Compte rendu
Construction, destruction objet

Table des matières

Introduction

L'objectif de ce TP est de d'utiliser des classes afin de manipuler un point dans un plan. On commencera par placer ce point, puis on pourra alors le déplacer et effectuer une homothétie de celui-ci.
Les programmes seront découpés avec des fichiers **Header** (**.h**) afin de simplifier leur lecture.

On aura donc pour chaque programme :

- Un fichier main.cpp qui contiendra la fonction **int main()**,
- Un fichier declarationClass.h contenant la **classe point**,
- Un fichier definitionClass.h contenant la définition de la **classe point.**

Exercice 1

La déclaration de la classe point se fait comme ceci (dans le fichier declarationClass.h) :

```
#include <iostream>

using namespace std;

class point {
private:
    float x, y;
public:
    void deplace (float x, float y, float leX, float leY);
    void affiche (float leX, float leY);
    void placer (float x, float y);

    point (float leX, float leY) {
        x=leX;
        y=leY;
    }

    void setX (float leX) {
        x=leX;
    }
    void setY (float leY) {
        y=leY;
    }

    float getX() {
        return x;
    }
    float getY() {
        return y;
    }

};
```

declarationClass.h

Cette classe contient 2 parties : une partie privée et une partie publique.

La partie privée contient les 2 variables flottantes **x** et **y** qui correspondent respectivement aux coordonnées x et y du point.

La partie publique contient quant à elle plusieurs éléments, à commencer par 3 fonctions :

- **Deplace**, qui permet de modifier la position du point (le déplacer),
- **Affiche**, qui permet d'afficher la valeur des variables **x** et **y** de la classe,
- **Placer**, qui permet simplement de placer le point.

Ces 3 fonctions seront définies dans le fichier definitionClass.h.
Il y a un **constructeur point** afin de pouvoir utiliser plus tard la classe.
La partie publique de la classe contient également 2 méthodes getter et 2 méthodes setter.
On configure les méthodes **get** (getter) et **set** (setter) pour **x** et **y** :

- Grâce aux méthodes **setX** et **setY** on va pourvoir initialiser et modifier les variables **x** et **y** de la classe point,
- Les méthodes **getX** et **getY** vont nous permettre de retourner les valeurs des variables **x** et **y** de la classe point.

On peut maintenant configurer les 3 fonctions de la classe point, dans le fichier definitionClass.h.

```cpp
#include <iostream>

using namespace std;

void point::placer (float x, float y) {
    cout<<"Coordonnée x du point : ";
    cin>>x;
    cout<<"Coordonnée y du point : ";
    cin>>y;

    setX(x);
    setY(y);
}

void point::deplace (float x, float y, float leX, float leY) {
    cout<<"Déplacer horizontalement de : ";
    cin>>x;
    cout<<"Déplacer verticalement de : ";
    cin>>y;

    setX(leX+x);
    setY(leY+y);
}

void point::affiche (float leX, float leY ) {
    cout<<"x : "<<getX()<<endl;
    cout<<"y : "<<getY()<<endl;
}
```

DefinitionClass.h

La fonction placer contient les éléments cout et cin afin d'afficher les instructions à l'utilisateur dans un premier temps, puis de récupérer sa saisie.
Les valeurs de **x** et **y** sont alors renvoyer à la **classe point** avec les méthodes **setX** et **setY**.

La fonction deplace fonction de la même manière que la fonction placer.

La fonction affiche utilise les méthodes **getX** et **getX** pour retourner à l'utilisateur les valeurs des variables **x** et **y**.

Voici maintenant un exemple d'exécution du programme :

```
Coordonnée x du point : 4
Coordonnée y du point : 6
x : 4
y : 6
Déplacer horizontalement de : 3
Déplacer verticalement de : 6
x : 7
y : 12
Program ended with exit code: 0
```

Exercice 2

Ce programme est similaire sur beaucoup de point au précédent, la différence se situe au niveau de la fonction affiche qui disparait sur ce programme afin de laisser place à 2 nouvelles fonctions : Abcisse et ordonnee. Ces deux fonctions fournissent en retour l'abscisse et l'ordonnée du point.

Pour declarationClass.h :
On déclare 2 fonctions : abscisse et ordonnee qui remplacent la fonction affiche

```
void deplace (float x, float y, float leX, float leY);
void abcisse (float x);
void ordonnee (float y);
void placer (float x, float y);
```

declarationClass.h

Pour definitonClass.h :
On affiche maintenant séparément la valeur des variables **x** et **y** de la **classe point**.

```
void point::abcisse(float x) {
    cout<<"x = "<<x<<endl;
}

void point::ordonnee(float y) {
    cout<<"y = "<<y<<endl;
}
```

definitionClass.h

Pour main.cpp :
L'appel des 2 nouvelles fonctions est alors nécessaires afin d'afficher les informations à l'utilisateur.

```
int main() {
    point point1(0,0);
    point1.placer(0,0);
    point1.abcisse(point1.getX());
    point1.ordonnee(point1.getY());
    point1.deplace(0,0, point1.getX(), point1.getY());
    point1.abcisse(point1.getX());
    point1.ordonnee(point1.getY());
}
```

main.cpp

Exercice 3

Dans ce dernier exercice, il est demandé d'a jouter à la classe précédente (comportant un constructeur et trois fonctions membre deplace, abscisse et ordonnee) une nouvelles fonctions membre homothetie qui effectue une homothétie dont le rapport est fourni en argument. Pour se faire :

Dans le fichier declarationClass.h :
On ajoute une nouvelle fonction homothetie.

```
void deplace (float x, float y, float leX, float leY);
void abcisse (float x);
void ordonnee (float y);
void placer (float x, float y);
void homothetie (float x, float y, float hm);
```

declarationClass.h

Dans le fichier definitionClass.h :
On demande d'abord à l'utilisateur de saisir le rapport de l'homothetie (avec cout et cin). Puis, on calcule les nouvelles coordonnées du point après l'homothétie grâce au setter **setX** et **setY** et au calcul suivant :

Nouvelle valeur du point = Ancienne valeur du point * rapport de l'homothétie

```
void point::homothetie(float x, float y, float hm) {
    cout<<"Rapport de l'homothetie : ";
    cin>>hm;

    setX(x*hm);
    setY(y*hm);
}
```

definitionClass.h

Dans le fichier main.cpp :
On appel la fonction homothetie avec le **constructeur point**.

```
int main() {
    point point1(0,0);
    point1.placer(0,0);
    point1.abcisse(point1.getX());
    point1.ordonnee(point1.getY());
    point1.deplace(0,0, point1.getX(), point1.getY());
    point1.homothetie(point1.getX(), point1.getY(), 0);
    point1.abcisse(point1.getX());
    point1.ordonnee(point1.getY());
}
```

main.cpp

Voici un exemple d’exécution du programme :

```
Coordonnée x du point : 13
Coordonnée y du point : 4
x = 3
y = 4
Déplacer horizontalement de : 7
Déplacer verticalement de : 8
Rapport de l'homothetie : 1.2
x = 12
y = 14.4
Program ended with exit code: 0
```