### 

### 8PRO128 Programmation orientée objet

###### Automne 2019

# Travail pratique #1 (10% de la session)(+5 sept)

##### Date de remise

**À remettre d’ici le 26 septembre 2019 (jeudi)**

(Peut se faire seul ou en équipe de 2 max.)

**Objectif général**

Ce travail a pour objectif d’introduire l’étudiant à la notion d’encapsulation en programmation objet et sa traduction en langage C++.

**Notions du langage C++ à acquérir**

Déclaration et définition de classe, droits d’accès aux membres, constructeurs, destructeurs, compilation en fichiers séparés.

**INSTRUCTIONS:**

* Votre travail pratique sera remis sous forme de dossier compressé. Ce dossier pourrait ne contenir qu’un document Word (le rapport décrit plus bas) et un dossier de projets (ou une solution contenant des projets) Visual Studio.
* Pour chacune des classes, vous aurez à créer un fichier de déclaration (.h) et un fichier de définition (.cpp). Le nom de chacun de ces fichiers doit être significatif de son contenu. Exemple : « Point.cpp », « Point.h » ou autre. Mettre des commentaires dans votre programme (fichier, but, auteur, date ou autre). Décrire les variables au besoin. Vous remettrez également une copie d’écran montrant l’exécution de votre programme.
* Remettre un diagramme de classe pour la classe *Point* et la classe *Segment.*

### Partie 1

Définir/décrire les termes suivants par rapport à la programmation objet. Soyez brefs et citez vos sources.

|  |  |
| --- | --- |
| **Terme** | **Définition/Description** |
| Classe |  |
| Objet |  |
| Attribut |  |
| Méthode |  |
| Accès public |  |
| Accès privé |  |
| Constructeur |  |
| Destructeur |  |
| Encapsulation |  |
| Instance |  |
| **Sources (ou URL)** |  |

### Partie 2

Vous devez créer une application traitant des objets de formes géométriques. Il vous faut donc d’abord créer les classes de base. Les classes de base seront:

D’abord une classe Point :

|  |
| --- |
| Point |
| x (double)  Exemple : point d’origine : 1,2  glisser (3,3) : le point devient 4,5  deplacer (3,3) : le point devient 3,3  y (double) |
| glisser (dx,dy)  deplacer (x,y)  afficher() |

Ajoutez les méthodes nécessaires à une bonne encapsulation (constructeurs, destructeurs et accesseurs).

Testez!

### Partie 3

Ajoutez ensuite les classes suivantes :

|  |  |  |  |
| --- | --- | --- | --- |
| Figure | Attributs | Comportement |  |
| Segment | a (Point)  b (Point) | glisser (dx,dy)  déplacer (PointA,PointB)  afficher()  longueur() | (les points doivent demeurer même sans segment) |
| Triangle | côté1 (Segment)  côté2 (Segment)  côté3 (Segment) | glisser (dx,dy)  calculerAire()  calculerPérimètre()  calculerHauteur(côtéi)  afficher() | (p\*(p-côté1)\*(p-côté2)\*(p-côté3)) ½  côté1 + côté2 + côté3  2 \* aire / côtéi  formule de Héron pour le calcul de l'aire:  <https://msdn.microsoft.com/fr-ca/library/ace5hbzh(v=vs.120).aspx> |
| Rectangle | base (Segment)  hauteur (Segment) | glisser (dx,dy)  calculerAire()  calculerPérimètre()  calculerDiagonale()  afficher() | base \* hauteur  (base + hauteur ) \* 2  (base2 + hauteur2 ) ½ |
| Cercle | Rayon (Segment)  (le centre sera le premier point donné au rayon) | glisser (dx,dy)  calculerAire()  calculerPérimètre()  calculerDiamètre()  afficher() | πr2 (π = M\_PI)  2πr  2r |

Vous ajouterez les accesseurs en lecture (get) et en écriture (set), une méthode d’affichage du contenu ainsi que les constructeurs nécessaires à l’instanciation d’objets (dont un constructeur par défaut) et les destructeurs.

**Bon travail!**