Objectif

Le but de ce TP est d'assimiler les notions suivantes :

- Surcharge de méthodes
- Surcharge d'opérateurs

Exercice 1 surcharge de méthode

Soit les fonctions de nom fct suivantes :

```
/* ...*/
int fct(int x);
int fct(float  y);
int fct(int x, float  y);
float fct(float x, int  y);
```

fonction.h

```
/* ...*/

int fct(int x){ std::cout<<"1:"<<x<<"\n"; return 0; }
int fct(float  y){ std::cout<<"2:"<<y<<"\n"; return 0; }
int fct(int x, float  y){ std::cout<<"3:"<<x<<y<<"\n"; return 0; }
float fct(float x, int  y){ std::cout<<"4:"<<x<<y<<"\n"; return 3.14; }

void exercice_surcharge(){
 int i=3,j=15;
 float x=3.14159,y=1.414;
 char c='A';
 double z=3.14159265;
 fct(i); //appel 1
 fct(x); //appel 2
 fct(i,y); //appel 3
 fct(x,j); //appel 4
 fct(c); //appel 5
 fct(i,j); //appel 6
 fct(i,c); //appel 7
 fct(i,z); //appel 8
 fct(z,z); //appel 9
}
```

fonction.cpp

Les appels de fonction sont-ils corrects et, si oui, quelles seront les fonctions effectivement appelées et les conversions mises en place ? Expliquer les appels incorrects et les corriger.

Exercice 2 surcharge d'opérateur

Nous reprenons l'exemple de la classe point du TP2 et nous souhaitons réaliser ce type d'opération logique :

```
int main() {
point my_point1(2.0, 3.0);
point my_point2(4.0, 5.0);
my_point1 += my_point2; }
```

1 - Définir et implémenter la méthode suivante pour surcharger l'opérateur += :

point & operator+=(point & point_);

Modifier alors les fichiers *point.h*, *point.cpp* et *main.cpp*

2 – Récrire le programme principale en simulant un exemple d'utilisation de l'opérateur +=. Réaffichez les nouvelles valeurs de l'abscisse et de l'ordonnée du point en question.

Exercice 3 surcharge d'opérateur

Soit la classe ***vecteur*** suivante :

```
class vecteur
{
private:
   float x,y;
public:
   vecteur(float abs,float ord);
   void affiche();
};

vecteur::vecteur(float abs =0,float ord = 0)
{
   x = abs; y = ord;
}

void vecteur::affiche()
{
   cout<<"x = "<< x <<"  y = "<< y <<"\n";
}
```

Surcharger l'opérateur somme + qui permettra d'écrire dans un programme principal :

vecteur v1, v2, v3;

v3 = v2 + v1;