

TP INITIATION - Langage C

Conseils et marche à suivre

- Il est impératif de garder à l'écran deux fenêtres : l'une contenant l'édition du programme courant (`vi`, `gedit` ou `kate`) et l'autre prête à exécuter votre compilation.
- Les programmes doivent être compilés, exécutés et testés régulièrement : à chaque nouvelle fonctionnalité par exemple.
- Lorsque l'on vous demande de modifier de manière importante la structure de votre programme (une variable globale qui devient locale, une nouvelle source d'acquisition de l'information), travaillez sur une copie du programme. Il est important de garder les versions antérieures fonctionnelles.
- Ne pas hésiter à tester à petite échelle (petit nombre d'éléments) avant d'augmenter
- Des affichages réguliers des variables ou l'utilisation du débogueur permet de vérifier le bon fonctionnement de votre programme.
- Commenter votre code !

1^{ère} séance : affichage, saisie, tests et boucles

Premier programme : « Hello World »

- Taper, compiler et exécuter le fameux « Bienvenue les ZZ1 »
- Commenter chaque ligne du code, une à la fois, compiler et noter les messages d'erreurs
- Définir dans la fonction `main()` quelques variables et tester les différents opérateurs et les `++`
- Tester les `cast` et vérifier les messages d'erreur ou d'avertissement du compilateur

Comprendre et exécuter un programme développé par un tiers

- Récupérer dans votre compte un fichier disponible sur http://fc.isima.fr/~loic/download/zz_a_recuperer.c
- Compiler et exécuter ce programme
- Fait-il ce que l'on attend de lui. Si non, faire ce qu'il faut

Résolution d'une équation de degré 2

- Demander à l'utilisateur les coefficients de l'équation à résoudre
- Afficher le nombre de solution(s)
- Afficher maintenant les solutions réelles quand elles existent. La fonction racine carrée s'appelle `sqrt()`. Elle est déclarée dans la bibliothèque `math.h`.
- Il faut modifier la ligne de compilation comme suit :

```
gcc prog.c -o prog -lm
```

Boucles

- Demander à l'utilisateur un nombre et décompter jusqu'à 0 : le faire avec toutes les boucles disponibles : `for`, `while`, et `do while`.
- Afficher les carrés jusqu'à un nombre donné
- Afficher les tables de multiplication

2^{ème} séance : fonctions et manipulation de tableaux

Variable globale

- Définir une variable globale et l'initialiser
- Vérifier sa valeur dans le programme principal
- Modifier la valeur de la variable dans une fonction
- Vérifier sa valeur dans le programme principal.

Variable locale

- Définir une variable locale au programme principale
- Passer cette variable à une fonction et la modifier
- Vérifier que la nouvelle valeur dans le programme principal

Calcul de factorielle et tests :

- Ecrire une fonction qui calcule avec une **boucle** la valeur de $n!$. Cette valeur ne sera pas affichée par la fonction mais renvoyée !
- Vérifier quelques valeurs dans le programme principal
- Ecrire une fonction **récursive** qui calcule la valeur de $n!$
- Le programme principal affichera et vérifiera que les deux fonctions renvoient bien les mêmes résultats sur quelques valeurs. Attention, on atteint très vite la limite de définition des `int`.

Premières manipulations de tableaux

- Déclarer un tableau de 10 éléments de type entier (`int`)
- Saisir les valeurs au clavier
- Afficher le tableau ainsi que la somme des éléments

Tableaux et fonctions (1)

- Déclarer un tableau de 50 éléments comme une variable **globale**.
- Mettre tous les éléments à 100 ;
- Ajouter l'indice de tableau à l'élément pour tout indice pair
- Enlever l'indice du tableau à l'élément pour tout indice impair.
- Ecrire une fonction qui affiche le tableau
- Ecrire une fonction qui renvoie le minimum.
- Ecrire une fonction qui renvoie le maximum.
- Ecrire une fonction qui renvoie la moyenne.
- Afficher ces valeurs dans le programme principal

Tableaux et fonctions (2)

- Ecrire un autre programme avec les mêmes fonctionnalités mais en considérant que le tableau est maintenant une variable **locale** à la fonction `main()`.
- Le tableau sera passé en paramètre aux différentes fonctions.

3^{ème} séance : « mastermind »

Mastermind

- Faire tirer à l'ordinateur un nombre aléatoire entre 0 et 100 sans l'afficher.
- Ecrire un programme qui demande à l'utilisateur un nombre tant que celui-ci est différent du nombre de départ
- On aidera l'utilisateur en lui indiquant si le nombre est trop petit ou trop grand.
- On affichera ensuite le nombre de tentatives effectuées.
- Modifier le jeu pour le pimenter :
- Demander au joueur le nombre de tentatives qu'il pense faire, avant que le jeu commence. Si le joueur ne trouve pas le nombre ou s'il se trompe sur l'estimation, c'est perdu !

Chaînes de caractères et texte

- Demander un prénom **p**
- Demander un nombre **n**
- Afficher n fois « Merci **p** »
- Compter le nombre de lettres de p sans utiliser `strlen()`
- Comparer le résultat avec `strlen()`
- Demander le nom de la personne.
- Placer dans des chaînes de caractères plus grandes (copie et concaténation) les formules (nom et prénom) et (prénom et nom) avec les fonctions standards `strcpy()` et `strcat()`.
- Pouvez-vous faire la même chose en réécrivant les deux fonctions ci-dessus.

Table ASCII

- Afficher les caractères de la table ASCII de 32 à 127
- Que se passe-t-il se passe pour les caractères inférieurs à 32 ou plus grands que 128 ?

4^{ème} et 5^{ème} séances : mo mo motus

Moteur de jeu

- Déclarer un tableau statique de 16 chaînes de caractères contenant au plus 6 caractères. Initialiser le tableau avec par exemple AGENT, AGILE, AIDER, AIMER, BADGE, BIÈRE, CABLE, CACHE, CACAO, GEEEK, GRIPE ☺, ISIMA, LAMPE, LARGE, MAGIE, POULE

```
char tableau[][6] = { "...", "...", ... }
```
- Choisir un mot au hasard dans le tableau sans l'afficher
- Demander à l'utilisateur de saisir un mot. Si le mot est trop court ou trop long, l'essai est perdu. Il faut afficher les lettres bien placées à leur place, un tiret pour les lettres non déterminées et à côté les lettres mal placées. Les lettres bien placées sont toujours connues. On pourra donner la première lettre
- On pourra limiter le nombre d'essais de l'utilisateur

Lire un fichier texte

- Déclarer un tableau comme pour le moteur de jeu avec plus de lignes
- Stocker dans ce tableau le contenu d'un fichier texte dont la première ligne est le nombre de mots contenus dans ce fichier et qui contient par la suite un mot par ligne. On impose que tous les mots aient le même nombre de lettres
- Afficher le tableau pour vérifier que la lecture est correcte
- Modifier le moteur de jeu pour prendre en compte cette nouvelle fonctionnalité. On choisira pour le tableau de chaînes de caractères une taille maximale fixée par une constante.

Elargir la base de jeu (bonus)

- Récupérer le fichier dictionnaire.txt sur <http://fc.isima.fr/~loic/download>
- Ecrire un programme capable d'extraire tous les mots d'une longueur donnée sur les 92482 présents pour les mettre dans un nouveau fichier texte
- Il faudra convertir les minuscules en majuscules et enlever les accents. Les mots contenant des apostrophes pourront être tout simplement enlevés.
- Modifier le moteur de jeu pour être capable de choisir un fichier de mots triés