```c
// la fonction ft_rev_int_tab prend en paramètres
// un pointeur vers un tableau d'entiers (tab)
// et la taille de ce tableau
// (son nombre d'éléments) (size)

void    ft_rev_int_tab(int *tab, int size)
{
        // on déclare start
        // qui correspondra à l'indice
        // du tableau en partant du début
        // de celui-ci
        // il sera initialisé à 0
        // puis il sera incrémenté
        // pour parcourir le tableau
        // du début vers la fin
        int     start;

        // on déclare end
        // qui correspondra à l'indice
        // du tableau en partant de la fin
        // de celui-ci
        // il sera initialisé à size - 1
        // (ce qui correspond à l'indice du
        // dernier élément du tableau)
        // (car les indices commencent à 0 en C)
        // puis il sera décrémenté
        // pour parcourir le tableau
        // de la fin vers le début
        int     end;

        // variable temporaire pour stocker
        // la valeur de l'élément actuel du tableau
        // à l'indice donné
        // avant l'échange
        int     temp;

        // on initialise start à 0
        start = 0;
```

```
// on initialise end à size - 1
end = size - 1;

// tant que start est inférieur à end :
// comme start sera incrémenté de 1
// et end sera décrémenté de 1
// à la fin de chaque tour de boucle
// la boucle continuera
// jusqu'à ce qu'elle atteigne (ou dépasse)
// le milieu du tableau
// (lorsque les indices seront égaux
// ou lorsque start deviendra supérieur à end
// (croisement des indices))
while (start < end)
{
        // on stocke la valeur de l'élément actuel
        // du début du tableau dans temp
        temp = tab[start];

        // l'élément actuel du début du tableau
        // prend la valeur de l'élément actuel
        // de la fin du tableau
        // (l'élément de fin du tableau
        // écrase celui du début du tableau)
        tab[start] = tab[end];

        // l'élément actuel de la fin du tableau
        // prend la valeur stockée temporairement dans temp
        // (qui est l'ancienne valeur de l'élément du début du tableau)
        // (l'élément de fin du tableau
        // est écrasé par celui du début du tableau
        // (auparavant stocké dans temp))
        tab[end] = temp;

        // on incrémente start
        // pour que l'indice de début
        // se déplace vers la droite du tableau
```

```
                start++;

                // on décrémente end
                // pour que l'indice de fin
                // se déplace vers la gauche du tableau
                end--;
        }
}

// main.c :

#include "ft_rev_int_tab.h"
#include "ft_putnbr.h"
#include <unistd.h>

void    ft_fill_tab(int *tab, int size);
void    ft_print_tab(int *tab, int size);

int     main(void)
{

        // on déclare un tableau de 10 entiers
        int     tab[10];

        // on déclare un entier size, qui contiendra la taille du tableau
        int     size;

        // on initialise size à 10
        size = 10;

        // on remplit le tableau tab avec des entiers de 0
        // à size -1 (voir plus bas)
        // donc de 0 à 9
        ft_fill_tab(tab, size);

        // on affiche les éléments du tableau
        // après son remplissage
        ft_print_tab(tab, size);
```

```
	// on saute une ligne
	write(1, "\n", 1);

	// on inverse l'ordre des éléments du tableau
	ft_rev_int_tab(tab, size);

	// on affiche les éléments du tableau
	// après leur inversion
	ft_print_tab(tab, size);
	write(1, "\n", 1);
	return (0);
}

void	ft_fill_tab(int *tab, int size)
{
	int	i;

	i = 0;

	// on remplit le tableau avec des valeurs croissantes
	// (incrémentation de i) à partir de 0
	// jusqu'à 9
	while (i < size)
	{
		tab[i] = i;
		i++;
	}
}

void	ft_print_tab(int *tab, int size)
{
	int	i;

	i = 0;

	// on parcourt le tableau et on affiche chacun de ses éléments
	while (i < size)
```

```
        {
                ft_putnbr(tab[i]);
                i++;
        }
}

// RESULTAT :
// 0 1 2 3 4 5 6 7 8 9
// 9 8 7 6 5 4 3 2 1 0
```