

Devoir en temps limité n°1 – 2h

Calculatrices autorisées

On veillera à présenter très clairement sa copie : il faut rédiger les réponses et encadrer les résultats. Pour le code, il doit être indenté, on ne commence pas une fonction en bas de page et on utilise de la couleur pour les commentaires.

Le code doit être commenté dès qu'il dépasse les 5 lignes.

Le code C demandé, sauf demande explicite, doit être écrit dans une fonction et pas dans le `main`. On supposera que toutes les bibliothèques nécessaires ont été importées.

Exercice 1 Un peu de C

1. Écrire une fonction `int factorielle(int n)` qui calcule la factorielle de n de manière **récursive**.
2. Écrire la fonction `main` et des instructions qui permettent d'afficher le résultat de la fonction factorielle sur 4 et 12.
3. Quelles lignes dans le terminal faut-il utiliser pour compiler et exécuter le code de la question précédente? (on supposera qu'il est dans un fichier nommé `code.c`)
4. Écrire une fonction `bool divisible(int a, int b, int c)` qui teste si $a * b$ est divisible par c .
5. Écrire une fonction `celsius_to_fahrenheit` qui prend en entrée une température en degrés celsius et renvoie une température en degrés fahrenheit. (La formule est $f = 32 + 9/5 * c$ avec f en fahrenheit et c en celsius) C'est à vous de trouver le type d'entrée et de sortie pertinent.
6. Écrire des fonctions qui affichent les figures suivantes, de taille n variable. (les figures présentées sont de taille $n = 6$)

```
*****
*       *
*       *
*       *
*       *
*****
```

```
      *
     ***
    *****
   *********
  ***********
 *****
*****
  *****
   *****
    *****
     ***
      *
```

7. Qu'est-ce qu'un pointeur? Dessiner l'état de la mémoire (pas besoin de faire apparaître les adresses) après les instructions suivantes :

```
void f(int x){
    int* p = &x;
    *p = 5;
    int** q = &p;
    int y = 4;
    *q = &y;
    y=6;
    x=12;
}
```

Exercice 2 Un peu de bash

On suppose que l'on se trouve dans le répertoire personnel `/home/bob` de Bob. On ne se déplacera pas dans un autre répertoire sauf si cela est explicitement demandé.

Vous pouvez utiliser la commande suivante :

- `cp chemin1 chemin2` copie le dossier ou fichier correspondant au chemin `chemin1` à l'endroit indiqué par le chemin `chemin2`. Exemple : si dans le répertoire courant il existe un fichier `original.txt`, alors la commande `cp original.txt /dossier_exemple/copie.txt` copie le fichier dans le dossier `dossier_exemple` situé dans le répertoire personnel et le renomme `copie.txt`.

Traduire chacune des requêtes suivantes de manière à pouvoir les exécuter successivement sur un terminal en bash :

1. Lister le contenu du répertoire courant.
2. Lister toutes les informations sur le fichier `fichier_secret_bob.txt` qui est dans le répertoire courant.

On suppose que le résultat est le suivant :

-r-rw-rw- 1 bob amis_bob 15466 1 janv. 2024 fichier_secret_bob.txt

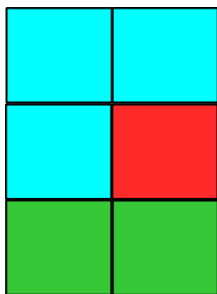
3. Qui est le propriétaire du fichier?
4. Que peuvent faire les amis de Bob avec le fichier?
5. Changer les droits de ce fichier pour donner les droits en lecture et écriture à son propriétaire, les droits en lecture à son groupe et rien aux autres.
6. Créer un répertoire vide de nom **info** dans le répertoire courant.
7. Aller dans ce répertoire.
8. Créer un fichier vide **colle.txt**.
9. Retourner dans le répertoire personnel de Bob.
10. Créer un répertoire vide **physique** dans le répertoire courant.
11. Copier le fichier **colle.txt** dans ce répertoire.
12. Aller dans **info**.
13. Sans changer de répertoire, donner les droits en lecture seule à tout le monde pour le répertoire **physique**.

Exercice 3 Un peu de binaire

1. Comment écrit-on 247 en binaire? Et 1459?
2. Qu'est ce qu'un octet? Combien d'octets faut il pour stocker le nombre 123456789?
3. Faire les opérations suivantes **en binaire**, en se limitant à un octet, et en expliquant la méthode et le résultat :
12+34, 124-56, 38-102

Informatiquement parlant, une image est une matrice de pixels. Le format d'image bitmap **.bmp** se code avec 3 octets par pixel, représentant chacun le niveau de rouge, de vert et de bleu dans la couleur du pixel. Il y a une subtilité supplémentaire, qu'on ignorera.

Par exemple l'image suivante de taille 2*3 devrait être représentée par (les valeurs RGB sont en décimal pour que ça soit plus lisible) :



0 255 255	0 255 255
0 255 255	255 42 42
55 200 55	55 200 55

4. Combien d'octets sont nécessaires pour une image de taille 2500*2500? Quelle est la taille maximale (en nombre de pixels) d'une image pouvant être stockée sur 2.0 Go, sur 250 Mio?

Exercice 4 Lecture de code

On considère les codes suivants

1. Dessiner un tableau avec les valeurs de x, y et i avant la boucle, à la fin de chaque tour de boucle et à la fin de la fonction, pour les entrées $x = 12$, $y = 937500$.

```
int f(int x, int y){
    int i=0;
    while(x>=0){
        y=y/5;
        i+=1;
        x=x-i;
    }

    if(i>6){
        y=y/2;
    }
    else{
        y=y-1;
    }
    return y
}
```

2. Donner l’affichage complet (avec les retours à la ligne) de cette fonction pour $n = 5$.

Que fait cette fonction ?

```
void mystere(int n){
    int i = 0;
    int a = 1;
    int j, b, c;

    while (i < n){
        j = 0;
        b = 1;
        c = a;

        while (j < i){
            printf("%d ", a/(b*c));
            c = c/(i-j);
            j = j + 1;
            b = b*j;
        }

        printf("%d ", a/(b*c));
        printf("\n");
        i = i + 1;
        a = a*i;
    }
}
```

Exercice 5 Un peu de récursivité

On définit une fonction A sur $\mathbb{N} \times \mathbb{N}$ par :

$$\begin{cases} A(0, p) = p + 1 & \text{pour } p \geq 0 \\ A(n, 0) = A(n - 1, 1) & \text{pour } n \geq 1 \\ A(n, p) = A(n - 1, A(n, p - 1)) & \text{si } n \geq 1, p \geq 1 \end{cases}$$

1. Calculer $A(0, 0)$ puis $A(2, 2)$. Il est conseillé d’être astucieux.
2. Implémenter la fonction A en C. La signature sera `fct_A(int n, int p)`.

On vérifiera que les entrées n et p sont valides avec l’instruction `assert`.

Remarque : la fonction A (qui s’appelle en fait "fonction d’Ackermann") croît très rapidement. Par exemple, $A(4, 2) > 10^{10000}$.

Exercice 6 Un peu de morpion

Dans cet exercice on va écrire un petit programme C qui permet à deux joueurs de jouer au morpion au clavier.

On utilisera pas un tableau, mais plutôt 9 variables représentant les cases du plateau : $x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8$. Elles peuvent prendre trois valeurs : le caractère espace ' ' si la case est vide, 'o' si la case contient un rond et 'x' si la case contient une croix.

Par exemple la grille :

X		X
	O	O
		O

est représenté par $x_0 = 'x', x_1 = ' ', x_2 = 'x', x_3 = ' ', x_4 = 'o', x_5 = 'o', x_6 = ' ', x_7 = ' ', x_8 = 'o'$.

1. Écrire une fonction

`void affiche_grille(char x0, char x1, char x2, char x3, char x4, char x5, char x6, char x7, char x8)`
qui affiche la grille ainsi (toujours sur le même exemple) :

x				x
-	-	-	-	-
		o		o
-	-	-	-	-
				o

2. Écrire une fonction

`bool est_gagnante(char x0, char x1, char x2, char x3, char x4, char x5, char x6, char x7, char x8)` qui teste s'il y a un alignement de 3 ronds ou de 3 croix sur une ligne, une colonne ou une diagonale. La fonction renvoie **true** et affiche quel joueur (croix ou rond) a gagné si c'est le cas. Sinon la fonction renvoie **false**.

On propose la fonction suivante qui permet de demander à un utilisateur de rentrer le numéro d'une case et qui le renvoie :

```
int entre_coup(){
    int x;
    scanf("%d\n", &x);
    assert (x<=8);
    assert (x>=0);
    return x;
}
```

3. Pour mettre le jeu en place, on se place dans la fonction `main`. Il vous faudra une boucle qui, tant que le jeu n'est pas fini, demande le coup du prochain joueur. (en se souvenant de si c'est le tour des croix ou des ronds) Écrivez le code correspondant en utilisant les fonctions précédentes. On pourra éluder les lignes qui se répètent en indiquant la répétition par un commentaire.