Υπερφόρτωση τελεστών

Operator Overloading

- Η υπερφόρτωση τελεστών μας δίνει τη δυνατότητα να επανακαθορίσουμε τη χρήση ενός τελεστή για ένα νέο τύπο δεδομένων
- Πχ **int** a=5, b=12

  a+b=17

  rectangle rec1(10,2), rec2(3,5)

  rec1 +rec2 =?

Operator Overloading

- Όταν υπερφορτώνεται ένας τελεστής, αποκτά σημασία μόνο για την κλάση για την οποία υπερφορτώνεται. Διατηρεί την αρχική του σημασία για τους υπόλοιπους τύπους δεδομένων στους οποίους εφαρμόζεται
- Ο ίδιος τελεστής μπορεί να υπερφορτωθεί πολλές φορές για διαφορετικές κλάσεις

Operator Overloading

- Δεν μπορούμε να υπερφορτώσουμε όλους τους τελεστές της C++
- Δεν μπορούμε να δημιουργήσουμε νέους τελεστές. Πρέπει να χρησιμοποιήσουμε τους υπάρχοντες
- Δεν μπορούμε να αλλάξουμε την προτεραιότητα των τελεστών
- Στον παρακάτω πίνακα αναφέρονται ποιοι τελεστές δεν υπερφορτώνονται

Operator Overloading

Τελεστής	Σύμβολο
Τελεστής συνθήκης	?:
Τελεστής πρόσβασης στα μέλη δομών και κλάσεων	.
Τελεστής απόδοσης μεγέθους	sizeof
Τελεστής επίλυσης εμβέλειας	::
Τελεστής αναφοράς σε μέλη κλάσης	.*

Υπερφόρτωση τελεστών με χρήση μεθόδων της κλάσης

- Είναι ο συνηθέστερος τρόπος υπερφόρτωσης τελεστών

τύπος *κλάση*:: operator <τελεστής> (ορίσματα)

{

Σώμα της μεθόδου, στο οποίο καθορίζεται η συμπεριφορά του τελεστή σε αυτή τη κλάση

}

- **Τύπος**: είναι ο τύπος δεδομένων που επιστρέφει η συνάρτηση. Συνήθως ο τύπος αυτός είναι ίδιος με τον τύπο της κλάσης για την οποία υπερφορτώνεται ο τελεστής
- **Κλάση**: είναι η κλάση για την οποία υπερφορτώνεται ο τελεστής
- **operator**: Δεσμευμένη λέξη της C++, προηγείται του συμβόλου του τελεστή
- **<τελεστής>** Το σύμβολο του τελεστή που υπερφορτώνεται
- **Ορίσματα**: Κανένα, ένα ή περισσότερα ορίσματα ανάλογα με τον τελεστή που υπερφορτώνεται

- Οι τελεστές χωρίζονται στους διμελείς και στους μονομελείς. Οι διμελείς είναι αυτοί που έχουν δυο μέλη πχ a/b, αριστερό και δεξί, και οι μονομελής ένα μέλος πχ ++

Υπερφόρτωση διμελών τελεστών

- Η μέθοδος υπερφόρτωσης εφαρμόζεται αυτόματα όταν χρησιμοποιούμε έναν τελεστή με κάποιο αντικείμενο της κλάσης για την οποία τον υπερφορτώσαμε
- Το αντικείμενο που καλεί τη μέθοδο υπερφόρτωσης ενός διμελούς τελεστή είναι το αριστερό μέλος του τελεστή

- Πριν υπερφορτώσουμε έναν τελεστή πρέπει καταρχήν να αποφασίσουμε τι δουλειά θα κάνει. Με τη παρακάτω μέθοδο υπερφορτώνεται ο τελεστής της πρόσθεσης (+) για την κλάση rectangle. Ο τελεστής υπερφορτώνεται με τέτοιο τρόπο ώστε όταν προστίθενται δυο αντικείμενα της κλάσης να προκύπτει ένα νέο αντικείμενο με διαστάσεις ίσες με το άθροισμα των διαστάσεων των δυο μελών του

```
rectangle rectangle :: operator+ (rectangle op2)
  {
    rectangle tt;
    tt.plevra_a=plevra_a+op2.plevra_a;
    tt.plevra_b=plevra_b+op2.plevra_b;
    return tt;
   }
```

Στη παράσταση **rec1 + rec2** η μέθοδος υπερφόρτωσης έχει πρόσβαση στο αριστερό μέλος **rec1** απευθείας και στο δεξί μέλος **rec2** μέσω της παραμέτρου **op2**. Οι μεταβλητές **plevra_a** και **plevra_b** αναφέρονται στο αριστερό μέλος της πράξης, δηλαδή στο αντικείμενο **rec1**

- Η μέθοδος υπερφόρτωσης τελεστή πρέπει να δηλωθεί μέσα στην κλάση όπως όλες οι υπόλοιπες μέθοδοι της

```
class rectangle
{
   float plevra_a, plevra_b;
   public:
      string xroma;
      rectangle (float a, float b);
      void info();
      rectangle operator+ (rectangle op2);
 }
```

- Το τι θα κάνει ένας υπερφορτωμένος τελεστής για μια κλάση αλλά και τι είδους τιμή θα επιστρέψει εξαρτάται από τον προγραμματιστή. Ο κώδικας της μεθόδου υπερφόρτωσης του συγκεκριμένου τελεστή καθορίζει τη λειτουργία του

```
rectangle rectangle :: operator+ (rectangle op2)
 {
    return tt;
  }

rectangle rec1(10,2), rec2(5,7), rec3;
cout << rec1 + rec2;
rec3 = rec1 + rec2;
```

55
Λάθος πρόταση

Υπερφόρτωση τελεστών σύγκρισης

- Οι τελεστές σύγκρισης έχουν την ιδιαιτερότητα ότι το αποτέλεσμα της σύγκρισης είναι τύπου bool οπότε η μέθοδος υπερφόρτωσης ενός συγκριτικού τελεστή πρέπει να επιστρέφει τιμή αυτού του τύπου
- Έστω ότι θέλουμε να υπερφορτώσουμε τον τελεστή > για τη κλάση rectangle ώστε να συγκρίνουμε το εμβαδό δυο αντικειμένων αυτής της κλάσης

```
bool rectangle :: operator> (rectangle op2)
 {
    if (emvado() >op2.emvado())
      return true;
    else
       return false;
   }

 rectangle rec1(10,2), rec2(5,7);
If (rec1>rec2)
  cout << “rec1 > rec2”;
else
 cout << “rec2>rec1”<<endl;
```

Υπερφόρτωση μονομελών τελεστών

- Για έναν μονομελή τελεστή η μέθοδος υπερφόρτωσης του καλείται από το μοναδικό μέλος στο οποίο εφαρμόζεται. Η μέθοδος αυτή έχει άμεση πρόσβαση στα μέλη του αντικειμένου στο οποίο εφαρμόζεται
- Η υπερφόρτωση των τελεστών ++ και -- έχει μια ιδιαιτερότητα γιατί μπορεί να είναι προθεματικοί ή επιθηματικοί του αντικειμένου που τους καλεί

- Έστω ότι θέλουμε με τις παραστάσεις ++rec1 και rec1++ να αυξήσουμε τις διαστάσεις του αντικειμένου κατά 1
- Θα πρέπει να έχουμε 2 εκδόσεις της μεθόδου υπερφόρτωσης του τελεστή ++
- Αυτό είναι μια περίπτωση υπερφόρτωσης της μεθόδου υπερφόρτωσης

- Στον ορισμό της υπερφορτωμένης μεθόδου του προθεματικού ++ η μέθοδος δεν διαθέτει καμία παράμετρο

```
rectangle rectangle :: operator++ ()
  {
      plevra_a++;
      plevra_b++;
      return *this;
    }
```

- Στον ορισμό της υπερφορτωμένης μεθόδου του επιθηματικού ++ υπάρχει μια παράμετρος τύπου int η οποία δεν χρησιμοποιείται οπότε δεν αναφέρεται το όνομα της

```
rectangle rectangle :: operator++ (int)
  {
      plevra_a++;
      plevra_b++;
      return *this;  //ο δείκτης this επιστρέψει το ίδιο αντικείμενο με νέες διαστάσεις
    }
```

- Και στις δυο περιπτώσεις ο δείκτης this δείχνει στο αντικείμενο που κάλεσε την μέθοδο υπερφόρτωσης τελεστή
- Η ίδια τεχνική εφαρμόζεται και στην περίπτωση του μονομελούς τελεστή --

Υπερφόρτωση του τελεστή ανάθεσης =

- Ο τελεστής ανάθεσης χρησιμοποιείται για την πιστή αντιγραφή ενός αντικειμένου σε ένα άλλο, η οποία γίνεται bit προς bit και αφορά φυσικά αντικείμενα της ίδιας κλάσης
- Τις περισσότερες φορές αυτό είναι αρκετό, αλλά υπάρχουν περιπτώσεις όπως για παράδειγμα όταν τα αντικείμενα μιας κλάσης χρησιμοποιούν δυναμικά κατανεμημένη μνήμη, και μπορεί να δημιουργηθούν προβλήματα
- Σε αυτές τις περιπτώσεις υπερφορτώνουμε τον τελεστή =

Έστω ότι θέλουμε να υπερφορτώσουμε τον τελεστή = ώστε όταν αναθέτουμε ένα αντικείμενο rectangle σε ένα άλλο, να αντιγράφονται οι διαστάσεις αλλά όχι το χρώμα

```
rectangle rectangle :: operator= (rectangle op2)
  {
    plevra_a=op2.plevra_a;
    plevra_b=op2.plevra_b;
     return *this;
    }
```

Υπερφόρτωση του τελεστή κλήσης συνάρτησης ()

- Η υπερφόρτωση αυτού του τελεστή δεν αλλάζει καθόλου τον τρόπο κλήσης των συναρτήσεων αλλά μας δίνει τη δυνατότητα να αναφερόμαστε στα αντικείμενα μιας κλάσης σε μορφή συναρτήσεων ακολουθούμενα από παρενθέσεις μέσα στις οποίες υπάρχουν ορίσματα πχ rec(4,6). Σε αυτή τη περίπτωση τα ορίσματα που ακολουθούν το αντικείμενο μεταβιβάζονται στη μέθοδο υπερφόρτωσης του τελεστή ()

```
rectangle rectangle :: operator() (float a, float b)
  {
    plevra_a=a;
    plevra_b=b;
     return *this;
    }
```

Παράδειγμα 1

- Το ολοκληρωμένο πρόγραμμα που ακολουθεί υπερφορτώνει διάφορους τελεστές για τη κλάση rectangle

```
#include <iostream>
#include <iomanip>
#include <string>
#include <cmath>
using namespace std;
class rectangle
{
    float plevra_a;
    float plevra_b;
public:
    string xroma;
    rectangle(float a, float b);
    rectangle();
    float emvado();
    void info();
    rectangle operator+(rectangle op2);
    float operator-(rectangle op2);
    rectangle operator*(float scale);
    bool operator==(rectangle op2);
    bool operator==(float op2);
    rectangle operator--();
    rectangle operator++();
    rectangle operator()(float a, float b);
    rectangle operator()(float a);

};
```

Δηλώσεις των μεθόδων υπερφόρτωσης τελεστών

Ο συγκριτικός τελεστής == είναι υπερφορτωμένος 2 φορές, για διαφορετικούς τύπους δεξιού μέλους

Ο τελεστής () είναι υπερφορτωμένος 2 φορές για διαφορετικά πλήθη ορισμάτων

```
float rectangle::emvado()
{
    return plevra_a * plevra_b;
}
void rectangle::info()
{
    cout    <<setprecision(1)<<fixed<< plevra_a << "x" << plevra_b
            << " Εμβαδό:" << emvado() << endl;
}
rectangle::rectangle(float a, float b)
{
    plevra_a = a;
    plevra_b = b;
    xroma = "";
}
rectangle::rectangle()
{
    plevra_a = 0;
    plevra_b = 0;
    xroma = "";
}
rectangle rectangle::operator+(rectangle op2)
{
    rectangle tt;
    tt.plevra_a=plevra_a+op2.plevra_a;
    tt.plevra_b=plevra_b+op2.plevra_b;
    return tt;
}
```

Η μέθοδος υπερφόρτωσης του τελεστή + επιστρέφει ως τιμή ένα αντικείμενο rectangle με πλευρές το άθροισμα των πλευρών των δυο μελών του

```
float rectangle::operator-(rectangle op2)
{
    return emvado()-op2.emvado();
}
rectangle rectangle::operator*(float scale)
{
    plevra_a=plevra_a*scale;
    plevra_b=plevra_b*scale;
    return *this;
}
bool rectangle::operator==(rectangle op2)
{
    if (plevra_a==op2.plevra_a && plevra_b==op2.plevra_b ||
        plevra_a==op2.plevra_b && plevra_b==op2.plevra_a)
        return true;
    else
        return false;
}
bool rectangle::operator==(float op2)
{
    if (emvado()==op2)
        return true;
    else
        return false;
}
```

Η μέθοδος υπερφόρτωσης του τελεστή – επιστρέφει ως τιμή τη διαφορά των εμβαδών των δυο μελών του

Η μέθοδος υπερφόρτωσης του τελεστή * επιστρέφει ως τιμή το ίδιο το αντικείμενο με νέες διαστάσεις βασισμένες στην κλίμακα scale της παραμέτρου της

Η μέθοδος υπερφόρτωσης του τελεστή == επιστρέφει τιμή true στη περίπτωση που τα δυο μέλη της κλάσης rectangle έχουν τις ίδιες διαστάσεις ανεξάρτητα από τη σειρά τους

Η δεύτερη μέθοδος υπερφόρτωσης του τελεστή == επιστρέφει τιμή true στην περίπτωση που το αριστερό μέλος έχει εμβαδό ίσο με τη τιμή του δεξιού μέλους

```
rectangle rectangle::operator--()
{
    float neo_emv,scale;
    scale=plevra_a/plevra_b;
    neo_emv=emvado()-emvado()*10/100;
    plevra_b=sqrt(neo_emv/scale);
    plevra_a=plevra_b*scale;
    return *this;
}
```

Η μέθοδος υπερφόρτωσης του προθεματικού τελεστή – μειώνει το εμβαδό ενός αντικειμένου rectangle κατά 10%. Επιστρέφει ως τιμή το ίδιο το αντικείμενο. Για να γίνει η μείωση αλλάζουν οι διαστάσεις αλλά οι αναλογίες διατηρούνται

```
rectangle rectangle::operator++()
{
    float neo_emv,scale;
    scale=plevra_a/plevra_b;
    neo_emv=emvado()+emvado()*10/100;
    plevra_b=sqrt(neo_emv/scale);
    plevra_a=plevra_b*scale;
    return *this;
}
```

Η μέθοδος υπερφόρτωσης του προθεματικού τελεστή ++ αυξάνει το εμβαδό ενός αντικειμένου rectangle κατά 10%. Επιστρέφει ως τιμή το ίδιος το αντικείμενο

```
rectangle rectangle::operator()(float a, float b)
{
    plevra_a=a;
    plevra_b=b;
    return *this;
}
```

Η μέθοδος υπερφόρτωσης του τελεστή () με δυο παραμέτρους καταχωρίζει τις τιμές τους στις διαστάσεις του αντικειμένου. Επιστρέφει ως τιμή το ίδιο το αντικείμενο

```
rectangle rectangle::operator()(float a)
{
    plevra_a=a;
    plevra_b=a;
    return *this;
}

int main()
{
    rectangle rec1(100,20), rec2(50,70), rec3;
    cout <<"rec1=";
    rec1.info();
    if (rec1==2000)
        cout << "Ναι έχει εμβαδό 2000"<< endl;
    else
        cout << "Οχι δεν έχει εμβαδό 2000"<< endl;
    ++rec1;
    cout <<"rec1=";
    rec1.info();
    rec3 = ++rec2;
    cout <<"rec3=";
    rec3.info();
    cout <<"rec2=";
    rec2.info();
    --rec2;
    cout <<"rec2=";
```

Η δεύτερη μέθοδος υπερφόρτωσης του τελεστή () με μια παράμετρο καταχωρίζει τη τιμή της και στις δυο διαστάσεις του αντικειμένου. Επιστρέφει ως τιμή το ίδιο το αντικείμενο

Ελέγχει αν το αντικείμενο rec1 έχει εμβαδό ίσο με 2000. Καλείται η δεύτερη μέθοδος υπερφόρτωσης του τελεστή ==

Αυξάνει το εμβαδό του rec1 κατά 10%. Αυτό έχει αποτέλεσμα την κατάλληλη μεταβολή των διαστάσεων του

Αυξάνει το εμβαδό του rec2 κατά 10%. Καταχωρίζει τις νέες διαστάσεις του rec2 στο rec3

Μειώνει το εμβαδό του rec2 κατά 10%.

```
    rec2.info();
    rec2(10);
    cout <<"rec2=";
    rec2.info();
    rec2(25,67);
    cout <<"rec2=";
    rec2.info();
    rec3(67,25);
    if (rec2==rec3)
        cout << "Ναι έχουν ίδιες διαστάσεις"<< endl;
    else
        cout << "Οχι έχουν διαφορετικές διαστάσεις"<< endl;
    rec2*10;
    cout <<"rec2=";
    rec2.info();
    return 0;
}
```

Θέτει ως διαστάσεις του rec2 10x10. Καλείται η δεύτερη μέθοδος υπερφόρτωσης του τελεστή ()

Θέτει ως διαστάσεις του rec2 25x67. Καλείται η πρώτη μέθοδος υπερφόρτωσης του τελεστή ()

Ελέγχει αν τα αντικείμενα rec1 και rec2 έχουν ίδιες διαστάσεις. Καλείται η πρώτη μέθοδος υπερφόρτωσης του τελεστή ==. Μετά το rec2 αποκτά δεκαπλάσιες διαστάσεις. Καλείται η μέθοδος υπερφόρτωσης του τελεστή *

```
rec1=100.0x20.0 emvado:2000.0
Nai exei emvado 2000
rec1=104.9x21.0 emvado:2200.0
rec3=52.4x73.4 emvado:3850.0
rec2=52.4x73.4 emvado:3850.0
rec2=49.7x69.6 emvado:3465.0
rec2=10.0x10.0 emvado:100.0
rec2=25.0x67.0 emvado:1675.0
Nai exoun tis idies diastaseis
rec2=250.0x670.0 emvado:167500.0
```

Υπερφόρτωση του τελεστή πίνακα []

- Ο τελεστής [] χρησιμοποιείται για την προσπέλαση των στοιχείων ενός πίνακα και μπορεί να υπερφορτωθεί για οποιαδήποτε κλάση ώστε να παρέχει έναν παρόμοιο τρόπο προσπέλασης. Η μέθοδος υπερφόρτωσης του τελεστή πίνακα δέχεται μόνο μια παράμετρο
- Στο πρόγραμμα που ακολουθεί υπερφορτώνεται ο τελεστής [] για τη κλάση student

```cpp
#include <iostream>
#include <string>
#include <cstdlib>
using namespace std;
class student
{
    float bathmoi[5];
public:
    string onoma;
    string eponymo;
    student(string o, string e);
    void display();
    string operator[](int pos);
};
student::student(string o, string e)
{
    int i;
    onoma=o;
    eponymo=e;
    for (i=0;i<5;i++) bathmoi[i]=rand()%11;
}
```

Δήλωση της μεθόδου υπερφόρτωσης του τελεστή []

Η μέθοδος δόμησης της κλάσης καταχωρίζει την τιμή της πρώτης παραμέτρου στη μεταβλητή onoma και την τιμή της δεύτερης παραμέτρου στη μεταβλητή eponymo. Επίσης καταχωρίζονται τυχαίες τιμές στον πίνακα bathmoi

```
void student::display()
{
    int i;
    cout<<"Βαθμοί:";
    for (i=0;i<5;i++) cout << bathmoi[i]<<" ";
    cout<<endl;
}
string student::operator[](int pos)
{
    if (pos==1)
        return onoma;
    else if (pos==2)
        return eponymo;
    else
        return "";
}

int main()
{
    student st1("Τιμολέων","Αραχτόπουλος");
    st1.display();
    cout << st1[2];
    return 0;
}
```

Η μέθοδος display() εμφανίζει τους βαθμούς

Η μέθοδος υπερφόρτωσης του τελεστή [] επιστρέφει το όνομα ή το επώνυμο ενός αντικειμένου κλάσης student ανάλογα με τη τιμή της παραμέτρου

Βαθμοί: 8 9 9 1 7
Αραχτόπουλος

- Συνήθως ο τελεστής [] υπερφορτώνεται με τρόπο που να επιτρέπει την προσπέλαση κάποιου μέλους πίνακα μιας κλάσης. Με αυτόν τον τρόπο διατηρείται επίσης η προεπιλεγμένη σημασία του. Στο επόμενο πρόγραμμα που αποτελεί παραλλαγή του προηγούμενου, ο τελεστής [] έχει υπερφορτωθεί έτσι ώστε να επιστρέφει ως τιμή τον αντίστοιχο βαθμό ανάλογα με την τιμή της παραμέτρου της

```
#include <iostream>
#include <string>
#include <cstdlib>
using namespace std;
class student
{
    float bathmoi[5];
public:
    string onoma;
    string eponymo;
    student(string o, string e);
    void display();
    float operator[](int pos);
};
student::student(string o, string e)
{
    int i;
    onoma=o;
    eponymo=e;
    for (i=0;i<5;i++) bathmoi[i]=rand()%11;
}
void student::display()
{
    int i;
    cout <<onoma<<" "<<eponymo<<endl;
}
```

Δήλωση της μεθόδου υπερφόρτωσης του τελεστή []

Η μέθοδος display() εμφανίζει όνομα και επίθετο

```
float student::operator[](int pos)
{
    return bathmoi[pos-1];
}
int main()
{
    student st1("Τιμολέων","Αραχτόπουλος");
    int i;
    st1.display();
    for (i=1;i<=5;i++) cout << st1[i]<<" ";
    return 0;
}
```

Η μέθοδος υπερφόρτωσης του τελεστή [] επιστρέφει έναν από τους βαθμούς του φοιτητή ανάλογα με τη τιμή της παραμέτρου της

```
Τιμολέων Αραχτόπουλος
8 9 9 1 7
```

- Με αυτό τον τρόπο ο κώδικας του προγράμματος έχει πρόσβαση στους επιμέρους βαθμούς του αντικειμένου st1 οι οποίοι βρίσκονται αποθηκευμένοι στο ιδιωτικό μέλος της κλάσης, τον πίνακα bathmoi
- Μία τέτοια πρόταση, st1[3]=10; Ως προσπάθεια καταχώρησης του βαθμού 10 στο 3^{ο} μάθημα θα ήταν λάθος διότι η μέθοδος υπερφόρτωσης του τελεστή [] επιστρέφει μια τιμή και δεν μπορεί να βρίσκεται στα αριστερά του τελεστή ανάθεσης =. Αυτό θα μπορούσε να γίνει μόνο στην περίπτωση όπου η μέθοδος επέστρεφε αναφορά

Μέθοδοι που επιστρέφουν αναφορά

- Μια συνάρτηση στη C++ μπορεί να επιστρέψει ως τιμή μια αναφορά και το ίδιο ισχύει και για μια μέθοδο. Το πρόγραμμα που ακολουθεί είναι μια ακόμα μικρή παραλλαγή του προηγούμενου παραδείγματος. Τώρα η μέθοδος υπερφόρτωσης του τελεστή [] επιστρέφει μια αναφορά στην αντίστοιχη θέση του πίνακα των βαθμών

```
#include <iostream>
#include <string>
#include <cstdlib>
using namespace std;
class student
{
    float bathmoi[5];
public:
    string onoma;
    string eponymo;
    student(string o, string e);
    void display();
    float &operator[](int pos);
};
student::student(string o, string e)
{
    int i;
    onoma=o;
    eponymo=e;
    for (i=0;i<5;i++) bathmoi[i]=rand()%11;
}
```

Δήλωση της μεθόδου υπερφόρτωσης του τελεστή []. Το πρόθεμα & υποδηλώνει ότι η μέθοδος επιστέφει αναφορά

```
void student::display()
{
    int i;
    cout <<onoma<<" "<<eponymo<<endl;
    cout<<"Βαθμοί:";
    for (i=0;i<5;i++) cout << bathmoi[i]<<" ";
    cout<<endl;
}
float &student::operator[](int pos)
{
    return bathmoi[pos-1];
}
int main()
{
    student st1("Τιμολέων","Αραχτόπουλος");
    st1.display();
    cout << st1[3]<<endl;
    st1[3]=10;
    st1[1]=3;
    st1[5]=8.5;
    st1.display();
    return 0;
}
```

Η μέθοδος υπερφόρτωσης του τελεστή [] επιστρέφει μια αναφορά στη θέση του πίνακα bathmoi ανάλογα με την τιμή της παραμέτρου της

Τιμολέων Αραχτόπουλος
Βαθμοί: 8 9 9 1 7
9
Τιμολέων Αραχτόπουλος
Βαθμοί : 3 9 10 1 8.5

- Από τη στιγμή που η μέθοδος υπερφόρτωσης του τελεστή [] επιστρέφει αναφορά, οι παραστάσεις της μορφής st1[i] είναι παραστάσεις αριστερής τιμής και επιτρέπεται να βρίσκονται αριστερά του τελεστή ανάθεσης =. Οι παρακάτω προτάσεις αλλάζουν τώρα τους βαθμούς του φοιτητή st1
- st1[3]=10;
- st1[1]=8.5;

Υπερφόρτωση τελεστών με τη χρήση συναρτήσεων που δεν είναι μέλη κλάσης

- Η υπερφόρτωση τελεστών μπορεί να υλοποιηθεί και με χρήση συναρτήσεων που δεν είναι μέλη μιας κλάσης. Συνήθως οι συναρτήσεις αυτές είναι φίλιες συναρτήσεις της κλάσης για την οποία υπερφορτώνεται ο τελεστής.
- Οι συναρτήσεις που δεν είναι μέλη μιας κλάσης δεν επενεργούν στα αντικείμενα της κλάσης και επομένως δεν έχουν άμεση πρόσβαση σε κανένα από τα μέλη του τελεστή που υπερφορτώνουν. Στις συναρτήσεις αυτές πρέπει να μεταβιβάσουμε και τα δυο μέλη του τελεστή με ρητό τρόπο, δηλαδή μέσω παραμέτρων της.

- Σε προηγούμενο παράδειγμα υπερφορτώσαμε τον τελεστή σύγκρισης > για την κλάση rectangle ο οποίος ορίστηκε να συγκρίνει το εμβαδό των δυο αντικειμένων αυτής της κλάσης
- Αν όμως θέλαμε να συγκρίνουμε το εμβαδό με μια συγκεκριμένη τιμή (rec1>50) δεν θα μπορούσε να κληθεί η μέθοδος υπερφόρτωσης έτσι όπως έχει δηλωθεί:

bool rectangle :: operator> (rectangle op2);

- Η μέθοδος υπερφόρτωσης του τελεστή > περιμένει ως δεύτερο μέλος του τελεστή σύγκρισης ένα αντικείμενο τύπου rectangle για να μεταβιβαστεί στην παράμετρο (op2) της μεθόδου
- Η λύση στο παραπάνω πρόβλημα βρίσκεται στη χρήση συναρτήσεων υπερφόρτωσης τελεστή στιες οποίες μεταβιβάζονται ρητά και τα δύο μέλη του τελεστή. Οι συναρτήσεις αυτές μπορεί να είναι ή να μην είναι φίλιες συναρτήσεις κάποιας κλάσης. Μια τέτοια συνάρτηση πρέπει να είναι φίλια μιας κλάσης μόνο στην περίπτωση που χρειάζεται να έχει πρόσβαση στα ιδιωτικά μέλη των αντικειμένων αυτής της κλάσης

- Στον παρακάτω κώδικα φαίνονται οι συναρτήσεις υπερφόρτωσης του τελεστή > οι οποίες καλύπτουν τις δυο περιπτώσεις που προαναφέρθηκαν

```
bool operator > (float value, rectangle op2)
{
  if (value>op2.embado())
    return true;
  else
    return false;
}
```

Καλείται σε περίπτωση όπου το πρώτο μέλος της σύγκρισης είναι συγκεκριμένη τιμή πχ 50>rec1

```
bool operator > (rectangle op1, float value)
{
  if (op1.embado()>value)
    return true;
  else
    return false;
}
```

Καλείται σε περίπτωση όπου το δεύτερο μέλος της σύγκρισης είναι συγκεκριμένη τιμή πχ rec1>50

Υπερφόρτωση μεθόδων και συναρτήσεων υπερφόρτωσης τελεστών

- Το πρόγραμμα που ακολουθεί χρησιμοποιεί πολλές περιπτώσεις υπερφόρτωσης μεθόδων και συναρτήσεων υπερφόρτωσης τελεστών

```
#include <iostream>
#include <string>
using namespace std;

class rectangle;

class circle
{
    float aktina;
public:
    string xroma;
    circle();
    circle(float r);
    float emvado();
    bool operator>(circle op2);
    bool operator>(float op2);
    friend bool operator==(circle op1,  rectangle op2);
    friend bool operator==(rectangle op1, circle   op2);
};

circle::circle()
{
    aktina=0;
    xroma="";
}
```

Προκαταβολική δήλωση της κλάσης rectangle

Δυο υπερφορτωμένες εκδόσεις της μεθόδου υπερφόρτωσης του τελεστή > για τη κλάση circle

Οι δυο υπερφορτωμένες εκδόσεις της συνάρτησης υπερφόρτωσης του τελεστή == δηλώνοντας ως φίλιες συναρτήσεις της κλάσης circle

```
circle::circle(float r)
{
    aktina=r;
    xroma="";
}
float circle::emvado()
{
    return aktina*aktina*3.14;
}
bool circle::operator>(circle op2)
{
    if (emvado()>op2.emvado())
        return true;
    else
        return false;
}
bool circle::operator>(float op2)
{
    if (emvado()>op2)
        return true;
    else
        return false;
}
```

Πρώτη έκδοση της μεθόδου υπερφόρτωσης του τελεστή >. Χρησιμοποιείται στην περίπτωση σύγκρισης αντικειμένων κλάσης circle

Δεύτερη έκδοση της μεθόδου υπερφόρτωσης του τελεστή >. Χρησιμοποιείται στην περίπτωση σύγκρισης ενός αντικειμένου της κλάσης circle με κάποια τιμή

```
class rectangle
{
    float plevra_a;
    float plevra_b;
public:
    string xroma;
    rectangle(float a, float b);
    rectangle();
    float emvado();
    bool operator>(rectangle op2);
    bool operator>(float op2);
    friend bool operator==(circle op1,  rectangle op2);
    friend bool operator==(rectangle op1,circle   op2);
};
float rectangle::emvado()
{
    return plevra_a * plevra_b;
}
rectangle::rectangle(float a, float b)
{
    plevra_a = a;
    plevra_b = b;
    xroma = "";
}
```

Δύο υπερφορτωμένες εκδόσεις της μεθόδου υπερφόρτωσης του τελεστή > για τη κλάση rectangle

Οι δύο υπερφορτωμένες εκδόσεις της συνάρτησης υπερφόρτωσης του τελεστή == δηλώνονται ως φίλιες της κλάσης rectangle

```
rectangle::rectangle()
{
    plevra_a = 0;
    plevra_b = 0;
    xroma = "";
}
bool rectangle::operator>(rectangle op2)
{
    if (emvado()>op2.emvado())
        return true;
    else
        return false;
}
bool rectangle::operator>(float op2)
{
    if (emvado()>op2)
        return true;
    else
        return false;
}

bool operator>(rectangle op1, circle op2)
{
    if (op1.emvado()>op2.emvado())
        return true;
    else
        return false;
}
```

Πρώτη έκδοση της μεθόδου υπερφόρτωσης του τελεστή >. Χρησιμοποιείται στη περίπτωση σύγκρισης αντικειμένων κλάσης rectangle

Δεύτερη έκδοση της μεθόδου υπερφόρτωσης του τελεστή >. Χρησιμοποιείται στη περίπτωση σύγκρισης ενός αντικειμένου της κλάσης rectangle με κάποια τιμή

Συνάρτηση υπερφόρτωσης του τελεστή > που δεν είναι μέθοδος κάποιας κλάσης. Χρησιμοποιείται στη περίπτωση σύγκρισης αντικειμένων κλάσης rectangle (στο αριστερό μέλος) με αντικείμενα κλάσης circle (στο δεξί μέλος)

```
bool operator>(circle op1,  rectangle op2)
{
    if (op1.emvado()>op2.emvado())
        return true;
    else
        return false;
}
bool operator==(circle op1,  rectangle op2)
{
    float per1,per2;
    per1=2*3.14*op1.aktina;
    per2=(op2.plevra_a+op2.plevra_b)*2;
    if (per1==per2)
        return true;
    else
        return false;
}
bool operator==(rectangle op1,circle   op2)
{
    float per1,per2;
    per1=(op1.plevra_a+op1.plevra_b)*2;
    per2=2*3.14*op2.aktina;
    if (per1==per2)
        return true;
    else
        return false;
}
```

Και άλλη έκδοση της συνάρτησης υπερφόρτωσης του τελεστή > η οποία δεν είναι μέθοδος κλάσης. Χρησιμοποιείται στη περίπτωση σύγκρισης αντικειμένων κλάσης circle (αριστερό μέλος) με αντικείμενα κλάσης rectangle (δεξί μέλος)

Συνάρτηση υπερφόρτωσης του τελεστή == (δεν είναι μέθοδος κάποιας κλάσης) Σύγκρισης αντικειμένων κλάσης circle (αριστερά) με αντικείμενα κλάσης rectangle (δεξιά) περίμετρο. Θεωρεί τα δυο αντικείμενα ίσα όταν έχουν ίδια περίμετρο

Δεύτερη έκδοση της συνάρτησης υπερφόρτωσης του τελεστή ==. Και αυτή δεν είναι μέθοδος κάποιας κλάσης. σύγκρισης αντικειμένων κλάσης rectangle (αριστερά) με αντικείμενα κλάσης circle (δεξιά) Θεωρεί ότι τα αντικείμενα είναι ίσα όταν έχουν ίδια περίμετρο

```
int main()
{
    circle c1(10);
    rectangle r1(30,10);
    if (c1>r1)
        cout << "Ο κύκλος είναι μεγαλύτερος από το παραλληλόγραμμο"<< endl;
    else
        cout << "Ο κύκλος δεν είναι μεγαλύτερος από το παραλληλόγραμμο"<< endl;
    if (c1==r1)
        cout << "Ο κύκλος έχει ίδια περίμετρο με το παραλληλόγραμμο"<< endl;
    else
        cout << "Ο κύκλος έχει άλλη περίμετρο από το παραλληλόγραμμο"<< endl;
    if (c1>300)
        cout << "Ο κύκλος έχει εμβαδό πάνω από 300"<< endl;
    if (r1>250)
        cout << "Το παραλληλόγραμμο έχει εμβαδό πάνω από 200"<< endl;
    return 0;
}
```

Ο κύκλος είναι μεγαλύτερος από το παραλληλόγραμμο
Ο κύκλος έχει άλλη περίμετρο από το παραλληλόγραμμο
Ο κύκλος έχει εμβαδό πάνω από 300
Το παραλληλόγραμμο έχει εμβαδό πάνω από 200