazi funksiyalar DoubleList üçün uygun olaraq icra olunurlar. Bu funksiyalar1n parametr siyah1lar1n1n eyni olmas1na nazar yetirin.

|  |  |
| --- | --- |
| **Collection** | **DoubleList** |
| add | addAtHeader |
| destroy | destroyFromHead |
| detach | detachFromHead |

Tayin edilan digar klassik funksiyalar va onlar1n qiymatlari isa a§ag1dak1 kimidir:

|  |  |
| --- | --- |
| **Üzv funksiya** | **Qiymati** |
| isA() | doubleListClass |
| nameOf() | “DoubleList” |
| hashValue() | 0 |

#### HashTable

Collection sinfindan töranan HashTable onun daxilindaki obyektlara daha tez müraciat edilmasini tamin edir. Normal olaraq obyektlarin istar massiv daxilinda, istarsa da siyah1 strukturu daxilinda saxlan1lmas1ndan as1l1 olmayaraq axtard1g1m1z obyektin yerini bilmiriksa, obyekt kolleksiyas1nda axtar1lan obyekta müraciat edana qadar bütün obyektlara bir-bir baxmaq laz1m galir. Bu da kolleksiyan1n ba§lang1c1na

yax1n obyektlarin axtar1lmas1nda vaxt bax1m1ndan yax§1 natica vermasina baxmayaraq uzaq obyektlar üçün manfi natica verir. Buna göra da istanilan bir elemanta müraciat etmak üçün, bütün elementlari gözdan keçirmamak üçün malumatlar strukturu tayin edilmi§dir. HashTable bu malumatlar strukturundan yaln1z biridir. HashTable malumatlar strukturunun i§lama qaydas1n1 Object sinfi daxilinda hashValue funksiyas1n1 §arh edarkan göstarmi§dik.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

HashTable kolleksiyalar1 daxilindaki obyektlar hashValue funksiyalar1n1n verdiklari qiymatlara göra qrupla§d1r1l1r. Axtar1§ apararkan da yaln1z axtar1lan obyekt hashValue funksiyas1 ila müayyan edilan qrup daxilinda axtar1lacaqd1r.

HashTable kolleksiyas1n1n neça qrupdan ibarat olacag1 proqramç1 tarafindan bu kolleksiya tartib edilarkan müayyan edilir. Buna göra da HashTable sinfinin layihalandiricisi

HashTable(sizeType *grup\_sayi* = DEFAULT\_HASH\_TABLE\_SIZE);

§aklinda tayin edilmi§dir. 8gar

HashTable *Memurlar(50)*;

kimi bir *Memurlar* kolleksiyas1 tartib edilarsa, bu kolleksiya 50 qrupdan ibarat bir HashTable olacaqd1r. Yox agar

HashTable *Memurlar*;

kimi tayin edilarsa,

DEFAULT\_HASH\_TABLE\_SIZE

aktiv qiymatini istifada edacakdir ki, bu qiymat da 111- dir. Yani 111 müxtalif qrupu olan *Memurlar* ad1nda HashTable kolleksiyas1 yarad1lacaqd1r.

Layihalandirici xaricindaki funksiyalar Collection daxilinda tayin edilan funksiyalard1r va eyni manan1 da§1y1rlar.

|  |  |
| --- | --- |
| **Üzv funksiya** | **Qiymati** |
| isA() | hashTableClass |
| nameOf() | “HashTable” |
| hashValue() | 0 |

#### Btree

Btree, Binary-Tree (ikilik agac) üsulunun yadda§ siniflari ila tatbiq edilmasidir. ikilik agac üsulunda malumatlar sanki bir agac üzarinda kökdan budaqlara dogru yerla§dirilirmi§ kimi saxlan1l1r. Bu üsulda yadda§

daxilinda yerla§dirilacak malumatlar1n öz aralar1nda s1ralanmas1 laz1md1r. Malumatlar bir yarpaq kimidir va budaqlar üzarinda yerla§irlar. Budaqlarda yarpaq va ya qiymat budaqlar1 yerla§ir. Bu budaqlar üzarinda s1ralamaya göra ard1c1l (bir-birini taqib edan) malumatlar yerla§ir. Budaqlar da bir-birina göra ard1c1l malumat qruplar1ndan ibarat budaqlar §aklinda agac üzarinda yerla§irlar. Belalikla, malumat1 q1sa müddatda gözdan keçirmakla süratli axtarmaq olar. Malumat yadda§da olmasa da, bütün malumatlar gözdan keçirilmadan bu malumat1n yadda§da olub olmamas1n1 müayyanla§dirmak olar.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Btree sinfi Collection sinfindan töranmi§ sinifdir.

|  |  |
| --- | --- |
| **Üzv funksiya** | **Qiymati** |
| isA() | btreeClass |
| nameOf() | “Btree” |

Btree sinfinin tayin edilmi§ bir layihalandiricisi vard1r.

Btree(int *Yarpaq\_Sayi* = 3);

Bu layihalandiriciya parametr kimi normal olaraq bir budaq üzarinda olmas1 laz1m galan *Yarpaq\_Sayt* daxil edilir. Bu qiymat verilmadikda 3 olur. Budaq üzarindaki yarpaqlar1n say1n1n az olmas1 axtarma amaliyyat1n1 süratlandirir. Lakin agac üzarindaki budaqlar1n say1n1

art1r1r. Buna göra da malumatlar artd1qca axtarma müddati da art1r. Yarpaqlar1n say1 çox oldugu zaman isa malumatlar1n say1n1n az olmas1na baxmayaraq axtarma müddati uzana bilar. Buna göra da yarpaqlar1n say1n1 malumat çoxlugunu nazara alaraq müayyanla§dirmak faydal1d1r.

int order();

8vvalcadan yarad1lm1§ bir agac1n budaqlar1 üzarinda ola bilacak yarpaqlar1n say1n1n müayyanla§dirilmasi üçün istifada olunur. Geri qaytarma qiymati olaraq budaq üzarinda ola bilacak yarpaq say1n1 qaytar1r.

virtual void add(Object& *Obyekt*);

Bu funksiya parametr kimi verilan *Obyekt* obyektinin agac üzarinda yerla§dirilmasini tamin edir. Burada agac üzarindaki malumatlar1n s1ralana bilan obyekt olduqlar1n1 nazara alaraq *Obyekt* obyektinin Sortable sinfindan töranmi§ sinfin obyekti olmas1na diqqat etmak laz1md1r.

Object& operator[ ](long *Index*) const;

Agac daxilinda yerla§an obyektlar agac üzarinda kiçikdan böyüya dogru s1ralanm1§ olurlar. ilk s1ra

public: Integer(int = 0);

char\* nameOf() const; classType isA() const;

hashValueType hashValue() const; int isEqual(const Object&) const;

int isLessThan(const Object&) const; void printOn(ostream&) const;

};

Integer::Integer(int x)

{ \_integer = x; }

char\* Integer::nameOf() const

{ return "Integer"; }

classType Integer::isA() const

{ return integerClass; }

hashValueType Integer::hashValue() const

{ return \_integer; }

int Integer::isEqual(const Object& object) const

{ return \_integer == ((Integer&)object).\_integer; }

int Integer::isLessThan(const Object& object) const

{ return \_integer<((Integer&)object).\_integer; }

void Integer::printOn(ostream& stream) const

{ stream<<\_integer; }

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include <conio.h>

#include <btree.h>

#include <stdlib.h>

#include <stdio.h>

#include <ltime.h>

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

nömrasi 0 olamaqla istanilan s1radak1 obyektin öyranilmasi üçün [ ] operatorundan istifada etmak olar. Bunun üçün Btree sinfinin obyektlarinin har birini massiv olaraq dü§ünmak va istifada etmak mümkündür.

Btree Stock;

...

cout<<”en kicik element = “<<Stock[0]<<”dir\n”;

kimi. Yaln1z [ ] operatoru vasitasila har hans1 bir s1raya obyekt manimsadila bilmaz. Bu operator sadaca istanilan s1radak1 elementin öyranilmasini tamin edir.

long rank(const Object& *Obyekt*) const;

Btree agac1 daxilinda oldugu malum olan obyektin agac1n hans1 s1ras1nda oldugunu öyranmak üçün rank funksiyas1ndan istifada etmak olar.

Collection, Container va Object siniflari üçün tayin edilmi§ digar funksiyalar1 Btree daxilinda eyni maqsadla istifada etmak olar.

//BTREE.CPP

#include <sortable.h>

#define integerClass 300

class Integer : public Sortable

{ int \_integer;

Btree Stock(15); main()

{ clrscr(); int i;

char buffer[120];

for(i = 0; i < 2500; i++)

Stock.add(\* new Integer(random(3000)));

cout<<"\nStokta axtardiginiz nomreleri daxil edin\n" "bitirmek ucun SON yazin\n";

while(!cin.rdstate())

{ int X; cin>>X;

if(cin.rdstate()) break;

Integer Find(X);

cout<<"Axtarisa baslama saati : "<<Time()<<endl; cout<<Find<<" Stokda"<<(Stock.hasMember(Find) ?

" Movcuddur" : " Mevcud deyil")<<endl; cout<<"Axtarisin bitmesi saati : "<<Time()<<endl;

}

cin.clear(); cin>>ws>>buffer;

cout<<"Rank qiymetini oyrenmek ucun "

"nomreleri daxil edin\nbitirmek ucun SON yazin\n"; while(!cin.rdstate())

{ int X; cin>>X;

if(cin.rdstate()) break;

Integer Find(X);

cout<<"Rank("<<Find<<") = "<<Stock.rank(Find)<<endl;

}

cin.clear(); cin>>ws>>buffer;

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

cout<<"Oyrenmek istediyiniz melumatin" " sira nomresini daxil edin\n" "bitirmek ucun SON yazin\n";

while(!cin.rdstate())

{ int X; cin>>X;

if(cin.rdstate()) break;

cout<<'#'<<X<<"-->"<<Stock[X]<<endl;

}

cin.clear(); cin>>ws>>buffer;

return 0;

}

Proqram ç1x1§1

Stokta axtardiginiz nomreleri daxil edin bitirmek ucun SON yazin

50 200 700 3000 122 son

Axtarisa baslama saati : 8:57:25.10 pm 50 Stokda Movcuddur

Axtarisin bitmesi saati : 8:57:25.10 pm Axtarisa baslama saati : 8:57:25.10 pm 200 Stokda Mevcud deyil

Axtarisin bitmesi saati : 8:57:25.10 pm Axtarisa baslama saati : 8:57:25.10 pm 700 Stokda Mevcud deyil

Axtarisin bitmesi saati : 8:57:25.10 pm Axtarisa baslama saati : 8:57:25.10 pm 3000 Stokda Mevcud deyil

Axtarisin bitmesi saati : 8:57:25.10 pm Axtarisa baslama saati : 8:57:25.10 pm 122 Stokda Mevcuddur

Axtarisin bitmesi saati : 8:57:25.10 pm

Rank qiymetini oyrenmek ucun nomreleri daxil edin

|  |  |
| --- | --- |
| isA() | bagClass |
| nameOf() | “Bag” |

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

#### Bag

bitirmek ucun SON yazin 50 200 3 son

Rank(50) = 44

Rank(200) = 177

Rank(3) = 2

Oyrenmek istediyiniz melumatin sira nomresini daxil edin bitirmek ucun SON yazin

0 1 2 2499 2500 son

#0 --> 1

#1 --> 2

#2 --> 3

#2499 --> 2997

#2500 --> Error

Müayyan maqsadlara göra bir-birina barabar olan obyektlarin malumatlar strukturu daxilinda olmas1na icaza verila va ya verilmaya bilar. indiya qadar gözdan keçirdiyimiz siniflarin ham1s1 kimi Collection sinfindan töranmi§ Bag sinfi da buna icaza verir. Set va Dictionary siniflari isa icaza vermirlar. Bag sinfinin HashTable sinfina ox§ar layihalandiricisi vard1r. Eyni mantiqla i§layir. Sadaca aktiv qiymati farqlanir.

Bag(sizeType *qrup\_sayt* = DEFAULT\_ BAG\_SIZE);

DEFAULT\_BAG\_SIZE parametrinin standart tayin olunmu§ qiymati 29-dur.

|  |  |
| --- | --- |
| **Üzv funksiya** | **Qiymati** |

#### Set

Set sinfi Bag sinfindan töranmi§ bir sinif kimi ondan farqli olaraq bir obyektin Collection daxilinda yaln1z bir dafa olmas1na icaza verir. Ekvivalenti kolleksiya daxilinda olan obyekti kolleksiyaya qabul etmir. Bag ila eyni layihalandiriciya malikdir. Yaln1z aktiv qiymati farqlidir.

Set(sizeType *qrup\_sayt* = DEFAULT\_ SET\_SIZE);

DEFAULT\_SET\_SIZE parametrinin standart tayin olunmu§ qiymati 29-dur.

|  |  |
| --- | --- |
| **Üzv funksiya** | **Qiymati** |
| isA() | setClass |
| nameOf() | “Set” |

#### Dictionary

Dictionary (Lügat) axtar1§a göra malumat kolleksiyas1 daxilinda bir obyekt tap1b bu obyektdan daha çox malumat alda etma üsuludur. Masalan, bir

§axsin ad1n1 göstararak o §axsin soyad, dogum yeri,

dogum tarixi kimi digar malumatlar1n1n da öyranilmasidir. Bunun an klassik tatbiqi lügatda öz aksini tap1r. Bir söza uygun galan o sözün verdiyi manalar va ya digar dillardaki manalar1 tap1l1b müayyanla§dirilir.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Dictionary Set sinfindan töranan Association sinfindan töranmi§ obyektlarin daxilinda ola bilacak xüsusi bir kolleksiya növüdür.

Dictionary sinfinin üzv funksiyalar1 Collection sinfi daxilinda §arh etdiyimiz üzv funksiyalar1d1r. Eyni prototiplarinin olmas1na baxmayaraq bu funksiyalardak1 *obyekt* parametrlarina uygun obyekt olaraq Association sinfindan töranmi§ obyektlar istifada edila bilar.

|  |  |
| --- | --- |
| **Üzv funksiya** | **Qiymati** |
| isA() | dictionaryClass |
| nameOf() | “Dictionary” |

#### AbstractArray

AbstractArray xüsusila da bütün proqramla§d1rma dillarina daxil olduguna göra çox istifada edilan malumatlar strukturu tipli massivlardir. Container Class kitabxanas1 daxilinda ümumi maqsadli iki sinif tayin edilmi§dir. Bunlar ümumi maqsadli massivlar üçün Array va s1ralanm1§ obyektlar üçün SortedArray siniflaridir.

AbstractArray isa Collection sinfindan töranan va bu iki sinfa baza yaradan mücarrad bir sinifdir.

8vvalca massiv anlay1§1n1 ayd1nla§d1raq. Massiv eyni tipli elementlarin kolleksiyas1d1r. Elementlar massiv daxilinda ard1c1l yerla§ir va har bir elementin s1ra nömrasi olur ki, buna da elementin indeksi deyilir. Har hans1 bir elementa bu indeksla müraciat edilir. indekslar

1 va ya 0-dan ba§lay1r. Üst sarhad isa proqramç1 tarafindan tayin olunur va bu sarhaddi keçmak olmaz.

Container Class kitabxanas1 daxilinda tayin olunmu§ massiv siniflari üçün da ox§ar qaydalar qüvvadadir.

A§ag1dak1 farqlar vard1r:

* Massivlar daxilinda Object sinfindan töranmi§ bütün obyektlar ola bilar;
* Alt va üst sarhadlar proqramç1 tarafindan tayin edila bilar;
* Massivin elementlarinin say1 kifayat etmazsa, o geni§landirila bilar;
* Qiymati verilmayan elementlar NOOBJECT kimi

qabul edilir.

Massivlar üçün AbstractArray sinfi daxilinda tayin edilan funksiyalar a§ag1dak1lard1r:

int lowerBound() const;

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Bu funksiya massivin alt sarhaddini geri qaytarma qiymati kimi qaytar1r. Bu, massiv daxilinda olan ilk qiymatin s1ra nömrasidir.

int upperBound() const;

Massivin üst sarhaddini geri qaytarma qiymati kimi qaytar1r. Bu, massiv daxilinda olan son qiymatin s1ra nömrasidir.

sizeType arraySize() const;

Bu funksiya massiv daxilindaki elementlarin say1n1 müayyanla§dirir. getItemsInContainer funksiyas1ndan bu xüsusiyyati ila farqlanir. getItemsInContainer kolleksiya daxilinda olan elementlarin say1n1 müayyanla§dirir.

virtual void detach(int *indeks*, int *tip* = 0);

Bu funksiya *indeks* ila yeri verilan obyekti massiv daxilinda oldugu zaman massivdan xaric edir. *tip* parametrinin qiymati 0-dan farqlidirsa, obyekti massivdan xaric etmakla barabar onu yadda§dan da silir.

void destroy(int *indeks*);

Bu funksiya isa *indeks* ila yeri verilan obyekti massiv daxilinda oldugu zaman massivdan xaric edir va sonra bu obyekti yadda§dan silir.

virtual int isEqual(const Object& *Obyekt*) const;

Massivlar üçün yenidan tayin edilan bu funksiya digar siniflarda oldugu kimi müqayisa edilan har iki massivin eyni s1rada va bir-birina barabar olan obyektlarinin olmas1 §artini qoyur. Bundan ba§qa massivlarin alt va üst sarhadlari barabar olaml1d1r.

virtual void PrintOn(ostream& *Stream*) const;

Massiv daxilindaki obyektlarin ax1na yaz1lmas1 talab olunarsa, printOn funksiyas1 va ya “<<” operatoru istifada edilir. Bunlar1n istifada olunmas1 naticasinda massiv daxilindaki obyektlar ax1na yaz1l1r, qiymati verilmami§ elementlar isa “error” masaj1 ila aks olunur (bo§ elementlara Error sinfinin obyekti olan NOOBJECT manimsadilir). 8gar buna ehtiyac olmazsa, yani yaln1z haqiqi obyektlarin göstarilmasi talab olunarsa,

virtual printContainerOn(ostream& *Obyekt*) const;

üzv funksiyas1ndan istifada etmak olar.

#### Array

Array sinfi ümumi maqsadlar üçün istifada olunan bir sinifdir. AbstractArray sinfindan töranmi§dir. Array sinfinin obyekti olan massivlara Object sinfindan töranmi§ bütün siniflarin obyektlarini yerla§dirmak mümkündür. Yerla§dirilan har element yerla§diyi yerda qal1r.

Bu sinfin layihalandiricisi a§ag1dak1 kimi tayin edilm§dir:

Array(int *Ust*, int *Alt* = 0, sizeType *Delta* = 0);

Burada ilk parametr olan *Ust* massivin an böyük indeksini göstarir. Bu qiymat mütlaq verilmalidir. ikinci parametr olan *Alt* isa massivin an kiçik indeksini göstarir. Bu qiymat verilmazsa, avtomatik olaraq 0 qabul edilir. Üçüncü parametr olan *Delta* isa massiva add üzv funksiyas1 ila element yerla§dirildiyi zaman bo§ yer olmazsa, massivin üst sarhaddinin na qadar geni§landirilacayini göstarir. Bu qiymat da verilmazsa, 0 oldugu, yani bela bir halda har hans1 bir geni§lanmanin olmayacag1n1 göstarir.

Qeyd edildiyi kimi massiva obyekt alava etmak

üzv funksiyas1 istifada edilir. Bu funksiya *Obyekt* obyektini massiv daxilindaki ilk bo§ yera yerla§dirir. Bo§ yer olmazsa, massiv layihalandirma zaman1 müayyan edilan miqdarda geni§landirilir. Geni§lanma olmazsa, sahv ba§ verir. Massiva alava edilan obyektlarin massiv daxilnda hans1 s1rada yerla§acayi yena bu funksiya ila müayyan edilir. 8gar yerla§acayi yerin da göstarilmasi talab olunarsa,

void addAt(Object& *Obyekt*, int *Indeks*);

üzv funksiyas1ndan istifada etmak laz1md1r. Bu halda *Obyekt Indeks* ila verilan yera yerla§dirilir. Yerla§dirma amaliyyat1ndan avval yerla§dirilacak yerda ba§qa bir obyekt olarsa, bu obyekt massivdan ç1xar1l1r va agar silina bilarsa, silinir.

Massiv daxilinda bir obyekti findMember üzv funksiyas1 ila axtarmaq olar. 8gar yeri malumdursa,

Object& operator[ ](int *Indeks*) const;

operator funksiyas1 istifada edila bilar. Masalan, bir massivin 5-ci elementina müraciat etmak üçün

üçün

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

virtual void add(Object& *Obyekt*);

Object& *Obyekt* = mas[5];

va ya ekrana ç1xarmaq üçün

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

cout<<mas[5]<<endl;

satrini yazmaq olar.

Digar funksiyalar1n geri qaytarma qiymatlari isa a§ag1dak1lard1r:

|  |  |
| --- | --- |
| **Üzv funksiya** | **Qiymati** |
| isA() | arrayClass |
| nameOf() | “Array” |

#### SortedArray

SortedArray asasan massivdir. Lakin Array sinfindan farqli olaraq elementlarini kiçikdan böyüya dogru s1ralanm1§ §akilda saxlay1r. Buna göra da ancaq Sortable sinfindan töranan siniflarin obyektlarini tarkibinda saxlay1r. SortedArray massivlari daxilinda olan obyektlar s1ralanm1§ §akilda massivin avvalinda yerla§irlar. 8lava edilian har obyekt s1ralamadak1 uygun yerda yerla§dirilir. Bu amaliyyat massivin daxilindaki digar obyektlar sürü§dürülarak yerina yetirilir. Bir obyekt massivdan

<1xar1ld1qdan sonra amala galan bo§lugu doldurmaq üçün obyektlar bu dafa da aks istiqamatda sürü§dürülür.

Bu sinfin layihalandiricisi da Array sinfinin layihalandiricisi kimi tayin edilmi§dir.

SortedArray(int *Ust*, int *Alt* = 0, sizeType *Delta* = 0);

Parametrlar Array sinfinin layihalandiricisinin parametrlari ila eyni funksiyalara malikdirlar. *Ust* massivin an böyük, *Alt* isa an kiçik indeksi, *Delta* isa kifayat qadar yer olmad1g1 zaman geni§lanma miqdar1n1 göstarir.

Digar funksiyalar1n geri qaytarma qiymatlari

a§ag1dak1lard1r:

|  |  |
| --- | --- |
| **Üzv funksiya** | **Qiymati** |
| isA() | sortedArrayClass |
| nameOf() | “Array” |

Bu sinifda yeri balli olan bir obyekt üçün

const Sortable& operator[ ](int *Indeks*) const;

operator funksiyas1ndan istifada etmak olar.

Bir diskdaki aktiv qovluqda olan fayllar1n geni§lanmalarinin siyah1s1n1n görünmasina aid bir proqram tartib edak.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

//EXT.CPP

#include <clsdefs.h>

#include <sortable.h>

#include <sortarry.h>

#include <set.h>

#include <dir.h>

#include <dos.h>

#include <string.h>

#include <iomanip.h>

#include <conio.h>

class NameExt : public Sortable

{ char ext[4];

public: NameExt();

NameExt(const NameExt&); NameExt(char \*n); classType isA() const;

char\* nameOf() const; hashValueType hashValue() const; int isEqual(const Object&) const;

int isLessThan(const Object&) const; void printOn(ostream&) const;

};

NameExt::NameExt()

{ ext[0] ='\0'; }

NameExt::NameExt(const NameExt& O)

{ strcpy(ext,O.ext); }

NameExt::NameExt(char \*n)

{ strcpy(ext, n); }

classType NameExt::isA() const

{ return firstUserClass; }

char\* NameExt::nameOf() const

{ return "File Name Extention"; }

hashValueType NameExt::hashValue() const

{ return (ext[0] == '\0' || ext[0] == ' ') ? 0 : ext[0] - 64; }

int NameExt::isEqual(const Object& Test) const

{ return strcmp(ext, ((NameExt&)Test).ext) == 0; }

int NameExt::isLessThan(const Object& Test) const

{ return strcmp(ext, ((NameExt&)Test).ext) < 0; }

void NameExt::printOn(ostream& Stream) const

{ Stream<<'\"'<<setw(3)<<setiosflags(ios::left)<<ext<<'\"'; }

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* class Extension : public Set

{ int sta;

public: Extension(int);

void printHeader(ostream&) const; void printSeparator(ostream& ) const; void printTrailer(ostream&) const;

};

Extension::Extension(int \_sta) : Set(26)

{ sta = \_sta; }

void Extension::printHeader(ostream& Stream) const

{ Stream<<"Genislenme siyahisi"<<endl<<endl; }

void Extension::printSeparator(ostream& Stream) const

{ if(sta)

Stream<<'\t'; else Stream<<'\n';

}

void Extension::printTrailer(ostream& Stream) const

{ Stream<<endl; }

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Extension ExtList(1);

void scan(char \*s)

{ struct ffblk SF; int Ok;

Ok = findfirst(s, &SF, FA\_ARCH); while(Ok == 0)

{ char \*p = strchr(SF.ff\_name, '.');

NameExt\* N = new NameExt((p == NULL) ? " " : p + 1); ExtList.add(\*N);

Ok = findnext(&SF);

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* main(int c, char \*a[])

{ clrscr();

char match[MAXDIR];

if(c > 1)

{ strcpy(match, a[1]); if(strchr(match, '.') == NULL)

strcat(match, ".\*");

}

else strcpy(match, "\*.\*");

scan(match); if(ExtList.isEmpty())

cout<<"Verilen genislenmeli fayl yoxdur."; else cout<<ExtList<<"\nCemi "

<<ExtList.getItemsInContainer()

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

<<" eded genislenme var"; return 0;

}

Proqram ç1x1§1

Genislenme siyahisi

"BAT" "BAK" "CSM" "386" "DFM" "DSK"

"CFG"

"COM" "CPP" "DAT"

"DPR" "DOS" "DIF" "DLL" "ERR" "EXE" FIL"" "FON" "H " "HLP"

"ILD""ILC"

"OBJ"

"ILF"

"PIF"

"ILS" "INI" "ICO" "MAK" "OVL"

"SYM" "SWP" "SYS" "TDS" "TXT" "TC " "TAH" "TCH" "TFH" "TDH"

Cemi 40 eded genislenme var

### 8.6 Yenilayicilar (Iterators)

**Yenilayicilar** Container sinfindan töranmi§ siniflarin obyektlari üzarinda göstarici kimi i§layarak, bu obyektlara har hans1 bir zarar vermadan, yadda§

xüsusiyyatina malik olan obyektlar daxilindaki obyektlara bir-bir müraciat edilmasini tamin edirlar. Yenilayici siniflarinin har biri ContainerIterator sinfindan töranmi§dir.

yarad1ld1g1 üçün yenilayici ila i§imizi tamamlad1qdan sonra delete operatoru ila onu silmak laz1md1r.

HashTableIterator

BtreeIterator

ContainerIterator

ListIterator

DoubleListIterator

ArrayIterator

delete &I;

Yenilayicilar üçün tayin edilmi§ ortaq funksiyalar

a§ag1dak1lard1r:

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Müxtalif yadda§ tiplari üçün tayin olunmu§ yenilayicilar ümumi xüsusiyyatlara malikdirlar.

Yaln1z DoubleListIterator sinfinin alava bir xüsusiyyati vard1r.

Yenilayicinin layihalandirilmasi üçün har yenilayici bir layihalandiriciya sahib olmaqla barabar, yenilama amaliyyat1n1n yerina yetirilacayi yadda§ obyektinin

vitual ContainerIterator& initIterator() const;

kimi tayin edilmi§ üzv funksiyas1ndan istifada etmak maqsadauygundur. Masalan, S Stack strukturu üzarinda yenilama amaliyyat1n1 aparacaq I yenilayicisini

ContainerIterator& I = S.initIterator();

§aklinda tayin etmak olar. Lakin burada qeyd etmak laz1md1r ki, initIterator() üzv funksiyas1 new operatoru ila

virtual operator int();

Yenilayicini int tipli bir adada çevirmak laz1m galarsa, onun yadda§ sinfi üzarinda yenilayacayi ba§qa obyektin qal1b qalmayacag1 sual1 ortaya ç1x1r. Natica s1f1rd1rsa, yenilanacak obyekt qalm1r, s1f1rdan farqlidirsa, yenilanacak digar obyektlarin da oldugu balli olur.

virtual Object& operator++(); virtual Object& operator++(int);

Bu operator isa yenilayicinin göstardiyi obyekti geri qaytarma qiymati kimi qaytard1qdan sonra onun yadda§ daxilinda növbati obyekti göstarmasini tamin edir.

virtual operator Object&();

Bu operator isa yenilayicinin göstardiyi obyektin öyranilmasini tamin edir. Ba§qa bir amaliyyat1 yerina yetirmaz.

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

virtual void restart();

Bu funksiya yenilayicinin yadda§ daxilinda ilk obyekti göstarmasini tamin edir. Belalikla, yadda§ üçün yeni bir obyekt tayin edilmadan yadda§ daxilindaki obyektlar yenidan gözdan keçirila bilar.

#### 8.6.1 DoubleListIterator

DoubleListIterator DoubleList sinfi üzarinda yenilama amaliyyat1n1 yerina yetirmak üçün layihalandirilmi§dir. DoubleList sinfi malumatlar1 bir-biri ila ham iraliya, ham da geriya dogru alaqalandirdiyi üçün DoubleListIterator yenilayicisi ila uygun istiqamatlarda getmak mümkündür.

Bunun üçün ++ operatoruna ox§ar olaraq bu sinif üçün -- operatoru da tayin edilmi§dir.

virtual Object& operator--(); virtual Object& operator--(int);

Bu operatorlar yenilayicinin göstardiyi obyekti geri qaytararaq, onun avvalki obyekti göstarmasini tamin edirlar.

### 8.7 Misal

Ekran üzarinda müxtalif böcaklar vard1r. Bu böcaklarin davran1§lar1 müxtalifdir. Masalan, bazilari ekran kanarlar1na toxunduqlar1 zaman geri qay1daraq yollar1na davam edirlar. Bazilari isa bir kanardan ç1x1b digar kanardan girarak yollar1na davam edirlar. Bir böcak isa klaviatura düymalarinin s1x1lmas1 ila verilan amrlarla (ox düymalarina s1xaraq) harakat etdirilir. Oyunun maqsadi da bu böcaklari ekrandan tamizlamakdir.

//CCDEMO.CPP

#include <queue.h>

#include <conio.h>

#include <stdio.h>

#include <stdlib.h>

#define KB\_TOP

#define KB\_PGUP

#define KB\_LEFT

#define KB\_RIGHT

#define KB\_END

#define KB\_BOTTOM

#define KB\_PGDN

72

73

75

77

79

80

81

class Bocek;

//Bocek adli sinfin daha sonra teyin edileceyini bildirir

|  |  |
| --- | --- |
| #include <dos.h> |  |
| #define SCREENWIDTH | 79 |
| #define SCREENHEIGHT | 24 |
| #define WAITTIME | 150 |
| #define KB\_HOME | 71 |

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Queue Bocekler;

//Boceklerin siyahisi

Bocek\* Canavar;

//Bocek sinfinin yalniz gostericisi istifade oluna biler

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Bocek Sinfi \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*// class Bocek : public Object

{ protected:

int type; //Bocegin tipi

int x, y; //Bocegin yeri

int u, v; //Bocegin sur'etleri

public: Bocek();

Bocek(int, int, int, int, int); virtual ~Bocek();

virtual char\* nameOf() const

{ return "Bocek"; }

virtual classType isA() const

{ return 12000; }

virtual hashValueType hashValue() const

{ return type; }

virtual void printOn(ostream&) const; virtual int isEqual(const Object&) const;

virtual void toxunma(Bocek&);

//Parametr ile verilen bocege toxunmasn

//neticesinde cagrilacaq uzv funksiya

virtual void solSerhed(); virtual void sagSerhed();

virtual void ustSerhed(); virtual void altSerhed();

//Ekran serhedlerine toxundugu zaman

//cagirilacaq uzv funksiyalar

virtual void sicra();

//Hereket vaxti geldiyi zaman

//hereketi temin eden uzv funksiya

void oldun();

//Bir boceyin basqa birine toxunmasn zamani

//olmesi halinda cagirilacaq uzv funksiya

int olumu();

//Bir boceyin olu olub olmadigini anlamaq

//ucun istifade edilecek uzv funksiya

protected:

virtual void nezaret();

//Bocegin ekran serhedlerine catib catmadigina

//ve diger boceklere toxunub toxunmadigina

//nezaret edecek funksiya

virtual void sil();

//Bocegi ekrandan silecek funksiya

virtual void cek();

//Bocegi ekranda cekecek funksiya

private:

int oldx, oldy;

//Bocegin evvelki yeri

};

Bocek::Bocek()

{ type = '\*';

oldx = oldy = x = y = u = v = 1;

}

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

Bocek::Bocek(int \_t, int \_x, int \_y, int \_u, int \_v)

{ type = \_t; oldx = x = \_x; oldy = y = \_y; u = \_u;

v = \_v;

}

Bocek::~Bocek()

{ }

void Bocek::printOn(ostream& Stream) const

{ Stream<<nameOf()<<"Tip : "<<type; Stream<<" Yer : "<<x<<", "<<y; Stream<<" Addim : "<<u<<", "<<v;

}

int Bocek::isEqual(const Object& Test) const

{ return type == ((Bocek&)Test).type && x == ((Bocek&)Test).x &&

y == ((Bocek&)Test).y && u == ((Bocek&)Test).u && v == ((Bocek&)Test).v;

}

void Bocek:: toxunma(Bocek& B)

{ if(B == \*Canavar) oldun();

else

{ u = -u; v = -v;

B.u = -B.u;

B.v = -B.v;

if(type == B.type && Bocekler.getItemsInContainer() < 20) Bocekler.put(\*new Bocek(type, x \* 2, y \* 2, -u, -v));

}

}

void Bocek::solSerhed()

{ x = SCREENWIDTH; }

void Bocek::sagSerhed()

{ x = 1; }

void Bocek::ustSerhed()

{ y = SCREENHEIGHT; }

void Bocek::altSerhed()

{ y = 1; }

void Bocek::sicra()

{ sil(); x+= u; y+= v;

nezaret(); cek();

}

void Bocek::nezaret()

{ if(x <= 1) solSerhed();

else if(x >= SCREENWIDTH) sagSerhed();

if(y <= 1) ustSerhed();

else if(y >= SCREENHEIGHT) altSerhed();

ContainerIterator& Iter = Bocekler.initIterator(); while(int(Iter))

{ Bocek\* Item = (Bocek\*) & (Iter++); if(Item->x == x && Item->y == y)

toxunma(\*Item);

}

delete &Iter;

}

void Bocek::sil()

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

{ gotoxy(oldx, oldy); putchar(' ');

}

void Bocek::cek()

{ gotoxy(x, y); putchar(type); oldx = x;

oldy = y;

}

void Bocek::oldun()

{ u = v =0;

sil();

}

int Bocek::olumu()

{ return u == 0 && v == 0; }

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* KenarBocek Sinfi \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

//Bir kenardan cixib diger kenardan giren bocek tipi // class KenarBocek : public Bocek

{ public:

KenarBocek() : Bocek()

{ }

KenarBocek(int a, int b, int c, int d, int e)

: Bocek(a, b, c, d, e)

{ }

virtual void toxunma(Bocek&); virtual void solSerhed(); virtual void sagSerhed(); virtual void ustSerhed(); virtual void altSerhed();

};

void KenarBocek::carpdin(Bocek& B)

{ if(B == \*Canavar) oldun();

else if(Bocekler.getItemsInContainer() < 20) Bocekler.put(\*new KenarBocek(type, x + 1, y, -u, -v));

}

void KenarBocek::solSerhed()

{ u = 1; }

void KenarBocek::sagSerhed()

{ u = -1; }

void KenarBocek::ustSerhed()

{ v = 1; }

void KenarBocek::altSerhed()

{ v = -1; }

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* NezaretBocek Sinfi \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//Oyuncunun ox duymeleri ile hereket etdirdiyi bocek // class NezaretBocek : public Bocek

{ public:

NezaretBocek() : Bocek()

{ u = v = 1; }

NezaretBocek(int a, int b, int c)

: Bocek(a, b, c, 1, 1)

{ }

virtual void toxunma(Bocek&); virtual void solSerhed(); virtual void sagSerhed(); virtual void ustSerhed(); virtual void altSerhed();

virtual void sicra();

};

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

void NezaretBocek:: toxunma(Bocek& B)

{ if(B == \*Canavar) return;

B.oldun();

}

void NezaretBocek::solSerhed()

{ x = 1; }

void NezaretBocek::sagSerhed()

{ x = SCREENWIDTH; }

void NezaretBocek::ustSerhed()

{ y = 1; }

void NezaretBocek::altSerhed()

{ y = SCREENHEIGHT; }

void NezaretBocek::sicra()

{ if(kbhit())

{ int ch = getch();

if(ch == 27 || ch == 3) exit(1);

if(ch == 0)

{ ch = getch(); switch(ch)

{ case KB\_HOME : y--;

case KB\_LEFT : x--; break; case KB\_PGDN : y++;

case KB\_RIGHT : x++; break; case KB\_PGUP : x++;

case KB\_TOP : y--; break; case KB\_END : x--;

case KB\_BOTTOM : y++; break;

}

sil(); nezaret();

}

}

cek();

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Ana Proqram \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

//Proqrami icra eden, ardicil olaraq butun boceklere mesaj

//gonderen alt proqram

void Run()

{ Bocek\* Item; int i;

while(\*(Item = (Bocek\*)&Bocekler.get()) != NOOBJECT)

{ if(Item->olumu())

{ delete Item; continue;

}

Item->sicra(); Bocekler.put(\*Item);

for(i = 0; i < WAITTIME; i++)

Canavar->sicra(); if(Bocekler.getItemsInContainer() <= 1)

{ clrscr(); gotoxy(20, 12);

cprintf("\a\aOyun Bitdi"); gotoxy(20, 14);

cprintf("Her hansi bir duymeni sixin"); getch();

while(kbhit()) getch();

break;

}

}

}

//Proqramdan cixarken istifade olunan altproqram void Cix()

{ clrscr();

[**r**](http://www.abbyy.com/buy)

[**r**](http://www.abbyy.com/buy)

\_setcursortype(\_NORMALCURSOR);

}

//Proqramin baslamasi ucun qurmaq main()

{ int i;

//Boceklerin yaradilmasi for(i = 0; i < 6; i++)

Bocekler.put(\*new Bocek('&', i + 10, 15, -1, 1));

//KenarBoceklerin yaradilmasi for(i = 0; i < 6; i++)

Bocekler.put(\*new KenarBocek('\*', i + 30, i + 5, 1, 1));

//Oyuncunun nezaret edecegi bocegin yaradilmasi Canavar = new NezaretBocek('#', SCREENWIDTH / 2,

SCREENHEIGHT / 2);

Bocekler.put(\*Canavar);

//Ekran gorunusunun hazirlanmasi

clrscr();

\_setcursortype(\_NOCURSOR);

//Cixis funksiyasinin menimsedilmesi atexit(Cix);

//Oyunun baslanmasi Run();

//Oyundan normal cixis

return 0;

}