La ligne de commande

1. Définition d'une fonction[1]

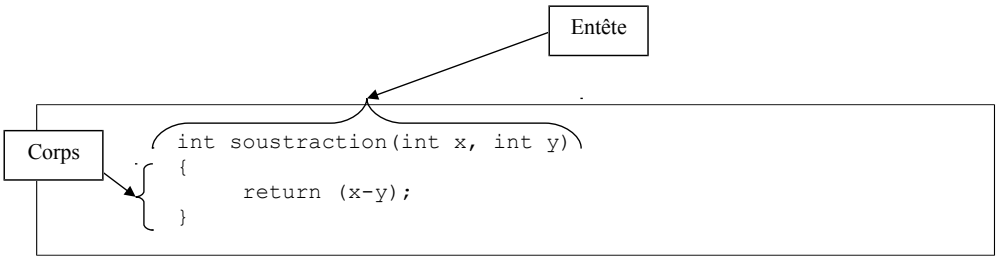
La définition se fait comme suit :

```
type identificateur (paramètres) {
    ... /* Instructions de la fonction. */
}
```

- « type » est le type de la valeur renvoyée.
- « identificateur » est le nom de la fonction.
- « paramètres » est une liste de paramètres.

1 Pour plus de détails, voir le chapitre 7 des notes de cours IFT1166 disponible à cette adresse : http://www.iro.umontreal.ca/~pift1166/cours/ift1166/communs/Cours/notes_de_cours.html

Une définition = entête + corps



- Une fonction en C++ peut être définie globalement dans une autre fonction ou dans une classe. Dans ce dernier cas, la fonction porte le nom de « méthode ».
- Il n'y a pas de fonctions en Java seulement des méthodes.

2. Fonction « main »

- La fonction « main » est une fonction ordinaire quelconque avec quelques petites particularités propres à elle.
- Cette fonction constitue le point d'entrée du programme.
- Quand un programme exécutable écrit en C++ est lancé, la première chose que ce programme va chercher à exécuter est la fonction « main ».
- Prenons l'exemple de cette commande :

```
test.exe fichier.txt 3 c
```

- Quand on exécute le programme « text.exe », ce dernier va exécuter le contenu de la fonction « main » codée dans le programme C++, « test.cpp ».
- En plus de cela, il va transmettre à la fonction « main » les éléments supplémentaires fournis sur la ligne de commande.
- Dans cet exemple, les éléments supplémentaires sont : « fichier.txt 3 c ».

- La fonction « main » est définie comme suit :

```
int main(int argc, char*argv[]) ;
```

- La fonction « main » est une fonction quelconque qui retourne un entier et accepte deux arguments.
- L'entier retourné permet de donner une indication sur l'état d'exécution du programme. Si la fonction retourne « 0 », l'exécution du programme s'est déroulée sans erreur. Dans le cas contraire, le programme a probablement planté pour une raison à déterminer.
- Les arguments de la fonction « main » représentent ce qui suit :
  - La variable « argc » contient le nombre d'arguments sur la ligne de commande.
  - La variable « argv » est un tableau de pointeurs[2] sur les arguments de la ligne de commande.

2 Pour plus de détails, voir le chapitre 8 des notes de cours IFT1166 disponible à cette adresse : http://www.iro.umontreal.ca/~pift1166/cours/ift1166/communs/Cours/notes_de_cours.html

- Pour l'appel « `test.exe fichier.txt 3 c` » :
  - Dans cet exemple, l'argument entier « argc » vaut « 4 ». Cette valeur représente le nombre d'éléments sur la ligne de commande, incluant le nom du programme.
  - L'argument « argv » va contenir la ligne de commande découpée en chaînes de caractères {" test.exe ", " fichier.txt ", " 3 " , " c "}.
  - En C++, l'argument « 0 » est le nom du programme exécutable.

Indice de l'argument	Argument	argv[ ]
0	`test.exe`	`argv[0]`
1	`fichier.txt`	`argv[1]`
2	`3`	`argv[2]`
3	`c`	`argv[3]`

- Les éléments sont transmis par le système d'exploitation (Linux, Mac ou Windows) à votre programme de manière transparente. Ceci ne doit pas constituer une préoccupation pour vous. Voyez cette transmission comme un appel ordinaire à une fonction à deux arguments :

```
int exemple (int vv, char** ww) {// une fonction quelconque};
int zz = 4 ;
char* xx = {" test.exe ", " fichier.txt ", " 3 ", " c "} ;
int aa = 0 ;
aa = exemple (zz, xx) ;
```

- Par convention, pour désigner une option, on utilise la convention « -lettre ».
- Pour l'appel « test.exe –h –e=1 » :

```
argc=3
```

Indice de l'argument	Argument	argv[ ]
0	test.exe	argv[0]
1	-h	argv[1]
2	-e=1	argv[2]

- On doit faire une analyse syntaxique pour extraire des informations pertinentes de la ligne de commande.
- Nous devons par exemple analyser « argv [2] » caractère par caractère, afin d'arriver à la conclusion, c'est une option qui a été choisie « - ». Elle porte le nom de « e » et elle a une valeur « 1 ».
- Les langages C/C++ fournissent la fonction « getopt » qui permet d'effectuer une analyse lexicale de la ligne de commande.
- Pour pouvoir utiliser les fonctionnalités de « getopt », il faudra inclure le fichier d'en-tête :

```
#include<getopt.h>
```


- La syntaxe de la fonction « getopt » est :

```
int getopt(int argc, char **argv, char *optstring) ;
```

- Il faudra définir dans le programme les variables suivantes :

```
extern char *optarg ;

extern int optind, opterr;
```

- La fonction « getopt » retourne la lettre de l'option courante de la ligne de commande qui correspond à l'une des lettres d'option présente dans la chaîne « optstring ».
- La chaîne de référence « optstring » doit contenir l'ensemble des lettres des options possibles.
- Si une lettre d'option est suivie d'un caractère « : », cette option annonce un ou plusieurs arguments séparés par un espace.
- La variable « optarg » est alors préinitialisée pour pointer sur le début de l'argument de l'option.
- La variable « optind » est l'index de la chaîne courante. Cet index est compris entre « 0 » et « argc ».