Mise à niveau UNIX
Licence Bio-informatique

TP4 - Pour bien démarrer en C.

Révisions Unix.

Exercice 1.

1. Dans votre répertoire `unix`, créez un dossier `tp4`.
2. Dans votre répertoire `tp2`, vous avez un fichier `hello.c` copiez le dans votre répertoire `tp4`.
3. Dans le répertoire `/ens/tassonc/programme`, vous trouverez un programme `a_corriger.c`, copiez-le dans le répertoire `tp4`.
4. Vérifiez le contenu de votre répertoire `tp4`.

Les débuts en C.

Premiers programmes.

Exercice 2.

Le programme `a_corriger.c` contient des erreurs.

1. Essayez de compiler ce programme avec la commande `gcc -Wall a_corriger.c`. Que se passe-t-il ?
2. Ouvrez le fichier `a_corriger.c` avec `emacs`. En vous inspirant de la présentation du programme `hello.c`, mettez en forme et debuggez le programme `a_corriger.c` jusqu'à ce que la commande `gcc -Wall a_corriger.c` ne détecte plus d'erreur.
3. Quand vous avez terminez et à partir du shell, renommez le fichier `a_corriger.c` en un fichier `corrige.c`.

Exercice 3.

Ecrivez un programme `nom.c` qui grâce à `printf` affiche votre nom. A l'aide de la commande
`gcc -Wall -o mon_nom nom.c` transformez ce fichier en un exécutable `mon_nom`. Exécutez votre programme `mon_nom` à l'aide de la commande `./mon_nom`.

Programmes interactifs.

Exercice 4.

Ecrivez un programme `cell.c` qui quand on lui rentre un nombre de cellules et un nombre total de protéines, renvoie le nombre moyen de protéines par cellule.

Pour cela, vous pouvez suivre les étapes suivantes :

- Déclarer les variables entières `cell` et `protein`.
- Afficher une invite demandant à l'utilisateur le nombre de cellules.
- Stocker sa réponse (à l'aide de `scanf`) dans la variable `cell`.
- Afficher une invite demandant à l'utilisateur le nombre total de protéines.
- Stocker sa réponse dans la variable `protein`.
- Renvoyer une phrase avec le nombre moyen de protéines par cellule, en précisant le nombre de cellules.

Après l'avoir compilé, testez votre programme. Que se passe-t-il si l'utilisateur rentre 0 comme nombre de cellules ?

Exercice 5.

Ecrivez un programme `age.c` qui demande à l'utilisateur son année de naissance et l'année courante puis lui dit son âge.

Compilez et testez votre programme.

Conditionnelles.

Exercice 6.

Le programme `sexiste.c` demande à l'utilisateur le premier chiffre de son numéro de sécurité sociale (1 pour les garçons, 2 pour les filles). Il répond `Bonjour madame.` si c'est une fille et `Salut mec !` sinon.

Programmez-le, compilez-le, testez-le.

Exercice 7.

Le programme `cellbis.c` est une version améliorée du programme `cell.c`. Si l'utilisateur ne rentre pas un nombre négatif ou nul il renvoie un message d'erreur, sinon, il renvoie le nombre moyen de protéines par cellules.

Programmez-le, compilez-le, testez-le.

Exercice 8.

Le programme `agebis.c` est une version améliorée du programme `age.c`. Il demande à l'utilisateur son année de naissance, et l'année courante. Puis il teste si les données sont cohérentes (l'année de naissance est antérieure à l'année courante) et renvoie l'âge ou un message d'erreur.

Programmez-le, compilez-le, testez-le.

Exercice 9.

Le programme `max.c` calcule le maximum (le plus grand) de deux nombres fournis par l'utilisateur.

Programmez-le, compilez-le, testez-le.

Exercice 10.

Le programme `divcent.c` teste si un nombre fourni par l'utilisateur est divisible par 100.

Indication : on pourra utiliser `%` qui calcule le reste de la division euclidienne.

Programmez-le, compilez-le, testez-le.

Exercice 11.

Le programme `max3.c` calcule le maximum de trois nombres fournis par l'utilisateur.

Programmez-le, compilez-le, testez-le.

Conditionnelles filées.

Exercice 12.

Le programme `trad.c` demande à l'utilisateur de lui rentrer un entier positif. Puis il traduit ce nombre de la façon suivante :

- Si ce nombre est négatif, alors il renvoie une erreur.
- Si ce nombre est 0, alors il renvoie `aucun`.
- Si ce nombre est 1, alors il renvoie `un`.
- Si ce nombre est 2, alors il renvoie `deux`.
- Dans les autres cas, il renvoie `plusieurs`.

Exercice 13.

D'après `wikipedia`, *Une année bissextile est une année de 366 jours au lieu de 365, c'est-à-dire une année comprenant un 29 février.*

Depuis l'instauration du calendrier grégorien, sont bissextiles, les années :

- *divisibles par* 4 *mais non divisibles par* 100
- *ou divisibles par* 400.

Le programme `bissextile.c` demande à l'utilisateur une année et lui dit s'il y a un 29 février.

Programmez-le, compilez-le, testez-le.

Envoyez vos codes.

Exercice 14.

1. Si vous ne l'avez pas fait, faites les exercices 1, 2, et 3 du TP3.
2. Créez une archive compressée `login_C.tar.gz` qui contient tous les programmes C que vous avez rédigé pendant ce TP.
3. Lancez la commande `pine` qui vous permet d'envoyer des mails avec pièces jointes. Envoyez un mail à votre chargée de TP avec en pièce jointe l'archive compressée contenant vos programmes C.

Pour aller plus loin

Rappel : Les exercices de ce paragraphe sont facultatifs.

Exercice 15.

Le programme `heure.c` demande à l'utilisateur l'heure de fin du tp et l'heure actuelle. Il lui renvoie le nombre de minute qu'il reste jusqu'à la fin du tp.

Programmez-le, compilez-le, testez-le.

Indication : il faudra lire deux paires de nombres (h, m) et (h', m') et distinguer le cas où $m \geq m'$ du cas $m < m'$.

On n'utilisera que des variables entières.