```c
// ce programme (composé d'une fonction main)
// affiche les arguments reçus en ligne de commande
// (un par ligne et dans l'ordre inverse
// de ceux reçus en ligne de commande)
// compilé avec gcc -Wall -Wextra -Werror ft_print_program_name.c
// puis exécuté avec ./a.out test1 test2 test3
// doit donc afficher test3 test2 test1 à l'écran
// (mais exécuté avec ./a.out seulement,
// ne doit rien afficher)
// tous les arguments doivent être affichés, sauf argv[0]

// pour pouvoir utiliser la fonction write
#include <unistd.h>

// fonction pour afficher la chaine de caractères
// d'un argument reçu en ligne de commande
// (voir ex01)
void    ft_write_argv(char *str);

// voir ex00
int main(int argc, char **argv)
{       // on déclare un entier i,
        // compteur du nombre d'arguments
        // passés en ligne de commande
        int     i;

        // on initialise i à argc - 1
        // (pour qu'il corresponde à l'indice du dernier
        // argument passé en ligne de commande)
        // ATTENTION :
        // argc correspond au nombre d'arguments
        // en comptant le nom du programme
        // donc au nombre d'arguments passés
        // en ligne de commande - 1
        // EXEMPLE AVEC 3 ARGUMENTS :
        // ARGC VAUDRA 4
        // DONC avec i = argc - 1
        // i vaudra 3
```

```
// et comme le premier argument (argv[0])
// correspond au nom du programme
// et que l'on veut afficher uniquement
// les arguments passés en ligne de commande
// le dernier indice (du 3ème argument) sera 3
// (argv[3] correspondra au 3ème argument
// (donc bien au dernier argument)
// passé en ligne de commande

// ainsi, i sera initialisé à l'indice
// du pointeur vers le dernier argument
// passé en ligne de commande
// (il sera décrémenté jusqu'à 1 ensuite
// pour parcourir les pointeurs vers l'adresse
// des arguments dans l'ordre inverse (voir plus bas))

// EXEMPLE AVEC 3 ARGUMENTS, test1 test2 et test3 :
// argc = 4
// i = argc - 1
// i = 4 - 1
// i = 3
// i > 0 ? OUI
// 1)
// on affiche la chaine de caractères pointée argv[3]
// (celle du troisième paramètre passé en ligne de commandes)
// test3
// i--
// i = 3 - 1
// i = 2
// i < 0 ? OUI
// 2)
// on affiche la chaine de caractères pointée argv[2]
// (celle du deuxième paramètre passé en ligne de commandes)
// test2
// i--
// i = 2 - 1
// i = 1
// 1 < 0 ? OUI
```

```
	// 3)
	// on affiche la chaine de caractères pointée argv[1]
	// (celle du premier paramètre passé en ligne de commandes)
	// test1
	// i--
	// i = 1 - 1
	// i = 0
	// 0 < 0 ? NON
	// FIN DE LA BOUCLE

	i = argc - 1;

	// tant que i est supérieur à 0
	// (car on ne veut pas afficher le nom du programme,
	// argv à l'indice 0)
	while (i > 0)
	{
		// on affiche la chaine de caractères en cours à l'écran
		ft_write_argv(argv[i]);

		// une fois la chaine de caractères affichée à l'écran
		// on saute une ligne
		write(1, "\n", 1);

		// on décrémente i de 1
		i--;
	}

	// on retourne 0 pour indiquer que le programme s'est bien exécuté
	return (0);
}

// pour écrire la chaine de caractères à l'écran
// argv[3] par exemple
// voir ex01
void	ft_write_argv(char *str)
{
	// voir ex01
```

```
        // on parcourt la chaine de caractères
        while (*str != '\0')
                // on écrit le caractère en cours (pointé par str)
                // puis on incrémente str
                write(1, str++, 1);
}
```