|  |  |  |
| --- | --- | --- |
| C:\Users\pc dadi\Desktop\logo.jpg |  |  |



Génie informatique

Semestre 1

Année universitaire 2021/2022

Module : Programmation Orientée Objet en

C++

Rapport de Devoir Libre

Programmation Orienté Objet en C

**Réalisé par :**

AZELMAT MOUHCINE

**Encadré par :**

MR.N.KANNOUF

# **Introduction**

# programmation orienté objet :

La programmation orientée objet (OOP) est un modèle de programmation informatique qui organise la conception de logiciels autour de données, ou d’objets, plutôt que de fonctions et de logique. Un objet peut être défini comme un champ de données ayant des attributs et un comportement uniques.

**C++ :**

C++ est un langage de programmation compilé permettant la programmation sous de multiples paradigmes, dont la programmation procédurale, la programmation orientée objet et la programmation générique. Ses bonnes performances, et sa compatibilité avec le C en font un des langages de programmation les plus utilisés dans les applications où la performance est critique.

# **Exercice 1 — Construction**

## Objectif : Le but de cet exercice est de construire les classes personnes, etudiant, enseignant , appreciation .

## Les classes :

* **La classe Personnes:** est caractérisée par son nom (string), prénom (string), Age (Int) et possède une méthode afficher, setter et un operator de l’affectation.
* **La classe Etudiant:** ayant les mêmes caractéristiques qu’une personne et définissant une liste de matières suivies et un ensemble de notes. Et possède une méthode afficher une note, et une méthode pour calculer les moyennes des notes.
* **La classe Enseignant:** possède également les mêmes membres que la classe
* Personne et définit en plus une liste de matières enseignées et un nombre d’heures par matière.
* **La classe Appreciation:** ayant pour caractéristiques le nom de l’enseignant
* portant l’appréciation, le nom de l’élève concerné et un commentaire

## Code Source :

/\*\*\*\*\*\*\*\*\*\*\* la classe Personne \*\*\*\*\*\*\*\*\*\*\*/

class Personne

{

protected:

string nom;

string prenom;

int age;

public:

Personne();

Personne(string,string,int);

void affichage();

void SetNom(string);

void SetPrenom(string);

void SetAge(int);

Personne operator =(const Personne &);

~Personne();

};

Défintion les méthodes :

Personne::Personne()

{

nom="";

prenom="";

age=0;

}

Personne::Personne(string no,string pr,int a)

{

nom=no;

prenom=pr;

age=a;

}

void Personne::affichage()

{

cout<<"Nom : "<<nom<<endl;

cout<<"Prenom : "<<prenom<<endl;

cout<<"Age : "<<age<<endl;

}

void Personne::SetAge(int a)

{

age=a;

}

void Personne::SetNom(string no)

{

nom=no;

}

void Personne::SetPrenom(string pr)

{

prenom=pr;

}

Personne::~Personne()

{

nom=prenom="";

age=0;

delete(this);

}

Personne Personne::operator=(const Personne &p)

{

this->nom=p.nom;

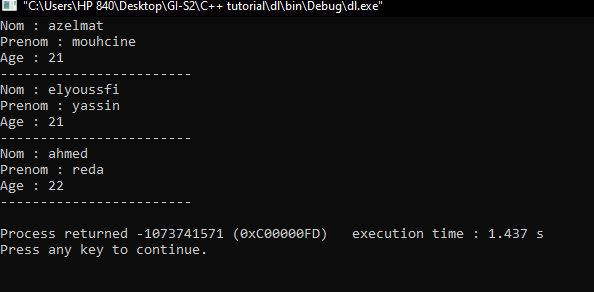
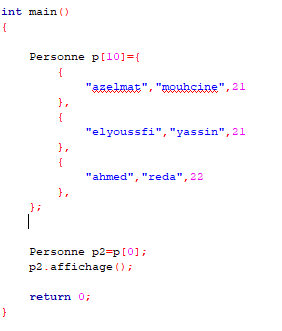
this->prenom=p.prenom;

this->age=p.age;

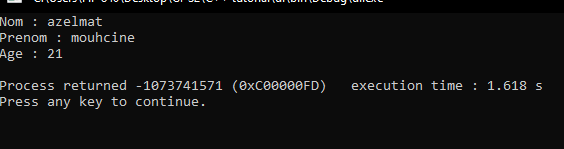
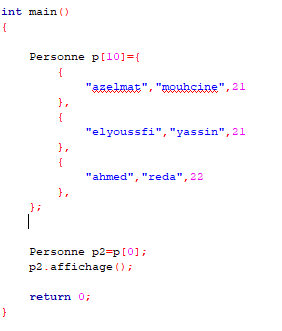
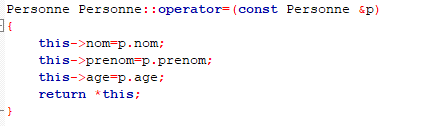
return \*this;

}

2)-tableau des personnes :



3)-l’opérateur d’affectation



B)-

1)-pour que les classes etudiant et enseignant peuvent accéder à les donnes de la class personne if faut mette le données de la class personne protected.

2)-

/\*\*\*\*\*\*\*\*\*\*\* la classe Etudiant \*\*\*\*\*\*\*\*\*\*\*/

class Etudiant : public Personne{

float \*notes;

string \*matiere;

int dimention;

public:

Etudiant();

Etudiant(string,string,int,string\*,float\*,int);

virtual ~Etudiant();

void affecterNote(string ,float);

void affichage();

friend void afficher(const Etudiant &);

};

Défintion les méthodes :

#include "Etudiant.h"

Etudiant::Etudiant()

{

notes=NULL;

matiere = NULL;

dimention=0;

}

Etudiant::~Etudiant()

{

delete notes;

delete matiere;

dimention=0;

}

Etudiant::Etudiant(string name,string pre,int a,string \* M,float \* N,int dim):Personne(name,pre,a)

{

dimention=dim;

notes=new float[dimention];

matiere=new string[dimention];

for(int i=0;i<dimention;i++)

{

matiere[i] = M[i];

notes[i]=N[i];

}

}

void Etudiant::affichage(){

Personne::affichage();

cout<<" affichage des matiere et la note correspondant : "<<endl;

for(int i=0;i<dimention;i++)

{

cout<<matiere[i]<<" : "<<notes[i]<<endl;

}

}

void Etudiant::affecterNote(string mat ,float n)

{

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

if(matiere[i]==mat){

notes[i]=n;

}

else{

cout<<"matier n'exist pas"<<endl;

}

}

}

/\*\*\*\*\*\*\*\*\*\*\* la classe Enseignant \*\*\*\*\*\*\*\*\*\*\*/

class Enseignant:public Personne

{

int dimention;

string\* matiere\_ensignee;

float\* nb\_heur;

public:

Enseignant();

Enseignant( string , string , int , string \* , float\* , int );

virtual ~Enseignant();

void affichage();

float nbTotalHeurs();

friend void afficher(const Enseignant &);

protected:

private:

};

Défintion les méthodes :

Enseignant::Enseignant()

{

matiere\_ensignee = NULL;

nb\_heur = NULL;

dimention = 0;

}

Enseignant::Enseignant( string name , string pre , int a , string\* m , float\* h , int dim):Personne(name,pre,a)

{

dimention = dim;

nb\_heur = new float[dimention];

matiere\_ensignee = new string[dimention];

for(int i=0;i<dimention;i++)

{

matiere\_ensignee[i]=m[i];

nb\_heur[i]=h[i];

}

}

Enseignant::~Enseignant(){

delete matiere\_ensignee;

delete nb\_heur;

dimention = 0;

}

void Enseignant::affichage(){

Personne::affichage();

cout<<" affichage des matiere et le nombre d'heure correspondant : "<<endl;

for(int i=0;i<dimention;i++)

{

cout<<matiere\_ensignee[i]<<" : "<<nb\_heur[i]<<endl;

}

}

float Enseignant::nbTotalHeurs()

{

float total=0;

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

total+=nb\_heur[i];

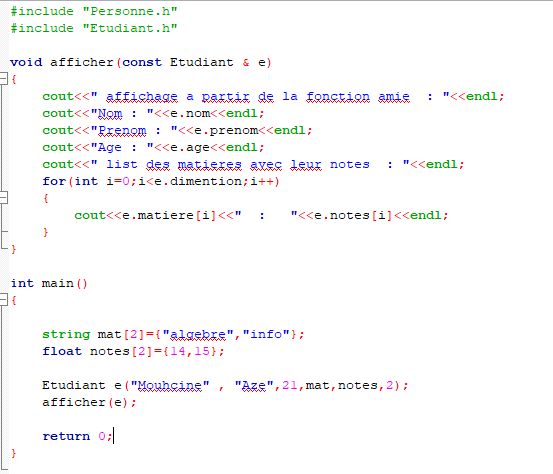
}

return total;

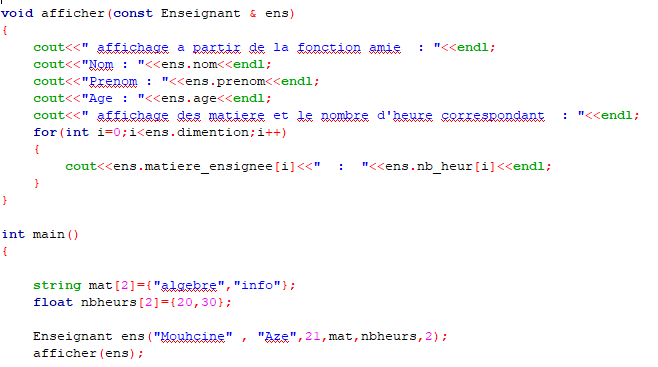
}

4)-fonction ordinaire d’affichage :

* Pour affichage d’un etudiant :

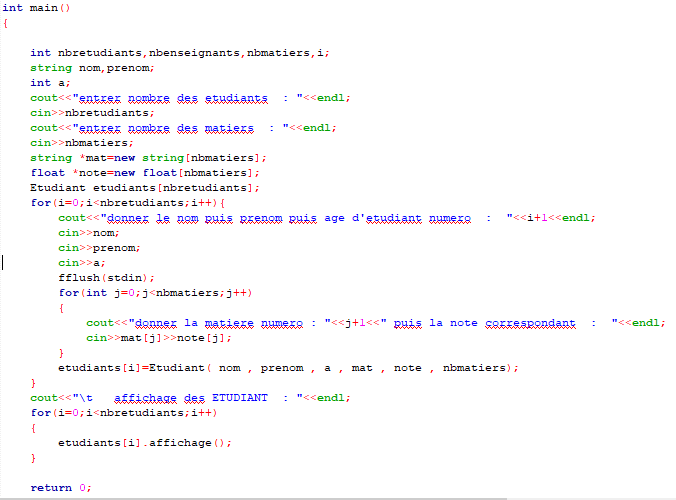


* Pour affichage d’un enseignant :

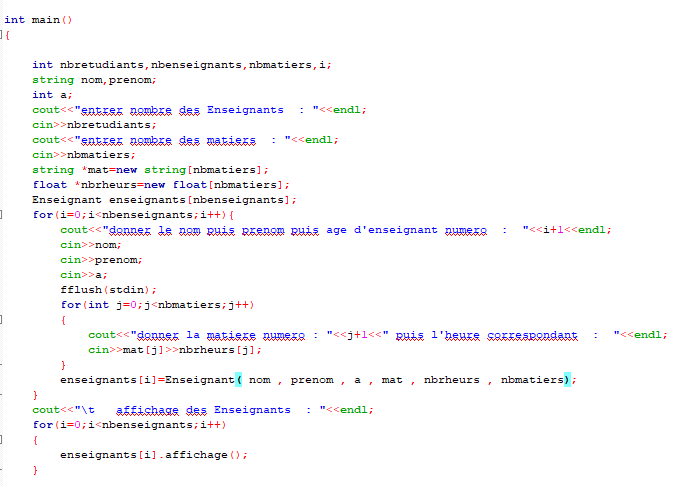


5)- tableau d’étudiants et un tableau d’enseignants :

* Tableau d’etudiants :



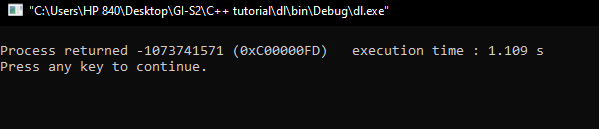
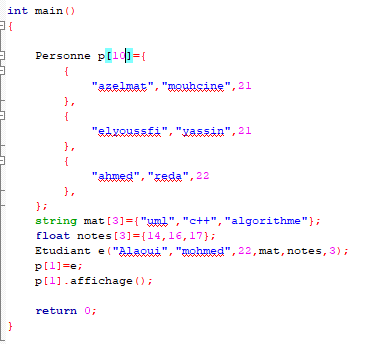
* Tableau d’enseignants :



6)-dans un tableau de personnes en affectant à certains éléments des objets de type Etudiant et

de type Enseignant:

en Remarque que dans l’affichage rien est afficher .



**C)-**

**1)-**

/\*\*\*\*\*\*\*\*\*\*\* la classe Appreciation \*\*\*\*\*\*\*\*\*\*\*/

class Appreciation

{

string nomEnseignant;

string nomEtudiant;

string commentaire;

friend class Enseignant;

public:

Appreciation();

~Appreciation();

};

* **Dans la class enseignant on doit ajouter les méthodes CreerAppreciation et AfficherAppreciation :**

Appreciation Enseignant::CreerAppreciation(string nom\_etudiant, string commentaire) {

Appreciation appreciation;

appreciation.nomEnseignant = nom;

appreciation.nomEtudiant = nom\_etudiant;

appreciation.commentaire = commentaire;

return appreciation;

}

void Enseignant::AfficherAppreciation(Appreciation appreciation) {

cout << "{enseignant: '" << appreciation.nomEnseignant << "', etudiant: '" << appreciation.nomEtudiant << "', commentaire: '" << appreciation.commentaire << "'}\n";

}

* **Exécution de main :**

