L'objectif de ce TP est de prendre en main le langage C. Comme vous avez déjà pu le constater en cours, les bases du C sont très proche de celle du ijava, pour ne pas dire identique dans de nombreux cas. Néanmoins, il existe un certain nombre de différences, en particulier dans la manière dont le programme est structuré.

Le but de ce TP est de prendre en main le langage C. Les exercices proposés sont majoritairement des exercices que vous avez déjà faits en AP ou en système. À la fin du TP, vous devez tous avoir au moins réussi les exercices 1 à 5 qui correspondent aux manipulations des composants de base en C. Les exercices suivants vous permettront d'aller toujours plus loin.

Pour ce premier TP, vous écrirez un fichier distinct pour chaque programme demandé. Si le programme à écrire est `exemple`, vous créerez et complèterez le fichier source `exemple.c`. Ensuite, vous devrez le compiler grâce à la commande `gcc -Wall -o exemple exemple.c`. Vous pourrez enfin l'exécuter comme tout fichier exécutable : `./exemple`.

À toutes fins utiles, il est rappelé que des fichiers exemples se trouvent sur moodle pour chacune des fonctionnalités dont vous pourriez avoir besoin. Il est aussi rappelé que votre fichier source doit respecter un format très précis rappelé ci-dessous.

```c
/* Inclusion des librairies standards */
#include <stdio.h>

/* Déclaration de la fonction principale : main */
int main (void)
{
  /* Déclaration des variables */
  int i = 4;

  /* Initialisation des variables */

  /* Corps de la fonction */
  printf("J'ai mangé %d pommes\n",i);

  /* Envoie de la valeur de retour (nombre d'erreurs) */
  return 0;
}
```

Vous effectuerez l'ensemble du TP dans le dossier `~/R2.04/TP01`.

Exercice 1 : Prise en main

Q1. Créez le dossier `~/R2.04/TP01` et déplacez vous dedans.

Q2. Recopiez le programme `exemple` présenté ci-dessus. Compilez-le et exécutez-le. (N'oubliez pas que le source doit s'appeler `exemple.c` et l'executable `exemple`.)

Q3. Les sections 2 et 3 de `man` contiennent le manuel des fonctions standards du C. Tout comme en R1.04, vous pouvez vous y référer pour avoir les détails d'une fonction. Lisez `man 3 sleep` et ajouter une instruction utilisant `sleep` avant l'instruction utilisant `printf`.

Exercice 2 : Par ici la monnaie

Dans cet exercice nous allons travailler avec les variables. Pensez à consulter l'exemple "variables" sur le moodle.

De retour de vacances passées en Angleterre, Alice et Bob ont dans leur bourse commune 60 livres Sertling. Afin de partager en deux la somme restante, ils souhaitent savoir la valeur totale en euros qu'ils détiennent.

Q1. Écrire un programme `conversion` qui déclare 2 variables `float devise` ici initialisé à 60, et `float taux_change` initialisé à 1.1226 (cas 1 livre serling = 1.1226 euros) et qui affiche la somme en euros que possède Alice en Bob puis la part qui revient à chacun. Votre programme doit afficher les lignes suivantes.

```
Alice et Bob ont 67.36 euros après conversion.
Ils gardent donc 33.68 euros chacun.
```

Q2. Alice et Bob sont en réalité partis en Septembre, à ce moment-là, le taux de change était de 1.09. Modifiez votre programme pour savoir quel aurait été la répartition dans ce cas. Votre modification doit être simplement une réaffectation de `taux_change` et votre programme doit donner les lignes suivantes.

```
Alice et Bob ont 65.40 euros après conversion.
Ils gardent donc 32.70 euros chacun.
```

Exercice 3 : Le plus petit nombre

Dans cet exercice nous allons travailler avec le conditionnel “if”. Pensez à consulter l’exemple “if_et_boucles” sur le moodle.

Écrivez un programme `plus_petit` qui déclare 3 entiers `int a;`, `int b;`, `int c;`, qui les initialise puis qui affiche la valeur du plus petit d’entre eux. Testez votre programme avec

- `a=8, b=5, c=4`,
- `a=3, b=5, c=4`,
- `a=8, b=5, c=9` et
- `a=3, b=5, c=9`.

Exercice 4 : Les boucles

Dans cet exercice nous allons travailler avec les boucles. Pensez à consulter l’exemple “if_et_boucles” sur le moodle.

Q1. Écrivez un programme `c_while` qui, en utilisant une boucle while, affiche les entiers de 0 à 9, avec un entier par ligne.

Q2. Écrivez un programme `c_for` qui, en utilisant une boucle for, affiche les entiers de 0 à 9, avec un entier par ligne.

Q3. Depuis votre terminal, en utilisant la commande `time`, comparez le temps d’exécution de `c_while` et de `c_for`. Comparez ensuite avec le temps d’exécution du script bash `compter_for` rédigé en R1.04 (une copie de ce script se trouve à l’emplacement `/home/public/baste/R2.04/tp/tp01/compter_for`. Comment expliquez-vous les similitudes et les différences de temps de calcul ?

Exercice 5 : Affichage du contenu d’un tableau

Dans cet exercice nous allons travailler avec les tableaux. Pensez à consulter l’exemple “tableaux” sur le moodle.

Q1. Écrivez un programme `affiche_tableau` qui déclare et initialise un tableau d’entier (ex : `int tableau[3] = {5,4,7};`) et qui affiche son contenu de la manière suivante.

```
tableau[0] contient 5
tableau[1] contient 4
tableau[2] contient 7
```

Q2. Dans ce même programme, déclarez un nouveau tableau de taille 100 que vous n’initialiserez pas, vous ajouterez une boucle pour afficher son contenu. Que pouvez-vous dire de ce contenu ?

Exercice 6 : Max d’un tableau

Q1. Écrivez un programme `max_tableau` qui déclare et initialise un tableau d’entier (ex : `int tableau[6] = {5,4,7,14,2,3};`), qui le parcourt puis qui affiche la valeur du plus grand entier de ce tableau.

Q2. Améliorez votre programme pour qu’il vous affiche, en plus, l’indice du tableau qui contient la plus grande valeur. Avec l’exemple précédent, vous devez obtenir l’affichage suivant.

```
La valeur maximale est 14.
Elle se trouve en position 3.
```

Exercice 7 : Diviseurs

Le but de cet exercice est d’écrire un programme qui calcule la liste des diviseurs d’un entier donné. On rappel qu’un entier m est un diviseur d’un entier n si “n modulo m vaut 0”, ce qui se test, en C par le test `n%m == 0`.

Q1. Écrivez un programme `diviseur` qui déclare et initialise une variable `int n;`, qui parcours tous les entiers de 1 à n et affiche les entiers qui divisent n.

Q2. Lorsque n est initialisé à 10, votre retour doit être le suivant.

```
1
2
5
10
```

Q3. Lorsque n est initialisé à 36, votre retour doit être le suivant.

```
1
2
3
4
6
9
12
18
36
```

Exercice 8 : FizzBuzz

Le test `FizzBuzz` est un exemple de test utilisé pour les entretients d'embauche pour obtenir une idée du niveau de programmation des candidats. Dans cet exercice, vous devrez écrire un programme `FizzBuzz` qui parcourt les entiers de 1 à 30 et qui pour chaque entier affiche :

— "fizz" si le nombre est divisible par 3 mais pas par 5
— "buzz" si le nombre est divisible par 5 mais pas par 3
— "fizzbuzz" si le nombre est divisible par 3 et par 5
— l'entier en question si rien d'autre ne s'applique

Q1. Quelle condition permet de tester qu'un nombre n est divisible par 3 ? par 5 ?

Q2. Écrivez un programme `FizzBuzz`. Vous devriez avoir le retour suivant.

```
1
2
fizz
4
buzz
fizz
7
8
fizz
buzz
11
fizz
13
14
fizzbuzz
16
17
fizz
19
buzz
fizz
22
23
fizz
buzz
26
fizz
28
29
fizzbuzz
```

Exercice 9 : Tri d'un tableau

Q1. Écrivez un programme `tri` qui déclare et initialise un tableau d'entier (ex : `int tableau[6] = {5,4,7,14,2,3};`), qui l'affiche comme dans l'Exercice 5, puis qui le tri dans l'ordre croissant et l'affiche à nouveau.

Q2. (Bonus) Écrivez un programme `tri_alpha` qui déclare et initialise un tableau de caractères (ex : `char tableau[6] = {'r', 't', 'd', 'a', 'p', 'b'};`), qui l'affiche comme dans l'Exercice 5, puis qui le tri dans l'ordre alphabétique et l'affiche à nouveau.

Q3. (Bonus) Écrivez un programme `frequence` qui déclare et initialise un tableau de caractères (ex : `char tableau[10] = {'r', 't', 'd', 'a', 'p', 'b', 'd', 'p', 'b', 'p'};`), qui l'affiche comme dans l'Exercice 5, puis qui le tri en plaçant d'abord les caractères les plus présents et en cas d'égalité, par ordre alphabétique. Appliqué sur l'exemple, `frequence` doit afficher :

```
tableau[0] contient p
tableau[1] contient p
tableau[2] contient p
tableau[3] contient b
tableau[4] contient b
tableau[5] contient d
tableau[6] contient d
tableau[7] contient a
tableau[8] contient r
tableau[9] contient t
```

Q4. (Bonus) Le tri rapide est l'un des tris considérés comme les plus efficaces en pratique. Renseignez-vous sur son fonctionnement (wikipedia : https://fr.wikipedia.org/wiki/Tri_rapide) et implémentez ce tri.