

Partie 1 : Les bases

- Le but de cette première section est de se familiariser avec les bases du C.
- L'utilisation des fonctions déjà implémentées n'est pas autorisée !

Mesure de la longueur des chaînes

Exercice 1 :

Dans cet exercice, nous allons essayer d'implémenter une fonction qui mesure la longueur d'une chaîne terminée par null. Cette fonction se comportera comme la fonction `strlen` de la bibliothèque standard C. On vous rappelle que `strlen` prend un pointeur de char et renvoie un entier. Le prototype de `strlen` :

```
1 int strlen(char* str);
```

Implémentez une fonction appelée `len(str)` qui prend un pointeur de char comme argument et renvoie un entier positif. Vous ne devriez pas utiliser la fonction `strlen`.

It's all binary

Exercice 2.1 : Leading zeros

Implémentez une fonction appelée `clz(bits)`¹ qui prend un entier positif comme argument et compte le nombre de bits mis à zéro en allant de gauche à droite jusqu'à ce qu'il atteigne un 1 ou la fin. (Il compte le nombre de zéros avant la première occurrence d'un)

Exemples:

```
1 Pour 00000000 00000000 00000000 00010000 clz doit retourner : 27
2 Pour 00000001 00000000 00000000 00010000 clz doit retourner : 7
3 Pour 00000000 00001000 11111111 00010000 clz doit retourner : 12
4 Pour 00000000 00000001 11111111 00010000 clz doit retourner : 15
```

Exercice 2.2 : Nextpow2

Implémenter une fonction `nextpow2(n)` qui prend un entier positif et renvoie la puissance de 2 suivante la plus proche.

Par exemple:

```
1 nextpow2(1)      doit retourner 1
2 nextpow2(3)      doit retourner 4
3 nextpow2(5)      doit retourner 8
4 nextpow2(31)     doit retourner 32
5 nextpow2(120)    doit retourner 128
```

Exercice 2.3 : Log2

Implémenter une fonction `log2(n)` qui prend un entier positif et renvoie un entier positif qui représente l'arrondi du \log_2 de l'entrée. La fonction peut être décrite par la formule suivante : $\log_2(x) = \lceil x \rceil$.

Par exemple:

```
1 log2(1)          doit retourner 0
2 log2(3)          doit retourner 2
3 log2(5)          doit retourner 3
4 log2(31)         doit retourner 5
5 log2(120)        doit retourner 7
```

¹count leading zeros

PGCD

Exercice 3

Implémenter une fonction `gcd(a, b)` qui prend 2 entiers positifs et renvoie leur pgcd.
Par exemple:

```
1 gcd(1, 3)      doit retourner 1
2 gcd(3, 5)      doit retourner 1
3 gcd(6, 8)      doit retourner 2
4 gcd(8, 32)     doit retourner 8
5 gcd(30, 120)   doit retourner 30
```

Partie 2 :

- Le but de cette partie est de traiter les boucles et les conditions et de se familiariser avec la fonction de `printf`.

Pyramide en C

Exercice 4 :

Ecrivez un programme C qui affiche un pyramide de hauteur `n`. Pour ce faire, crée une fonction `pyramide` qui prend comme paramètre un entier `n` qui représente la hauteur de la pyramide et ne renvoie rien. L'argument `n` doit être passé à l'exécutable dans la ligne de commande. Exemples :

```
1 ./pyramide 1
2 *
```

```
1 ./pyramide 3
2 *
3 * *
4 * * *
```

```
1 ./pyramide 4
2 *
3 * *
4 * * *
5 * * * *
```

Triangle de Pascal

Exercice 5 :

Ecrivez un programme C, qui affiche un triangle de Pascale jusqu'au niveau `h`.

```
1 ./pascal 3
2 1
3 1 1
4 1 2 1
```

```
1 ./pascal 4
2 1
3 1 1
4 1 2 1
5 1 3 3 1
```

Partie 3 :

- Le but de cette partie est de manipuler les allocations dynamiques de mémoire.

Tableau de Tableau (Les Matrices)

Exercice 6 :

Il existe 3 façons différentes de représenter une matrice en C. Créez trois fonctions appelées respectivement `mat1`, `mat2` et `mat3` qui prend deux arguments `N` et `M` et créer une matrice de dimension `NxM`. `N` et `M` sont spécifiés à l'exécution via les arguments du programme.