Constructor

Copy Constructor

Assignment Operator

Constructor

Καλείται κατά την δημιουργία ενός αντικειμένου.
Δηλαδή:

- με τον ορισμό του αντικειμένου (για στατική δέσμευση)

```
Student student;
Car group_of_cars[100];
```

- με την χρήση του τελεστή new (για δυναμική δέσμευση)

```
Student *student = new Student("John");
Car *group_of_cars = new Car[100];

House *houses[100];
for (i=0;  i<100; i++)
        houses[i] = new House(i+1);
```

Copy Constructor

Καλείται κατά την δημιουργία ενός αντικειμένου, όταν το όρισμα του constructor είναι ένα υπάρχον αντικείμενο της ίδιας κλάσης. Δηλαδή:

- με τον ορισμό του αντικειμένου (για στατική δέσμευση)

```
Student student;
Student new_student (student);
Student another_student = student;
```

- με την χρήση του τελεστή new (για δυναμική δέσμευση)

```
Student student;
Student *new_student = new Student(student);

House original_house;
House *houses[100];
for (i=0;  i<100; i++)
        houses[i] = new House(original_house);
```

Assignment Operator

Καλείται για να αναθέσει την τιμή ενός υπάρχοντος αντικειμένου σε ένα επίσης υπάρχον αντικείμενο. Δεν δημιουργείται καινούργιο αντικείμενο.
Δηλαδή:

```
Student first_student;
Student first_student_copy;

first_student_copy = first_student;
```

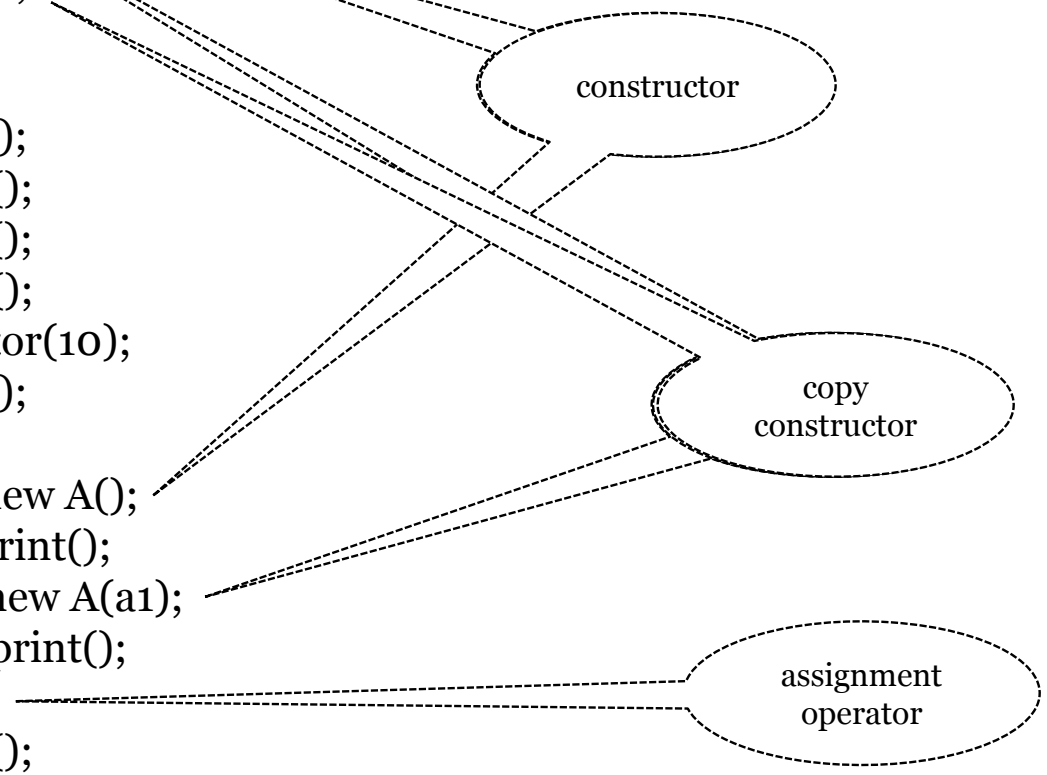
Τόσο ο copy constructor όσο και ο assignment operator είναι καλό να υλοποιούνται, γιατί η συμπεριφορά των default copy constructor/assignment operator μπορεί να μην είναι επιθυμητή (π.χ. η αντιγραφή πινάκων ή συμβολοσειρών θα γίνει με απλή ανάθεση της τιμής του ενός δείκτη στον άλλο, και όχι την αντιγραφή αυτού καθεαυτού του πίνακα ή της συμβολοσειράς)

Παράδειγμα

```
class A {  int a;
           public:
               A()
               {
                    a = 5;
                    cout << "Creating a class A object!" << endl;
               }
               A(const  A&   Acopy)
               {
                     a = Acopy.a;
                     cout << "Creating a class A object via copy constructor!" << endl;
               }
               A& operator=(const   A&   Acopy)
               {
                   if (this != &Acopy)
                   {
                      a = Acopy.a;
                      cout << "Assignment operator, no object was created!" << endl;
                   }
                   return  *this;
              }
               ~A()
               {
                     cout << "Destroying a class A object " << endl;
               }
               void print()
               {    cout << "<--- a is " << a << " --->" << endl; }
               void mutator(int i)
               {   a = i;  }
          };
```

```
int main()
{
   A a1, a2;
   A a3(a1);
   A a4 = a2;
   A *a_p1;
   A*a_p2;
   a1.print();
   a2.print();
   a3.print();
   a4.print();
   a1.mutator(10);
   a1.print();

   a_p1 = new A();
   a_p1->print();
   a_p2 = new A(a1);
    a_p2->print();
   a2 = a1;
   a2.print();
   delete a_p1;
   delete a_p2;
   return 0;
}
```



Αποτέλεσμα

```
Creating a class A object!
Creating a class A object!
Creating a class A object via copy constructor!
Creating a class A object via copy constructor!
<--- a is 5 --->
<--- a is 5 --->
<--- a is 5 --->
<--- a is 5 --->
<--- a is 10 --->
Creating a class A object!
<--- a is 5 --->
Creating a class A object via copy constructor!
<--- a is 10 --->
Assignment operator, no object was created!
<--- a is 10 --->
Destroying a class A object
Destroying a class A object
Destroying a class A object
Destroying a class A object
Destroying a class A object
Destroying a class A object
```