TP 1 - C

Les Tps se feront sous un environnement linux, vous pouvez utiliser l'éditeur de texte que vous preferez.

Partie I : Bases

1. "Bonjour !"

Le but de cet exercice est de vous familiariser avec les notions de base de la compilation. Nous allons créer un programme qui affiche « Bonjour ! »

- Créez un répertoire projet ainsi qu'un fichier .c :

```
mkdir tp1
cd tp1
touch exercice1.c
```

- Ouvrez le fichier exercice1.c dans votre éditeur et écrivez le code suivant :

```
#include <stdio.h>
int main() {
      printf("Bonjour\n");
}
```

- Sauvegardez le fichier.
- Compilez le fichier avec la commande suivante : `gcc exercice1.c`
- Exécutez ce programme : `./a.out`

2. Pair ou impair :

Écrivez un programme qui demande à l'utilisateur d'entrer un entier et l'enregistre dans une variable de type char.

a) Affichez dans la console cet entier.
b) Affichez « pair » ou « impair » dans la console selon si l'entier est pair ou impair.

note : Un entier a est pair si a % 2 == 0.

note : Vous pouvez vous aider de la diapositive 16 du cours.

3. Attraction :

On considère un parc d'attraction pour lesquels les billets dépendent de l'age. Le prix d'un ticket est :

- 10€ pour les plus de 60ans.
- 50€ pour les personnes de 20 à 59 ans.
- 15€ pour les personnes de 5 à 19 ans.
- gratuit pour les personnes de moins de 5ans.

a) Écrivez un programme qui demande son age à l'utilisateur et lui indique le prix qu'il devra payer pour un ticket.

4. Comparaisons d'entiers

Créez un programme qui demande deux entiers à l'utilisateur.

a) Affichez le plus grand de ces deux entiers dans la console.
b) Affichez la valeur absolue de ces deux entiers.

c) Affichez l'entier avec la plus grande valeur absolue dans la console.

5. Calculs avec des flottants

Créez un programme qui demande à l'utilisateur un flottant représentant le rayon d'un cercle :

a) Affichez à l'écran la surface du cercle (on considère que $\text{surface} = 3.14 * \text{rayon}^2$).
b) Affichez à l'écran le périmètre du cercle ($\text{perimetre} = 2 * 3.14 * \text{rayon}$).

6. Initiales

Créez un programme qui demande la première lettre des prénoms de trois personnes et qui affiche ces lettres dans l'ordre alphabétique.

7. Course de vitesse

On considère ici une course de vitesse dans laquelle les participants gagnent des points selon leur performance et leur age : Le nombre de points gagné est calculé ainsi :

$$\text{points} = \frac{\left(\frac{\text{distance}}{\text{temps}}\right)^2}{\text{age}}$$

a) Créez un programme qui va demander la première lettre du prénom (charactère), l'age (entier) et le temps pour courir 100m (flottant) de deux personnes et affiche la première lettre du prénom de la personne ayant le meilleur score.
b) Modifiez le programme pour qu'il demande la distance parcourue et affiche le score du gagnant.
c) Modifiez le programme pour qu'il affiche « Bravo » à la fin du programme si le joueur donc le prénom arrive avant dans l'alphabet a gagné et « Dommage » sinon.

Partie II : Exercices avancés

7. Opérateurs binaires

On cherche ici à enregistrer 5 entiers compris entre 0 et 63 en utilisant le minimum d'espace mémoire possible. Pour cela on utilisera un entier 32 bits et on stockera les nombres dans l'espace mémoire qui lui est reservé.

Exemple :

> On veut enregistrer les nombres : 10, 1, 9, 63 et 33 dans un entier de 32 bits :
> Les représentations binaire de ces nombres sont : 001010, 000001, 001001, 111111 et 100001
> On peut donc les sauvegarder en mémoire comme le nombre avec la représentation binaire 00001010000001001001111111100001 soit 168067041

Utilisez les opérateurs binaires (<<, >>, &, |, ^) pour effectuer ces opérations et créez un programme qui aura deux modes (on demandera à l'utilisateur d'entrer 1 ou 2 pour choisir le mode)

- Le mode 1 demandera à l'utilisateur d'entrer 5 entiers et affichera à l'écran un entier dont la représentation binaire contient les entiers entrés précédement
- Le mode 2 fera l'opération inverse et demandera à l'utilisateur d'entrer 1 entier puis affichera les 5 entiers de 6 bits qu'il contient

8. Opérateur ternaire

Le C propose un opérateur, écrit « ? » appelé ternaire, il permet de choisir une valeur selon le résultat d'une condition.

La syntaxe est la suivante :

```
condition ? resultat_si_vrai : resultat_si_faux
```

Par exemple pour affecter la valeur 17 ou 24 à un entier selon si un charactère et égal à p on écrira :

```
int a = (c == 'p' ? 17 : 24)
```

a) En utilisant l'opérateur ternaire plutôt que if/else, écrivez un programme qui demande 6 variables à l'utilisateur et affiche la somme des entiers pairs multipliée par le nombre d'entiers impairs .

Exemple : si on entre 2 3 4 5 6 8 on affiche 40 car

$$(2 + 4 + 6 + 8) * 2 = 4 = 0$$

9. Macros

Le langage C permet d'écrire des « macros » cela permet de remplacer automatiquement des morceaux de texte par d'autres dans notre fichier.

Par exemple si on veut écrire une macro qui remplace le texte `SET_PLUS_ONE(x, y)` par `x = y + 1` on écrit :

```
#define SET_PLUS_ONE(x, y) x = y + 1
```

ainsi a chaque fois que nous écrirons `SET_PLUS_ONE(a, b)` le code sera remplacé par `a = b + 1`

Par exemple

`SET_PLUS_ONE(x, 5 + 5)` sera remplacé par `x = 5 + 5 + 1`

a) écrivez une macro CHANGE(i, pos, val) qui permet de changer la valeur du bit d'un entier à une position donnée.

Exemple : le code suivant changera la position du premier bit de l'entier a et affichera donc 11 dans la console:

```
int a = 10 ;
CHANGE(a, 0, 1)
printf(«%d\n », a) ;
```

a) En vous inspirant du code de l'exercice 7, créez des macros PUT et GET qui permettent d'acceder aux entiers de 6 bits enregistrés dans l'entier de 32 bits.

Exemple :

```
int a ;
PUT(a, 0, 8) // enregistre 8 dans les 6 premiers bits de a
PUT(a, 1, 37) // enregistre 37 dans les 6 bits suivants de a
printf(«%d\n », GET(a, 1)) ; // affiche 37
printf(«%d\n », GET(a, 0)) ; // affiche 8
```