La ligne de commande

Programmation séquentielle en C, 2020-2021

Orestis Malaspinas (A401), ISC, HEPIA

2019-10-16

Inspirés des slides de F. Glück

Point d'entrée d'un programme

- Le point d'entrée est la fonction `main()`.
- Elle peut être déclarée de 4 façon différentes:
  1. `void main().`
  2. `int main().`
  3. `void main(int argc, char **argv).`
  4. `int main(int argc, char **argv).`
- `argc` est le nombre d'arguments passés à la ligne de commande: **le premier est celui du programme lui-même**.
- `argv` est un tableau de chaînes de caractères passés sur la ligne de commande.

Exemple d'utilisation

Pour la fonction dans le programme `prog`

```
int main(int argc, char **argv)
```

Pour l'exécution suivante on a

```
$ ./prog -b 50 file.txt
```

```
argc == 4
argv[0] == "prog"
argv[1] == "-b"
argv[2] == "50"
argv[3] == "file.txt"
```

Conversion des arguments

- Les arguments sont toujours stockés comme des **chaînes de caractère**.
- Peu pratique si on veut manipuler des valeurs numériques.
- Fonctions pour faire des conversions:

```
int atoi(const char *nptr);
long atol(const char *nptr);
long long atoll(const char *nptr);
double atof(const char *nptr);
int snprintf(char *str, size_t size,
             const char *format, ...);
// str: buffer, size: taille en octets max à copier,
// format: cf printf(), ret: nombre de char lus
```

Exemple d'utilisation

```c
#include <stdio.h>
#include <stdlib.h>
#include <libgen.h>

int main(int argc, char **argv) {
    if (argc != 3) {
        char *progname = basename(argv[0]);
        fprintf(stderr, "usage: %s name age\n", progname);
        return EXIT_FAILURE;
    }

    char *name = argv[1];
    int age = atoi(argv[2]);

    printf("Hello %s, you are %d years old.\n", name, age);
    return EXIT_SUCCESS;
}
```

```
$ ./prog Paul 29
Hello Paul, you are 29 years old.
```