Licence E.E.A.
TD N°5 de Langage C/C++

Ce TD a pour objectif de vous faire découvrir et utiliser le langage *C++*. Il s'agira de transcrire sous la forme d'un programme *C++*, des algorithmes afin de réaliser les fonctions demandées.

1 Affectation et Affichage

Le but de ce programme est d'effectuer l'affectation de 3 entiers, d'effectuer la somme de deux d'entre eux et écrire à l'écran le mot "bonjour le monde !".
Notions introduites : la directive `cout`.

2 Actualités : Élections

Dans une élection, I est le nombre d'inscrits, V le nombre de votants, Q le quorum, $P = 100 \times V/I$ le pourcentage de votants, $M = V/2 + 1$ le nombre de voix pour obtenir la majorité absolue.
Le quorum est le nombre minimum de votants pour que le vote soit déclaré valable.
Écrire un programme qui :

1. demande à l'utilisateur de saisir I, Q et V,
2. teste si le quorum est atteint,
3. si oui calcule et affiche P, M, sinon affiche un message d'avertissement.

3 Jeu : le 421

En utilisant `srand` et `rand`, écrire le programme qui permet de jouer au 421.

4 Programme pour l'électronique

Saisir les 3 couleurs d'un résistance ; et affiché sa valeur. Une fonction `conversion` calcule le nombre associé à chaque couleur. `couleur` est une chaîne de caractère.

5 Les arguments par défaut

Écrire une fonction `puissance` qui renvoie x^n sans utiliser la fonction `pow`. Par défaut cette fonction devra calculer x^4.

6 La surdéfinition de fonctions

Écrire une fonction `puissance(double, double)` qui retourne x^y (et qui utilise la fonction `pow`).
Écrire une *autre* fonction `puissance(double, int)` qui retourne x^n en utilisant la fonction écrite dans le programme précédent.

7 Notion de classe

Définir une classe `point` qui définit un point en coordonnées cartésiennes. Cette classe possédera les méthodes suivantes :

1. le constructeur `point`,
2. `initialise` qui permet d'initialiser un point,
3. `deplace` qui permet de modifier les coordonnées d'un point,
4. `affiche` qui permet d'afficher les coordonnées du point,
5. le destructeur `~point()`.

8 Surdéfinition et fonctions membres en ligne

Réécrire la classe point en utilisant la surdéfinition de fonction membre pour définir plusieurs constructeurs de point. On utilisera aussi les fonctions en ligne pour la définition de ces constructeurs.

9 Objet transmis et retourné par une fonction membre

À partir de la classe précédente, ajouter la méthode `coincide` qui permet de tester si deux points sont identiques (passage d'un paramètre à une fonction membre).
Ajouter la méthode `symetrique` qui renvoie un `point` (retour de paramètre par valeur), puis la même méthode qui renvoie le point par adresse et une enfin dernière méthode qui renvoie le point par référence.

10 Le mot clé `this`

Ce mot clé désigne l'adresse de l'objet invoqué. Il est utilisable uniquement au sein d'une fonction membre.
Modifier la méthode `coincide` pour qu'elle prenne en paramètre un `point *` et utiliser le mot clé `this`.