ΤΕΙ ΣΕΡΡΩΝ

ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΑΦΗΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

Ευάγγελος Γ. Ούτσιος
Θεόδωρος Γ. Λάντζος

Διάλεξη No2-No3

Κανόνες Ομαλής Λειτουργίας

- Ερχόμαστε στην ώρα μας
- Δεν καπνίζουμε και τρώμε εντός της αίθουσας

- Επιτρέπετε το νερό, τα αναψυκτικά και ο καφές με την προϋπόθεση να μην λερώνουμε το χώρο και πετάμε τα σκουπίδια εκτός των καλαθιών.
- Κινητά αθόρυβα και μόνον σε περίπτωση άμεσης ανάγκης.

- Σε περίπτωση συναγερμού, αποχωρούμε από την αίθουσα για το σημείο συγκέντρωσης σταδιακά, χωρίς πανικό και πιέσεις.

Συναντήσεις και Forum

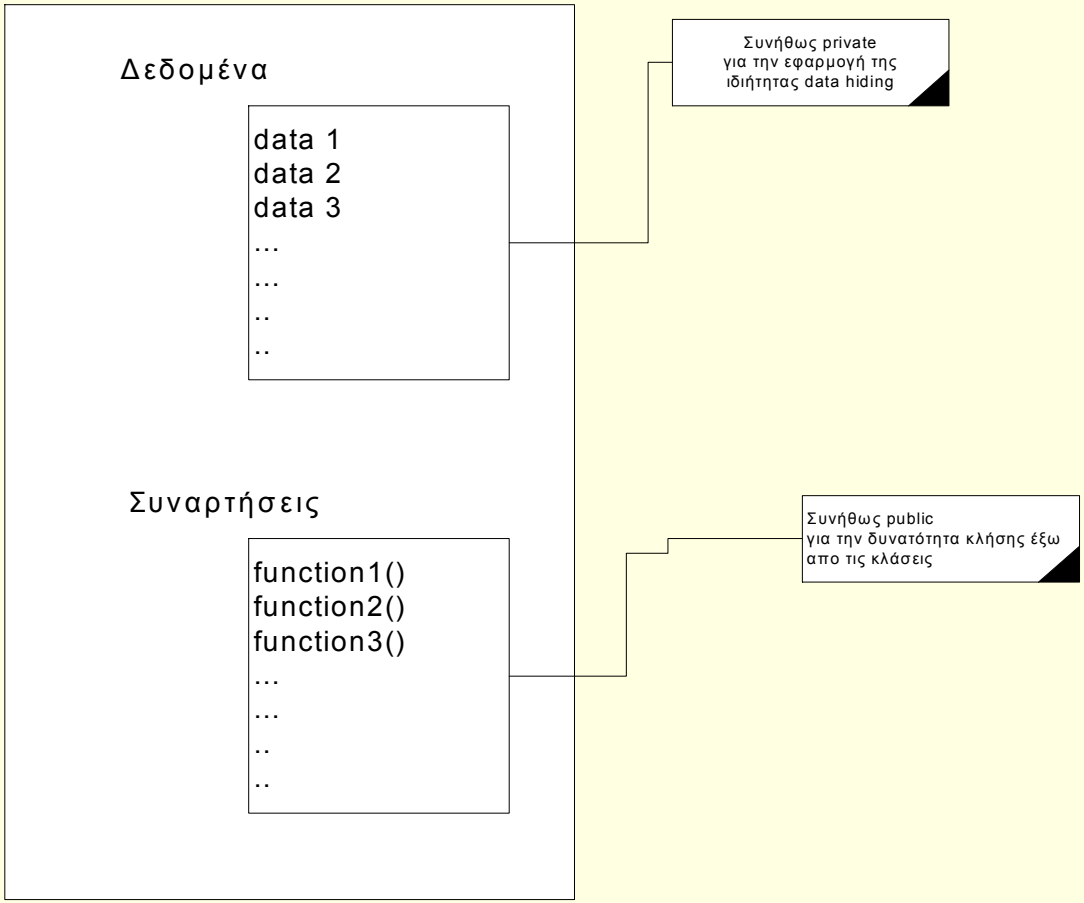
- Οι διαλέξεις θα διεξάγονται κάθε Πέμπτη 12-14 στην αίθουσα 106 ΣΤΕΦ
- Κάθε Πέμπτη 15-17 Θα υπάρχει ανοιχτό discussion forum προγραμματισμού στις επάνω αίθουσες της ΣΤΕΦ
- Ανακοινώσεις στην σελίδα μου

  http://www.teiser.gr/icd/staff/lantzos
- Ερωτήσεις δια μέσο email οι οποίες όμως θα απατούνται και αναλύονται στο forum.

  lantzos@teiser.gr

ΑΝΤΙΚΕΙΜΕΝΑ ΚΑΙ ΚΛΑΣΕΙΣ

Κλάση



Πού δηλώνουμε την κλάση στην γλώσσα C++ ;

1. Στην ίδια ακριβώς θέση όπου δηλώναμε και την struct. Δηλαδή, μετά το τέλος των directive εντολών (include, define) και πριν από την main().
2. Σε ξεχωριστά αρχεία class_name.h οι δηλώσεις της δομής της κλάσης και class_name.cpp ο κώδικας των συναρτήσεων. Στο κυρίως πρόγραμμα απλά χρειάζεται να δηλώσουμε ένα include “class_name.h”

Πού & Πώς ορίζουμε αντικείμενα;

- Με τον ίδιο ακριβώς τρόπο όπως όταν δηλώναμε μεταβλητές μια δικιάς μας δομής. (Δηλαδή. Στο χώρο δήλωσης μεταβλητών).
- Ορίζουμε ένα νέο αντικείμενο απλά δηλώνοντας το με το όνομα Κλάσης όνομα αντικειμένου.

  Π.χ. Person p1;

  Π.χ. Person p1,p2,p3;

Πώς δίνουμε αρχικές τιμές σε αντικείμενα;

- Συναρτήσεις Εγκατάστασης (constructor)
- Περισσότερες από μία
- Οι συναρτήσεις εγκατάστασης περιγράφονται στο ίδιο μέρος όπου περιγράφονται όλες οι συναρτήσεις
- Οι συναρτήσεις εγκατάστασης έχουν το ίδιο όνομα με την κλάση και μπορούνε να υπερφορτωθούν δια μέσο διαφορετικών παραμέτρων

Πώς καταστρέφουμε αντικείμενα;

- Συναρτήσεις αποσύνδεσης (destructors)
- Είναι συνάρτηση η οποία δηλώνεται μαζί με όλες τις άλλες συναρτήσεις
- Έχει το ίδιο όνομα με την κλάση
- Ξεχωρίζει από τον constructor διότι μπροστά από το όνομα έχει μια περισπωμένη ~

  π.χ ~Person()

Πώς γράφουμε και διαβάζουμε τα δεδομένα ενός αντικειμένου;

- Η κάθε κλάση χρειάζεται να έχει μηχανισμούς για ανάθεση & ανάγνωση τιμών στα χαρακτηριστικά της. Δηλαδή έναν τρόπο προσπέλασης στα δεδομένα της.
- Η προσπέλαση στα δεδομένα του αντικειμένου γίνεται με συναρτήσεις που επιστρέφουν ή τοποθετούν τιμές στα χαρακτηριστικά του αντικειμένου
- Για ανάγνωση τιμής ενός χαρακτηριστικού δημιουργούμε συνάρτηση την οποία ονομάζουμε get_variable_name() και επιστρέφει την τιμή του χαρακτηριστικού στο οποίο αναφέρεται.

```
π.χ. int get_age()
     {
          return age;
     }
```

- Για ανάθεση τιμής σε χαρακτηριστικό αντικειμένου δημιουργούμε συνάρτηση την οποία συνήθως τις ονομάζουμε set_variable_name (τύπος μεταβλητής).Αυτό γίνεται για κάθε χαρακτηριστικό της κλάσης

```
Π.χ.  Void set_age(int a)
     {
          age =a;
     }
```

Πώς λειτουργεί ένα ΟΟ πρόγραμμα;

- Δημιουργώντας αντικείμενα μέσα στο κυρίως πρόγραμμα, και καλώντας της συναρτήσεις του (user interface) αντικειμένου.
- Μια συνάρτηση ενός αντικειμένου καλείται με το όνομα_αντικειμένου.όνομα_συνάρτησης

Π.χ

```
main()
{  Person P1;
   P1.get_age();
}
```

```
#include <iostream.h>
class Person
{
   private:
         char name[30];
         int age;
   public:
         void readData()
         {
            cout << "Enter name:";
            cin >> name;
            cout << "Enter age:";
            cin >> age;
         }
         void printData()
         {
            cout << "The name of the person is " << name << endl;
            cout << "The age of the person is " << age << endl;
         }
};          // τέλος κλάσης
void main()
{
   Person  p1, p2;        // δήλωση δύο αντικειμένων

   p1.readData();     // κλήση συνάρτησης-μέλους για ορισμό δεδομένων
   p1.printData();    // κλήση συνάρτησης μέλους για εμφάνιση δεδομένων
   p2.readData();
   p2.printData();
}
```

```
#include <iostream.h>
#include <string.h>
class Person
{
   private:
         char name[30];
         int age;
   public:
         void setData(char name1[], int age1)
         {
            strcpy(name, name1);
            age = age1;
         }
         void printData()
         {
            cout << "The name of the person is " << name << endl;
            cout << "The age of the person is " << age << endl;
         }
};          // τέλος κλάσης
void main()
{
   Person  p;
   p.setData("PAPADOPOULOS", 25);
   p.printData();
}
```

```
#include <iostream.h>
class Account
{
   private:
        float  balance;
   public:
        Account()
        {
           balance = 0;
        }
        void  withdraw(float  money)
        {
           if (money <= balance)
              balance = balance – money;
           else
              cout << "Το ποσό ανάληψης υπερβαίνει το τρέχον!" << endl;
        }
        void deposit(float  money)
        {
           balance += money;
        }

        float getBalance()
        {
          return  balance;
        }
};
main()
{
   Account  ac;
   ac.deposit(100.0);
   cout << "Τρέχον ποσό λογαριασμού:" << ac.getBalance() << endl;
   ac.withdraw(70.0) ;
   cout << "Τρέχον ποσό λογαριασμού:" << ac.getBalance() << endl;
}
```

Συνάρτηση Αποσύνδεσης

```
#include <iostream.h>
class Account
{
   private:
        float  balance;
   public:
        Account()
        {
           balance = 0;
        }
        ~Account()          // συνάρτηση αποσύνδεσης
        {}
        .
        .
        .
};
```

```
#include <iostream.h>
class Account
{
   private:
        float  balance;
   public:
        Account()      // συνάρτηση εγκατάστασης χωρίς ορίσματα
        {
           balance = 0;
        }
        Account(float  balance1)   // συνάρτηση εγκατάστασης με όρισμα
        {
           balance = balance1;
        }
        void  withdraw(float  money)
        {
           if (money <= balance)
              balance = balance – money;
           else
              cout << "Το ποσό ανάληψης υπερβαίνει το τρέχον!" << endl;
        }
        void deposit(float  money)
        {
           balance += money;
        }
        float getBalance()
        {
           return  balance;
        }
};
main()
{
   Account  ac1, ac2(50.0), ac3(100.0);

   cout << "Τρέχον ποσό λογαριασμού ac1:" << ac1.getBalance() << endl;
   cout << "Τρέχον ποσό λογαριασμού ac2:" << ac2.getBalance() << endl;
   cout << "Τρέχον ποσό λογαριασμού ac3:" << ac3.getBalance() << endl;
}
```

Υπερφόρτωση

```
#include <iostream.h>
class Account
{
   private:
        float  balance;
   public:
        Account()      // συνάρτηση εγκατάστασης χωρίς ορίσματα
        {
           balance = 0;
        }
        Account(float  balance1)     // συνάρτηση εγκατάστασης με όρισμα
        {
           balance = balance1;
        }
        void  withdraw(float  money)
        {
           if (money <= balance)
              balance = balance – money;
           else
              cout << "Το ποσό ανάληψης υπερβαίνει το τρέχον!" << endl;
        }
        void deposit(float  money)
        {
           balance += money;
        }
        float getBalance()
        {
            return  balance;
        }
        void addBalance(Account  x, Account y)
        {
            balance = x.balance + y.balance;
        }
};
main()
{
   Account  ac1(100.0), ac2(70.0), ac3;

   ac3.addBalance(ac1, ac2);
   cout << "Τρέχον ποσό λογαριασμού ac1:" << ac1.getBalance() << endl;
   cout << "Τρέχον ποσό λογαριασμού ac2:" << ac2.getBalance() << endl;
   cout << "Συνολικό ποσό λογαριασμών:" << ac3.getBalance() << endl;
}
```

Αντικείμενο ως παράμετρος συνάρτησης

Κλήση

```
#include <iostream.h>
class Account
{
   private:
        float  balance;
   public:
        Account()       // συνάρτηση εγκατάστασης χωρίς ορίσματα
        {
           balance = 0;
        }
        Account(float  balance1)     // συνάρτηση εγκατάστασης με όρισμα
        {
           balance = balance1;
        }
        void  withdraw(float  money)
        {
           if (money <= balance)
              balance = balance – money;
           else
              cout << "Το ποσό ανάληψης υπερβαίνει το τρέχον!" << endl;
        }
        void deposit(float  money)
        {
           balance += money;
        }
        float getBalance()
        {
           return  balance;
        }
        Account  sumBalance(Account  ac)
        {
           Account  temp;
           temp.balance = balance + ac.balance;
           return temp;
        }
};
main()
{
   Account  ac1(100.0), ac2(70.0), ac3;

   ac3 = ac1.sumBalance(ac2);
   cout << "Τρέχον ποσό λογαριασμού ac1:" << ac1.getBalance() << endl;
   cout << "Τρέχον ποσό λογαριασμού ac2:" << ac2.getBalance() << endl;
   cout << "Συνολικό ποσό λογαριασμών:" << ac3.getBalance() << endl;
}
```

Ορισμός

Δημιουργία

Κλήση
+
Διαχείριση
Επιστρεφόμενης
Τιμής (αντικείμενο)

Επιστροφή Αντικειμένων από συναρτήσεις

Συναρτήσεις-μέλη ορισμένα έξω από την κλάση

- Χρησιμοποιώντας τον τελεστή διάκρισης εμβέλειας (scope resolution operator) μπορούμε να καθορίσουμε σε ποια κλάση ανήκει μια συνάρτηση. Συνεπώς η δήλωση αλλάζει μορφή έχοντας

τύπος_επιστρεφόμενης_τιμής Όνομα_κλάσης :: όνομα_συνάρτησης(παράμετρος1,..παράμετροςN)

Π.χ void Account ::deposit(float money)

```
{
        Balance += money;
}
```

- Δυνατότητα αλλαγής της δομής του προγράμματος με δημιουργία ευέλικτων δομών με βιβλιοθήκες και αύξηση της επαναχρησιμοποίησης τους με την χρήση των directive εντολών include
- Αναλυτικά τα πρωτότυπα της κλάσης δηλώνονται σε ένα αρχείο class_name.h πχ. Account.h
- Κώδικας των συναρτήσεων δηλώνετε σε ένα αρχείο Account.cpp
- Το πρόγραμμα το οποίο θέλουμε να δημιουργήσουμε και θα χρησιμοποιεί την κλάση Account πρέπει να περιέχει μια δήλωση στην αρχή #include “Account.h”

Αντικείμενα και κοινά δεδομένα

- Δια μέσο της δήλωσης static εμπρός μια δήλωσης χαρακτηριστικού
- Πχ.static int count;
- Το χαρακτηριστικό αυτό θα είναι ευρέως γνωστό για όλα τα αντικείμενα της κλάσης
- Το παράδειγμα που ακολουθεί παρουσιάζει το πώς ένα χαρακτηριστικό της κλάσης μπορεί να είναι ορατό από όλα τα αντικείμενα

```
#include <iostream.h>
class Account
{
  private:
    static int count = 0;
    float  balance;
  public:
    Account()      // συνάρτηση εγκατάστασης χωρίς ορίσματα
    {
      count++;
      balance = 0;
    }
    Account(float  balance1)    // συνάρτηση εγκατάστασης με όρισμα
    {
      count++;
      balance = balance1;
    }
    int  getCount()
    {
      return  count;
    }
    void  withdraw(float  money)
    {
      if (money <= balance)
        balance = balance – money;
      else
        cout << "Το ποσό ανάληψης υπερβαίνει το τρέχον!" << endl;
    }
    void deposit(float  money)
    {
      balance += money;
    }
    float getBalance()
    {
      return  balance;
    }
    Account  sumBalance(Account  ac)
    {
      Account  temp;
      temp.balance = balance + ac.balance;
      return temp;
    }
};
main()
{
  Account  ac1(100.0), ac2(70.0), ac3;

  ac3 = ac1.sumBalance(ac2);
  cout << "Τρέχον ποσό λογαριασμού ac1:" << ac1.getBalance() << endl;
  cout << "Τρέχον ποσό λογαριασμού ac2:" << ac2.getBalance() << endl;
  cout << "Τρέχον ποσό λογαριασμού ac3:" << ac3.getBalance() << endl;
  cout << "Αριθμός πελατών:" << ac1.getCount() << endl;
  cout << "Αριθμός πελατών:" << ac2.getCount() << endl;
  cout << "Αριθμός πελατών:" << ac3.getCount() << endl;
}
```

Δήλωση

Χρήση