TP2 : Introduction au C++

Exercice 1 :

a) Compilez, corrigez, et exécutez le programme en utilisant l'IDE de votre choix:

```
#include <iostream.h>
#define multiplier (x,y) (x * y)

int main(void)
{
        int a;
        int b;
        cout << "Entrez une valeur :" << endl;
        cin << a;
        cout << "Entrez une autre valeur :" << endl;
        cin << b;

        cout << multiplier (a+1, b+1) << endl;
}
```

b) Au regard de (a), explorez la possibilité d'utiliser les mots clés constexpr et consteval vus en cours pour remplacer la macro « multiplier » ci-dessus..

Exercice 2 :

1) Ecrire un programme C++ qui affiche la table de multiplication d'un chiffre entre 1 et 9 saisi par l'utilisateur.

2) Ecrire un programme C++ qui affiche l'intégralité de la table de multiplication parfaitement alignée (en utilisant la librairie *<iomanip>*)

Exercice 3 :

1) Ecrire un programme C++ qui analyse un texte à partir d'un fichier en affichant le nombre de lignes, de mots et de lettres (l'utilisation de la classe «stringstream» est préconisée).

2) Modifier le programme pour qu'il affiche aussi le nombre d'occurrences de chaque lettre de l'alphabet (sans différencier les majuscules des minuscules).

Exercice 4 :

Soit la classe *Point3D* définie comme suit (dans *Point3D.hpp*) :

```
class Point3D {
        private:
                float x,y,z;   // private attributes

        public:
                // constuctors
                Point3D(); // fill X Y Z with random values (from 0 to 100)
                Point3D(const float &newx, const float &newy, const float &newz); // fill XYZ values

                // Setters and getters
                void setXYZ(const float &newx, const float &newy, const float &newz);
                void setX(const float &newx);
                void setY(const float &newy);
                void setZ(const float &newz);
                float getX();
                float getY();
                float getZ();

                // other methods
                void print();    // prints the coordinates of the point
                float distanceTo(const Point3D &otherPoint3D);
};
```

1) Développer l'ensemble des méthodes dans le fichier *Point3D.cpp*.

2) Tester les différentes méthodes dans la fonction *main()*.

Remarque : Pour la génération de nombres aléatoires, faire appel aux fonctions *rand()* et *srand()* de la bibliothèque *<cstdlib>* (ou les nouvelles fonctions de la bibliothèque « randow » propre au C++).

Exercice 5 :

Soit la classe *Trajectory* définie comme suit (dans *Trajectory.hpp*) :

```
#include "Point3D.hpp"

constexpr size_t numberOfPoints = 10;
class Trajectory{
        private:
                Point3D points[numberOfPoints];
        public:
                void print(); // print the coordinates of all points
                Point3D & getPoint(const int &n); // returns the reference of point n
                float getTotalDistance();
};
```

1) Développer l'ensemble des méthodes dans le fichier *Trajectory.cpp*.

2) Tester les différentes méthodes dans la fonction *main()*.

3) Modifier la classe pour que le nombre de points soit dynamique. Le constructeur prend obligatoirement le nombre de points en paramètre et alloue l'espace nécessaire. Il faut aussi développer un destructeur pour la classe.