### Créer un exécutable qui prend des arguments en ligne de commande en C

Il est possible de passer des paramètres à un programme C compilé `prgm` au moment où on lance son exécution en ligne de commande, à condition d'avoir déclaré dans le code source du programme la fonction `main` avec la signature suivante `int main(int argc, char* argv[]);`
**Attention :** , contrairement à ce qu'on peut faire avec les arguments d'une fonction, les paramètres d'un programme tous des chaînes de caractères.
Au moment de l'exécution du programme, la fonction `main` est alors appelée avec des valeurs d'arguments qui dépendent de la ligne de commande

- `argc` a pour valeur le nombre de mots constituant la ligne de commande, y compris la commande elle-même, c'est donc le nombre de paramètres réellement souhaités plus 1
- `argv` est un tableau de `argc` chaînes de caractères initialisées selon les mots de la ligne de commande, en particulier `argv[0]` est le nom de la commande (donc rarement utile).

**Exemples**

• La ligne de commande `./prog a 12`, `argc` vaut 3 (la commande + les 2 paramètres), et `argv[0]` pointe vers la chaîne `"./prog"`, `argv[1]` vers `"a"` et `argv[2]` vers `"12"`. Pour récupérer la valeur d'un entier à partir d'une chaîne de carcatères contenant son écriture décimale, on peut utiliser la fonction `atoi` de la librairie standard.

• Si `add` est le fichier obtenu en compilant le code ci-contre, alors `./add 12 24` affiche `12 + 24 = 36`, mais `./add 12 24 36` affiche `on attend 2 entiers` avant l'erreur d'assertion.

**Exercice** Créer en C un programme `echo` qui affiche la ligne de commande qui l'a lancé, hormis le premier mot `./echo`.

```c
#include <assert.h>
#include <stdbool.h>
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char* argv[]){
  int nb_arg_attendu = 2;
  if( argc != nb_arg_attendu +1){
    printf("on attend 2 entiers\n");
    assert(false);
  }
  int a = atoi(argv[1]);
  int b = atoi(argv[2]);

  printf("%d + %d = %d\n", a, b,
  ↪  a+b);

  return 0;
}
```