TD1 : Nouveautés C++ et Orienté Objet

Durant cette séance, vous allez :

- Connaitre l'historique sur les approches de programmation (procédurale et orientée objet) et citer les avantages de l'OO.
- Avoir une idée sur l'approche objet par rapport aux autres et son intérêt.
- Connaître les nouvelles possibilités apportées par le langage C++ par rapport au langage C indépendamment de la programmation orientée objet.
- Découvrir les concepts fondamentaux de la P.O.O (classe, instance, méthode, attribut).
- Développer une classe et exécuter le résultat.
- Découvrir la classe String avec ses méthodes.

1 *Exercice: Quelques nouveautés C++*

Réécrire le programme suivant en C++, en ne faisant appel qu'aux nouvelles possibilités d'entrées- sorties du C++ :

```
#include <stdio.h>
#include <string.h>
#define MAX 10
int main()
{
     int n;
     float x;
     int test=0 ;
     char message [10];
     printf("donnez un entier et un flottant\n");
     scanf("%d %f",&n,&x);
     if(x>MAX)
     {
          test=1 ;
          strcpy(message, "true") ;
          printf("le produit de %d par %f\n est %f\n et message est %s",n,x,n*
     x,message);
     }
     return 0;
}
```

2 *Exercice: Notion Classe, Objet et méthode*

Le consortium de compagnies aériennes Blue Sky veut un logiciel pour gérer les vols, les avions et les achats des billets.

- un vol est ouvert et fermé à l'achat de billet et à l'enregistrement sur ordre de la compagnie.
- un client peut acheter un ou plusieurs billets, pour des passagers différents mais il doit fournir un numéro de passeport pour chaque billet.
- un billet concerne un seul passager.
- l'enregistrement peut donner lieu au paiement d'un supplément bagage.
- un billet peut être annulé tant que le vol n'a pas eu lieu.
- un vol a un aéroport de départ et un aéroport d'arrivée ainsi qu'un jour et une heure de départ et d'arrivée.
- un billet peut comporter plusieurs vols avec des escales.

Vous êtes amenés à informatiser les informations relatives à ces différents objets avec le langage de programmation C++ :

1. Combien de classe peut-on créer ?
2. Combien de fichier peut-on créer ?
3. Combien d'objet peut-on créer ?
4. Une instance correspond à une classe ou à un objet ?
5. Sachant que vous êtes déjà un as dans le langage C, vous avez commencé par ce bout de code et on vous demande de le transformer en C++ :

```
#include <stdio.h>
#include <stdlib.h>

struct Vol
{

 char adress_destination [20];
 char adress_depart[20];
 char date_arrivee;
 char date_depart;

};
void afficher(struct Vol V);
void saisir(struct Vol* V);
```

3 *Exercice: Implémentation d'une classe*

On voudrait gérer les étudiants d'une institution à l'aide d'une classe Etudiant définie par les attributs suivants :

—nom : nom d'un étudiant

—prenom : prénom d'un étudiant

—tab_notes : tableau contenant les notes d'un étudiant, sachant qu'un étudiant a au total 10 notes initialisées tous à zéro.

les méthodes suivantes :

—void saisir(), permettant la saisie d'un étudiant

—void afficher(), permettant l'affichage d'un étudiant

—float moyenne(), retourne comme résultat la moyenne des notes d'un étudiant

—bool admis(), retourne comme résultat la valeur true, si un étudiant est admis et la valeur false, sinon. Un étudiant est considéré comme étant admis lorsque la moyenne de ses notes est supérieure ou égale à 10.

—int exae_quo(Etudiant E), retourne comme résultat :

- ✓ la valeur - 1, si l'étudiant passer en paramètre a une moyenne supérieure
- ✓ la valeur 0, si les deux étudiants ont la même moyenne
- ✓ et la valeur 1, sinon.

Implémentez cette classe en C++.

4 *Exercice: Définir une classe*

Développez la classe Article avec la main afin de fournir cette exécution :

```
Donner le taux de TVA pour tous les articles : 20
Le taux TVA est : 20%

Article 1:
Référence : 0
Désignation :
Prix HT : 0
Le prix TTC est 0
```

5 *Exercice: Surcharge des méthodes*

Donnez la sortie de ce programme et commentez chaque étape brièvement:

```
#include <iostream>

using namespace std;
void func( int i )
{
    cout << "Called file-scoped func : " << i << endl;
}

void func( string sz )
{
   cout << "Called locally declared func : " << sz << endl;
}

int main()
{

   func(3);  //étape 1
   func( "s" );  //étape 2
}
```

6 *Exercice: Argument par défaut*

Donnez la sortie de ce programme:

```
// C++ Program to demonstrate working of default argument
#include <iostream>
using namespace std;
void display(char c= '*', int n= 1)
{for(int i = 1; i <= n; ++i)
    {
        cout << c;
    }
    cout << endl;}
void add(int a, int b = 3)
{
cout<< « la somme est « <<a+b<<endl ;
}
int main()
{ cout << "No argument passed:\n";
  display();
  cout << "\nFirst argument passed:\n";
  display('#');
  cout << "\nBoth argument passed:\n";
  display('$', 5);
   add(4) ;
return 0;
}
```

7 *Exercice: Les références*

Donnez la sortie de ce programme et commentez chaque étape brièvement:

```
void swap(int i, int j)
{
    int tmp = i;
    i = j;
    j = tmp;
}
void swap2(int& i, int& j)
{
    int tmp = i;
    i = j;
    j = tmp;
}

int main()
{
    int x = 1;
    int y = 2;
    swap(x, y); // étape 1
    swap2(x, y); // étape 2

}
```

7 *Exercice: La Classe String*

Donnez la sortie de chaque étape dans ce code :

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


int main()
{
string str("Les sanglots longs des violons");
string st(str,0,12);
cout << "\"" << st << endl; //étape 1

string str2("de l'automne blessent mon coeur");
cout << str.replace(str.size()-1,0," "+str2) << endl; //étape 2

int n=str.find("sanglots");
cout << n << "\n";      //étape 3

for(int i=0;i<str.size();i++)
cout << str.at(i) << " " << str[i] << "\n";   //étape 4

return 0;
}
```