TP n° 1

I3 : Prise en main de l'environnement de programmation

Dans votre répertoire utilisateur, créez un répertoire `I3`. Ensuite, pour chaque TP, vous créerez dans ce répertoire un répertoire `TPX`, où X est le numéro du TP. Par exemple, aujourd'hui vous devez créer un répertoire `TP1`. Dans ce répertoire, vous créerez un répertoire par exercice.

Exercice 1 [Compilation d'un programme]

1. Placez-vous dans le répertoire `~/I3/TP1/exercice1` que vous venez de créer.
2. Considérons le programme C suivant :

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main(){
5    int i;
6    i = 3;
7    printf( "Bonjour !\n" );
8    /* Fin du programme */
9    return EXIT_SUCCESS;
10 }
```

3. Que fait ce programme ?
4. Recopiez ce programme dans un éditeur de texte (vim ou emacs). Enregistrez-le dans le fichier `programme1.c`. Nous allons maintenant le compiler étape par étape.
5. Avec le compilateur gcc, on peut n'appeler que le pré-processeur en utilisant l'option `-E`. Enregistrez le code obtenu dans un fichier `programme1.i`.
6. Regardez le fichier `programme1.i`. Que constatez-vous ?
7. Avec le compilateur gcc, on peut générer le code en langage d'assemblage avec l'option `-S`. Transformez le code du fichier `programme1.i` pour obtenir le fichier en langage d'assemblage avec l'aide du compilateur.
8. Regardez le fichier `programme1.S`. Que constatez-vous ?
9. Avec le compilateur gcc, on peut générer le code objet avec l'option `-c`. Compilez le code `programme1.S` et enregistrez le résultat dans le fichier `programme1.o`.
10. Regardez le fichier `programme1.o`. Que constatez-vous ?
11. On peut effectuer l'édition des liens avec le compilateur gcc en utilisant l'option `-o`. Enregistrez l'exécutable obtenu dans un fichier `programme1`.
12. Exécutez le programme programme1.

Exercice 2 [Entrée et sortie standard, appels de fonctions]

1. Écrivez un programme C qui demande à l'utilisateur de saisir un entier, puis un autre.
2. Écrivez une fonction en C qui effectue cette saisie. La fonction prend les deux entiers en paramètre d'entrée-sortie (attention à la façon dont sont passés ces paramètres).

3. Appelez cette fonction depuis la fonction `main()` d'un programme C, et affichez les valeurs saisies par l'utilisateur.
4. Écrivez une fonction C qui inverse les valeurs de deux entiers passés en paramètres. On veut récupérer les valeurs de sortie de ces deux entiers, attention à la façon dont sont passés les paramètres.
5. Appelez cette fonction depuis la fonction `main()` du programme C écrit précédemment (après la saisie et l'affichage des deux variables). Affichez le résultat après l'inversion.

Exercice 3 [Catégorie d'âge]

Écrire un programme qui demande à l'utilisateur d'entrer l'âge d'un enfant et affiche la catégorie à laquelle il appartient :

- moins de 12 ans : trop jeune
- 13 à 14 ans : minime
- 15 à 16 ans : cadet
- 17 à 18 ans : junior
- plus de 18 ans : senior

Exercice 4 [Calcul des termes d'une suite]

1. On veut calculer et afficher les N premiers termes de la suite définie par $u_n = \sum_{i=1}^{n} u_i$. Écrivez une fonction en C qui prend comme paramètre l'entier n et retourne le résultat du calcul.
2. Écrivez un programme C qui demande à l'utilisateur de saisir un entier n, appelle cette fonction pour calculer la valeur de la suite u_n et l'affiche.

Exercice 5 [Boucles imbriquées et étoiles]

1. Écrire une procédure en C qui affiche la fonction suivante en prenant en paramètre la hauteur du dessin :

```
*----
**---
***--
****-
*****
```

2. Écrire un programme en C qui demande à l'utilisateur de saisir la hauteur du dessin et appelle la procédure qui effectue l'affichage du dessin.