LANGAGE C – TP 4
Fonctions et modules

Les codes proposés devront impérativement avoir la structure définie dans la fiche de TP1 et des commentaires pertinents seront ajoutés. Une grande importance doit être apportée à la lisibilité des codes.

Exercice 1

1) Ecrire une fonction qui prend en paramètre un flottant correspondant à une température en degrés Celsius et renvoie un flottant correspondant à la température en degrés Fahrenheit. La conversion de degrés Celsius vers degrés Fahrenheit se fait suivant la formule : $T_F = \frac{9}{5}T_C + 32$

   De même écrire une fonction effectuant la conversion des degrés Fahrenheit vers les degrés Celsius.

   Le programme principal demandera à l'utilisateur quelle conversion il souhaite faire, appellera la fonction adéquate et affichera le résultat de la conversion demandée. Les températures seront affichées avec 2 chiffres après la virgule (format %.2f) et un message d'erreur sera affiché si l'utilisateur saisit au clavier une option différente de celles proposées, à savoir 1 pour la conversion Celsius-Fahrenheit et 2 pour la conversion Fahrenheit-Celsius.

2) On s'intéresse à l'exécution du programme amenant à la sortie suivante :

```
Quelle conversion souhaitez-vous ?
        1 - Conversion Celsius vers Fahrenheit
        2 - Conversion Fahrenheit vers Celsius
1

Temperature en degres Celsius a convertir :  20.5

Une temperature de 20.50 degres C correspond a 68.90 degres F

Appuyez sur une touche pour continuer...
```

   Décrire l'état de la pile à chaque étape de l'exécution de ce processus.

Exercice 2

On souhaite écrire un code calculant les surfaces et volumes de certaines figures géométriques.
Le projet sera défini en trois parties :

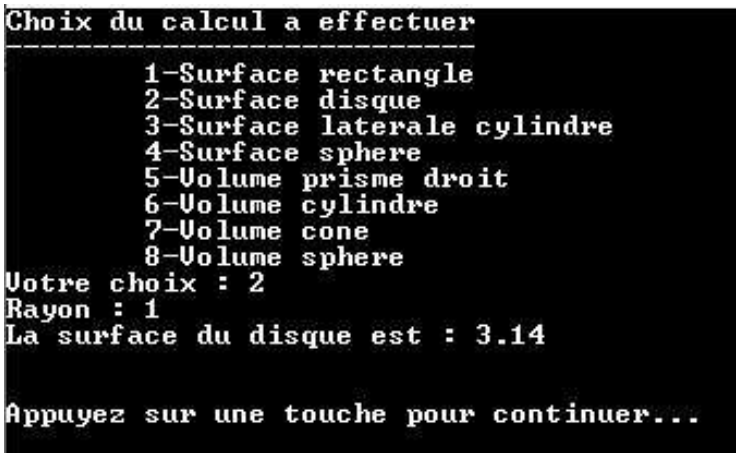
- le programme principal, permettant une interface avec l'utilisateur.
- un module surface, permettant de calculer les surfaces de certaines figures géométriques. Les fonctions définies dans ce modules doivent pouvoir être utilisées par le programme principal et par le module volume ;
- un module volume, permettant de calculer les volumes de certaines figures géométriques. Ce module doit pouvoir utiliser les calculs de surfaces afin d'effectuer ses propres calculs.

Si l'environnement ne permet pas la création de modules (C online, par exemple), définir les fonctions en séparant bien les différentes familles de fonctions.

1) Créer le projet et le programme principal.
2) Créer le module surface, qui permet de calculer les surfaces suivantes :
   - `double rectangle (double largeur, double hauteur)`, qui retourne la surface d'un rectangle (longueur × largeur) ;
   - `double disque (double rayon)`, qui retourne la surface d'un disque ($\pi \times \text{rayon}^2$) ;
   - `double enveloppeCylindre (double rayon, double hauteur)`, qui retourne la surface latérale d'un cylindre ($2 \times \pi \times$ rayon × hauteur) ;
   - `double surfaceSphere (double rayon)`, qui retourne la surface d'une sphère ($4 \times \pi \times \text{rayon}^2$).

   Une constante `pi = 3.14159` sera définie, qui devra avoir pour portée l'ensemble du module surface. Cette variable ne pouvant être modifiée (donnée constante plutôt que statique), elle sera déclarée comme suit : `const double pi = 3.14159 ;`

3) Créer le module volume, qui permet de calculer les volumes suivants, qui doit importer le module surface :
   - `double volumePrisme (double largeur, double hauteur, double longueur)`, qui retourne le volume d'un prisme droit (surface du rectangle de base × hauteur) ;
   - `double volumeCylindre (double rayon, double hauteur)`, qui retourne le volume d'un cylindre (surface du disque × hauteur) ;
   - `double volumeCone (double rayon, double hauteur)`, qui retourne le volume d'un cône (volume du cylindre associé / 3) ;
   - `double volumeSphere (double rayon)`, qui retourne le volume d'une sphère ($4/3 \times \pi \times \text{rayon}^3$).
4) Dans le cas d'écriture de modules, écrire le fichier headers.
5) Ecrire le programme principal, qui demande parmi les possibilités énumérées ci-dessus sous forme d'un menu numéroté le choix de l'utilisateur, demande les valeurs nécessaires aux calculs en fonction du choix, effectue le calcul demandé (message d'erreur en cas de réponse non valide) et affiche le résultat. La demande formulée à l'utilisateur se fera grâce à une fonction, `int affiche_menu()`, définie dans le programme principal. On prendra $\pi = 3.14159$ en première intention.



Exercice 3

On s'intéresse à une chaîne de caractères initialisée dans le programme. Cette chaîne de caractères ne contient que des minuscules et aucun caractère accentué ou signe de ponctuation. Le code ASCII pour le caractère 'a' est 97, celui de 'b' est 98, …

On rappelle que le module `string` contient une fonction `strlen(chaine)` retournant le nombre de caractères contenus dans la chaîne nommée `chaine`. Par ailleurs, les formats pour les caractères et chaînes de caractères sont respectivement `%c` et `%s`.

Enfin, la bibliothèque `assert`, permet d'effectuer des tests : `assert (`*`condition`*`)` provoquera la sortie du code si la condition spécifiée n'est pas respectée.

1) Ecrire une fonction de signature `bool recherche(char chaine[], char cherche)` permettant de tester la présence du caractère `cherche` dans la chaîne de caractères `chaine`. Indiquer l'état de la pile pour l'appel de `recherche (texte, caractere)` si `chaine = "aabbccddee"` et `caractere = 'a'`.
2) Ecrire une fonction de signature `int rechercheMaxi (int tableau[], int n)`, recevant un tableau d'entiers de taille `n` et renvoyant l'indice de l'élément le plus grand.
3) Ecrire une fonction de signature `char plusFrequent (char chaine[])` renvoyant le caractère le plus fréquent dans la chaîne passée en argument. Pour ce faire, la fonction devra construire un tableau de 26 entiers correspondant à la fréquence de chaque caractère de l'alphabet. La correspondance entre table ASCII ('a' -> 97) et les entiers pourra être judicieusement utilisée. Par ailleurs, cette fonction devra utiliser la fonction `rechercheMaxi` et s'assurer que l'entier renvoyé est compatible avec la taille du tableau (utiliser le `assert`).