# TP – Préparation Bilan de mi-semestre - SN1

## Passage d’arguments à la fonction main()

Rappel : la fonction *main()* est le point d’entrée du programme.

On peut donner en argument des valeurs au programme. Pour ce faire, il faut lancer le programme dans une console DOS (cmd) en faisant suivre le nom du programme par les arguments séparés par des espaces. Chaque argument est vu par la fonction *main()* comme étant des chaînes de caractères.

Un autre argument est confié à la fonction *main()* est le nombre d’arguments. Attention, ce nombre tient compte du nom du programme. Ce nombre n’est pas saisi, c’est l’interpréteur de commandes qui compte le nombre d’arguments (programme compris) et le confie à la fonction *main()*

Explications :

Par exemple, notre programme s’appelle « prog.exe ». on veut lui « passer » les valeurs 1 et 2. On tapera alors :



La fonction *main()* du programme *prog.exe* est définie comme telle :

#include <iostream>

using namespace std;

int **main**(int argc, char \*argv[])

{

 cout << "il y a " << argc-1 << " arguments passes a " << argv[0] << endl;

 for (int i = 1 ; i < argc ; i++)

 {

 cout << argv[i] << endl;

 }

 return 0;

}

* Résultat de l’exécution :



On peut constater que les valeurs 1 et 2 sont des chaînes de caractères tout comme le nom du programme.

### Pourquoi déclare-t-on la variable contenant les arguments sous la forme : char \*argv[] ?

Finalement, argv[] est équivalent à \*argv

Donc char \*argv[] est équivalent à char \*\*argv

C’est donc un tableau de chaîne de caractères !

|  |  |  |  |
| --- | --- | --- | --- |
| *argv->* | *0* | *1* | *2* |
| 0 | ‘p’ | ‘1’ | ‘2’ |
| 1 | ‘r’ | **‘\0’** | **‘\0’** |
| 2 | ‘o’ |  |  |
| 3 | ‘g’ |  |  |
| 4 | ‘ .’ |  |  |
| 5 | ‘e’ |  |  |
| 6 | ‘x’ |  |  |
| 7 | ‘e’ |  |  |
| 8 | **‘\0’** |  |  |

Si on accède à la case *argv[0][1]* par exemple, on accède en fait au caractère ‘r’ de « prog.exe »

Si on accède à la case *argv[2][0]*, on accède au caractère ‘2’ de l’argument « 2 »