Arguments de la ligne de commande en C

Recevoir les arguments

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

Cette forme de `main` permet de recevoir les arguments fournis au lancement du programme. La déclaration `char *argv[]` est ici équivalente à `char **argv`.

argc

Entier donnant le nombre de chaînes dans `argv`, nom du programme compris. Vérifier sa valeur avant d'accéder aux arguments attendus.

argv

Tableau de pointeurs vers des chaînes de caractères. `argv[0]` contient généralement le nom utilisé pour lancer le programme ; `argv[1]` à `argv[argc - 1]` contiennent les arguments fournis. `argv[argc]` vaut `NULL`.

Par exemple, avec la commande `./produit 10 3` :

```
argc == 3
argv[0] : "./produit"
argv[1] : "10"
argv[2] : "3"
argv[3] : NULL
```

Les arguments numériques sont eux aussi transmis sous forme de chaînes.

atoi : convertir une chaîne en entier

Fonction déclarée dans `<stdlib.h>` :

```
int atoi(const char *s);
```

Convertit le début de la chaîne en une valeur de type `int`. Par exemple, `atoi("123")` renvoie 123, `atoi("12abc")` renvoie 12 et `atoi("abc")` renvoie 0. Elle ne permet pas de distinguer une conversion impossible de celle de `"0"`. On suppose la valeur représentable par le type `int`.

Exemple : produit de deux entiers

On suppose les arguments valides et les deux entiers ainsi que leur produit représentables par `int`.

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

int main(int argc, char *argv[])
{
    if (argc != 3) {
        fprintf(stderr, "Usage : %s entier1 entier2\n", argv[0]);
        return EXIT_FAILURE;
    }
    int a = atoi(argv[1]);
    int b = atoi(argv[2]);
    printf("%d * %d = %d\n", a, b, a * b);
    return EXIT_SUCCESS;
}
```