JUNIA Grande école d'ingénieurs

HEI · ISEN · ISA

TP3 / C++

Exercice 1 :

Soit le code C++ suivant :

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

class A {
public :
        A () { x = 0;                           cout << "1"; }
        A (int px) { x = px;                    cout << "2"; }
        A (const A& pa) { x = pa.x; cout << "3"; }
protected :
        int x;
};

class B {
public :
        B (const A& pa=A()) : a(pa) {                   cout << "4"; }
        B (const A& pa, int py) { a = pa; y = py;       cout << "5"; }
protected :
        A a;
        int y;
};

class C : public B {
public :
        C (int pz=1) { z = pz;                          cout << "6"; }
        C (A pa) : B(pa) { z = 0;                       cout << "7"; }
        C (const B& pb) : B(pb), a(1) { z = 0;          cout << "8"; }
protected :
        A a;
        int z;
};

int main() {
cout << "-- A --\n";
A a0;           cout << endl;
A a1(3);        cout << endl;
A a2(a1);               cout << endl;
A a3 = a2;              cout << endl;
a3 = a1;        cout << endl;
cout << "-- B --\n";
B b0(a0,3);     cout << endl;
B b1(a1);               cout << endl;
B b2;           cout << endl;
cout << "-- C --\n";
C c0;           cout << endl;
```

```
C c1(a1);                cout << endl;
C c2(b2);                cout << endl;
}
```

1) Compiler, exécuter ce programme.
2) Interpréter, ligne par ligne, les sorties de la fonction *main()* ;

Exercice 2 :

1) Modifier l'ensemble des classes dérivant de la classe *Shape* (premier cours) en introduisant la classe *Point2D* (une version simplifiée de la classe *Point3D* vue ci-dessus et profitez-en pour modifier cette classe afin qu'elle dérive de *Point2D*) permettant de positionner les formes géométriques (un cercle par un point et un rayon, un rectangle par deux points, un triangle par trois points…). Ces informations seront utilisées pour les dessiner à l'écran un peu plus tard.

2) Ecrire des méthodes du calcul pour l'estimation du nombre π (en utilisant la méthode de Monte Carlo et la somme des inverses des carrés). Utiliser la somme des inverses des carrés pour écrire une méthode d'estimation du nombre π où ce nombre est calculé à la compilation.

Exercice 3 :

1) développer une classe *Vecteur*, ainsi que les méthodes pour :

- construire le vecteur à partir de deux points (Point3D ou Point2D),
- calculer la somme de deux vecteurs,
- calculer le produit par un réel,
- vérifier l'égalité entre deux vecteurs.

Exercice 4 :

En utilisant la classe *Point*, écrire une classe *Polygon* :

- Un tableau de points représente les sommets.
- Une méthode permet de retourner l'aire du polygone en utilisant la formule (le dernier point coïncidant avec le premier afin que le polygone soit fermé):

$$\frac{1}{2}\sum x_i y_{i+1} - x_{i+1} y_i$$

Vérifier votre méthode pour un triangle, un carré et un rectangle dans une fonction *main*.