Πανεπιστήμιο Δυτικής Μακεδονίας
Τμήμα Μηχανικών Πληροφορικής & Τηλεπικοινωνιών
Αντικειμενοστρεφής Προγραμματισμός Ι
Εργαστήριο 9

- Πολυμορφισμός

**Εκφώνηση**

Στο εργαστήριο θα δημιουργήσετε μια κλάση με το όνομα **Person** η οποία θα έχει τα ακόλουθα δεδομένα μέλη:

**Δεδομένα**

- string firstName
- string lastName

**Συναρτήσεις**

- 2 Constructors
- void ReadData ()
- void Print() (virtual συνάρτηση η οποία θα τυπώνει τα δεδομένα μέλη)
- bool isExcellent() (pure virtual συνάρτηση θα είναι ίση με το 0)

Στη συνέχεια θα δημιουργήσετε μια κλάση **Student** η οποία κληρονομεί και υλοποιεί την Person η οποία θα έχει τα ακόλουθα μέλη:

**Δεδομένα**

- float DegreeRate

**Συναρτήσεις**

- 2 Constructors
- void ReadData ()
- void Print() (virtual συνάρτηση η οποία θα τυπώνει τα δεδομένα μέλη)
- bool isExcellent() (η οποία θα επιστρέφει true αν ο βαθμός του φοιτητή είναι πάνω από 8,5 αλλιώς false)

Στη συνέχεια θα δημιουργήσετε μια κλάση **Professor** η οποία κληρονομεί και υλοποιεί την Person η οποία θα έχει τα ακόλουθα μέλη:

**Δεδομένα**

- float numberofPublications

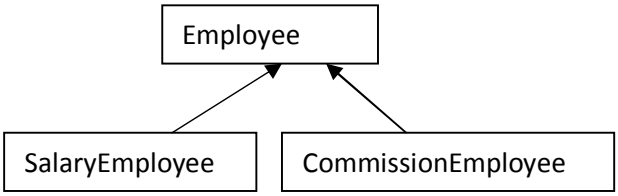
**Συναρτήσεις**

- 2 Constructors
- void ReadData ()
- void Print() (virtual συνάρτηση η οποία θα τυπώνει τα δεδομένα μέλη)
- bool isExcellent() (η οποία θα επιστρέφει true αν ο αριθμός δημοσιεύσεων του καθηγητή είναι πάνω από 50 αλλιώς false)

Στη συνέχεια στη main Θα γίνεται το εξής: Θα δηλώσετε ένα δείκτη σε πίνακα Person 100 θέσεων. Θα ρωτάτε τον χρήστη αν θέλει να εισάγει καθηγητή ή φοιτητή και μετά θα διαβάζετε τα αντίστοιχα δεδομένα και θα αποθηκεύετε την οντότητα στον πίνακα. Η διαδικασία επαναλαμβάνονται έως ότου ό χρήστης το επιθυμεί. Στη συνέχεια για κάθε Person του πίνακα θα τυπώσετε τα δεδομένα του καθώς και ένα μήνυμα αν το άτομο είναι άριστο ή όχι.

**Παράδειγμα**

**Υλοποιήση της ακόλουθης ιεραρχίας κλάσεων**




Employee.h

```
#include <string> // C++ standard string class
using namespace std;
#ifndef EMPLOYEE_H
#define EMPLOYEE_H
class Employee
{
public:
  Employee();
  Employee(string fn ,  string ln,  string ssn );
  void setEmployee(string fn ,  string ln,  string ssn); // set employee

  // pure virtual function makes Employee abstract base class
  virtual double earnings() = 0; // pure virtual
  virtual  void printEmployee();// virtual
```

```
private:
  string firstName;
  string lastName;
  string socialSecurityNumber;
};
#endif
```

**Employee.cpp**

```
#include <iostream>
#include "Employee.h"

using namespace std;
Employee::Employee()
{
 firstName="";
 lastName="";
 socialSecurityNumber="";
}
Employee::Employee(string fn ,  string ln,  string ssn )
{
 firstName=fn;
 lastName=ln;
 socialSecurityNumber=ssn;
}
void Employee::setEmployee(string fn ,  string ln,  string ssn)
{
 firstName=fn;
 lastName=ln;
 socialSecurityNumber=ssn;
}
void Employee::printEmployee()
{
 cout<<firstName<<"  "<<lastName<<"  "  <<endl<<"Social  Security
Number:"<<socialSecurityNumber;

}
```

**SalaryEmployee.h**

```
#include "Employee.h" // Employee class definition
#ifndef SALARIED_H
#define SALARIED_H
class SalaryEmployee : public Employee
```

```
{
public:
  SalaryEmployee();
  SalaryEmployee( string fn, string ln ,string ssn, double sal );

  void setSalaryEmployee( string fn, string ln ,string ssn, double sal  ); // set weekly salary

  virtual double earnings() ; // calculate earnings
  virtual void printEmployee() ; // print SalariedEmployee object
private:
  double weeklySalary; // salary per week
};
#endif
```

**SalaryEmployee.cpp**

```
#include <iostream>
#include "SalaryEmployee.h"

using namespace std;
SalaryEmployee::SalaryEmployee():Employee()
{
 weeklySalary=0;
}
SalaryEmployee::SalaryEmployee(string fn , string ln, string ssn, double sal
):Employee(fn,ln,ssn)
{
 weeklySalary=sal;
}
void SalaryEmployee::setSalaryEmployee(string fn , string ln, string ssn, double sal)
{
 Employee::setEmployee(fn,ln,ssn);
 weeklySalary=sal;
}
void SalaryEmployee::printEmployee()
{
 Employee::printEmployee();

 cout<<endl<<"Weekly Salary:"<<weeklySalary<<endl;


}
double SalaryEmployee::earnings()
{
  return weeklySalary * 4;
}
```

**CommissionEmployee.h**

```
#include "Employee.h"
#ifndef CEMPLOYEE_H
#define CEMPLOYEE_H

class CommissionEmployee: public Employee
{
public:
  CommissionEmployee();
  CommissionEmployee(string fn ,  string ln,  string ssn, double  gs, double cr );
  void setCommissionEmployee(string fn ,  string ln,  string ssn, double  gs, double cr); // set
employee

  // pure virtual function makes Employee abstract base class
  virtual double earnings(); // pure virtual
  virtual  void printEmployee();// virtual
private:
  double grossSales; // gross weekly sales
  double commissionRate; // commission percentage
};
#endif
```

**CommissionEmployee.cpp**

```
#include <iostream>
#include "CommissionEmployee.h"

using namespace std;
CommissionEmployee::CommissionEmployee():Employee()
{
 grossSales=0;
 commissionRate=0;
}
CommissionEmployee::CommissionEmployee(string fn ,  string ln,  string ssn,double gs,
double cr ):Employee(fn,ln,ssn)
{
 grossSales=gs;
 commissionRate=cr;
}
void CommissionEmployee::setCommissionEmployee(string fn ,  string ln,  string ssn, double
gs, double cr)
{
 Employee::setEmployee(fn,ln,ssn);
 grossSales=gs;
```

```
  commissionRate=cr;
}
void CommissionEmployee::printEmployee()
{
  Employee::printEmployee();
  cout<<endl;
  cout<<"Gross Sales: "<<grossSales<<endl;
  cout<<"Commission: "<<commissionRate<<endl;

}
double CommissionEmployee::earnings()
{
   return commissionRate * grossSales;
}
```

**Main.cpp**

```
#include <cstdlib>
#include <iostream>

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

#include "CommissionEmployee.h"
#include "SalaryEmployee.h"

int main(int argc, char *argv[])
{
   Employee * EmpPtr[100];
   string choice,ssn,fn,ln;
   double wsal,gs,cr;
   int i=0;
   do
   {
   cout<<"Provide Type of Employee CommissionEmplyee(c) or Salary Employee(s)";
   cin>>choice;
   if (choice=="s")
     {
      SalaryEmployee *se=new SalaryEmployee;
      cout<<"FirstName:";
      cin>>fn;
      cout<<"LastName:";
      cin>>ln;
      cout<<"Social Security Number:";
      cin>>ssn;
```

```
    cout<<"Weekly salary:";
    cin>>wsal;
    se->setSalaryEmployee(fn,ln,ssn,wsal);
    EmpPtr[i++]=se;
   }
   if (choice=="c")
   {
    CommissionEmployee *ce=new CommissionEmployee;
    cout<<"FirstName:";
    cin>>fn;
    cout<<"LastName:";
    cin>>ln;
    cout<<"Social Security Number:";
    cin>>ssn;
    cout<<"Gross Sales:";
    cin>>gs;
     cout<<"Commission rate:";
    cin>>cr;
    ce->setCommissionEmployee(fn,ln,ssn,gs,cr);
    EmpPtr[i++]=ce;
   }
  cout<<"Do you want to continue with another employee(y/n)";
  cin>>choice;
  }
  while (choice=="y");
  for(int n=0;n<i;n++)

   {
   cout<<endl;
   EmpPtr[n]->printEmployee();
   cout<<"Total monthly income: "<<EmpPtr[n]->earnings();
   cout<<endl;

   }


   system("PAUSE");
   return EXIT_SUCCESS;
}
```