

Syntaxe du C

Aide-mémoire

MP2I Lycée Pierre de Fermat

Instructions basiques

Déclaration de variable

```
1 float x;  
2 int n, m, i, j;  
3 /* Le type bool est un renommage du type int.  
4    Il faut la librairie stdbool.h pour l'utiliser. */  
5 bool t;
```

Définition de variable

```
1 x = 25.3;  
2 n = x + 2;  
3 t = true;  
4 // on peut mélanger déclaration et définition :  
5 bool b = (x > 20);  
6 float y = 0, z = 8.3;
```

Opérations arithmétiques

`+, x, *, -, %`

```
1 n = ((8+7)*((65-4)/3))%2;
```

Opérateurs de comparaison

`<, >, <=, >=, ==, !=`

```
1 int x = 5;  
2 int y = 11;  
3 bool b1 = x < y;  
4 bool b2 = x + 8 != y;
```

Opérateurs booléens

`!, &&, ||`

```
1 int x = 5;  
2 int y = 11;  
3 bool b1 = (x < y) && (y < 2 * x); // ET  
4 bool b2 = (x == y) || (x > y+5); // OU  
5 bool b3 = !b2; // NEGATION  
6  
7 // On peut tout combiner comme on veut :  
8 bool b4 = (b1 && !b2) || (x == 1) || !b3 ;
```

Affichage et saisie

Les fonctions d’affichage/saisie nécessitent l’usage de la librairie ***stdio.h***. Vous devez l’importer au début des programmes avec `#include <stdio.h>`.

Affichage

```
1 printf("Bonjour\n"); // \n est un retour à la ligne
2
3 int x = 85;
4 float z = -6.978;
5 printf("%f est un flottant, %d un entier\n", z, x);
6 printf("%d + 1 vaut %d\n", x, x+1);
7
8 // Une valeur sera affichée différemment selon le type qui est indiqué.
9 printf("%c est le %d-eme caractère.\n", x, x);
```

Saisie

```
1 int x, y;
2 float z;
3 scanf("%d %d", &x, &y);
4 scanf("%f", &z);
```

Temps et aléatoire

La fonction time : Elle renvoie le nombre de secondes écoulées depuis le 01/01/1970 à minuit.

```
1 #include <time.h>
2 int main(){
3     long int t = time(NULL);
4     printf("%ld\n", t);
5 }
```

Génération d’aléatoire : Initialisation avec `srand` une fois au début du main.

```
1 #include <time.h> // pour la fonction time
2 #include <stdlib.h> // pour les fonctions srand et rand
3 #include <stdio.h>
4
5 // renvoie un nombre aléatoire entre 10 et 20 inclus
6 int note(){
7     return 10 + rand()%11;
8 }
9
10 int main(){
11     srand(time(NULL)); // initialisation du générateur aléatoire
12     int x = note();
13     int y = note();
14     printf("%d, %d\n", x, y);
15 }
```

Conditions et boucles

If-else

```
1 int x = 3;
2 float y = 5.75;
3 if (x > y){
4     y = 2*y;
5 } else {
6     y = y/2;
7 }
8 // On peut faire un if sans else:
9 bool b = (x == 3);
10 if (b){
11     printf("Truc\n");
12 }
```

Boucles while

```
1 int i = 3, j = 985;
2 while (i != j && j > 0){
3     i++;
4     j--;
5 }
```

Boucles for

```
1 for (int i=7; i<98; i = i + 3){
2     print("%d\n", i);
3 }
4
5 /* Les boucles for peuvent être utilisées de très
6     nombreuses manières */
7 for (char c = 'a'; c <= 'z'; c++){
8     printf("%c", c);
9 }
10
11 /* En C, les boucles for sont aussi puissantes
12     que les boucles while (ce qui n'est pas le cas
13     en algorithmique. N'utilisez pas les boucles
14     for du C pour faire des boucles while !
15     Exemple à ne pas reproduire chez soi: */
16 for ( ; ; ){
17     // cette boucle tourne à l'infini sans rien faire
18 }
```

Fonctions

Déclaration de fonction

```
1 bool est_pair(int x);
2 float somme(float x, float y);
```

Définition de fonction

```
1 bool est_pair(int x);
2 est_pair(int x){
3     return (x%2 == 0);
4 }
5
6 // Comme pour les variables on peut fusionner déclaration et définition
7
8 float somme(float x, float y){
9     return x+y;
10 }
```

Appel de fonction Attention à bien vérifier le type des arguments!

```
1 float somme(float x, float y){
2     return x+y;
3 }
4
5 int main(){
6     float x = -5;
7     float y = 6.22;
8     float z = somme(x, y);
9 }
```

Assertions

```
1 #include <assert.h>
2
3 float inverse(float x){
4     assert (x != 0);
5     return 1/x;
6 }
7
8 int main(){
9     float x = 5, y = 0;
10    x = inverse(x); // pas de souci
11    y = inverse(y); // Le programme s'arrête
12
13    return 0;
14 }
```