```
// ATTENTION :
// pour utiliser la fonction strlcpy originale :
// - installer libbsd-dev avec la commande suivante :
// sudo apt-get install libbsd-dev
// - inclure le fichier d'entête bsd string.h :
// #include <bsd/string.h>
// - compiler avec gcc en utilisant le flag -o au début et le flag -lbsd à la fin
// exemple :
// gcc -o a.out ft_strlcpy.c main.c -lbsd

// cette fonction copie une chaine de caractères de la source (src)
// vers la destination (dest)
// en ne dépassant pas la taille maximale spécifiée (size)
// attention : size doit prendre en compte le caractère de fin de chaine '\0' !

// la valeur de retour est la longueur totale de la chaine source (src_size)
// cela permet de savoir si la chaine src a été tronquée (en la comparant à size)
// (par exemple, si size est égal à 5 et que la valeur de retour est égale à 10
// alors la chaine src a été tronquée (de 5 caractères))
// si size < src_size : la chaine a été tronquée
// si size >= src_size : la chaine n'a pas été tronquée

// la valeur de retour est un unsigned int
// car elle ne peut pas être négative (c'est une taille)

// Remarque : pour comparer 2 variables, il faut qu'elles soient exactement du même type !
// il est donc utile de déclarer la valeur de retour comme un unsigned int
// car size est aussi de type unsigned int
// elles pourront ainsi être comparés plus facilement dans la fonction appelante

// dest et src sont des pointeurs vers des chaines de caractères
// (ou plutôt vers des tableaux de caractères, dest étant déclaré mais vide
// et src comportant des caractères)

unsigned int    ft_strlcpy(char *dest, char *src, unsigned int size)
{
        // compteur pour suivre la position actuelle dans la chaine source
        // et celle de destination lors de la copie
```

```
// (lecture de src et écriture dans dest)
// il est de type unsigned car il ne peut pas être négatif
// et aussi car il sera comparé à size
// qui est aussi de type unsigned int
unsigned int    i;

// variable dont la valeur sera retournée
// par la fonction
// elle est utilisée pour calculer la longueur de la chaine source
unsigned int    src_size;

// on initialise src_size 0
// il sera incrémenté de 1 pour chaque caractère de src
// qui n'est pas le caractère de fin de chaine
// src_size sert aussi d'indice pour parcourir le tableau
// de caractères (à partir de l'indice 0)
src_size = 0;

// tant que le caractère du tableau src à l'indice src_size
// n'est pas \0
while (src[src_size] != '\0')
        // on incrémente le compteur de caractères
        // ATTENTION : une fois le dernier caractère atteint
        // src_size sera aussi incrémenté !
        // src_size correspondra donc au nombre de caractères
        // trouvés avant '\0' + 1
        // ce qui correspond à la taille totale de src
        // en comptant le caractère '\0'
        src_size++;

// si la taille fournie à la fonction est 0
if (size == 0)
        // on retourne directement src_size
        // sans effectuer de copie
        // (car une size de 0 signifie
        // que 0 caractères maximum doivent être copiés)
        return (src_size);
```

```
        // on initialise i à 0
        i = 0;
        // tant que le caractère nul n'est pas
        // rencontré dans src
        // et que la limite size - 1 n'est pas atteinte
        // (car il faut laisser une place pour '\0' à la fin
        // qui est compté dans size)
        while (src[i] != '\0' && i < size - 1)
        {
                // on copie le caractère à l'indice en cours
                // dans src vers l'emplacement de dest à l'indice en cours
                dest[i] = src[i];

                // on incrémente i
                i++;
        }
        // on ajoute le caractère de fin de chaine à la fin de dest
        // (i vient d'être incrémenté après la copie du dernier caractère)
        dest[i] = '\0';

        // on retourne src_size
        return (src_size);
}

#include "ft_strcpy.h"
#include "ft_putstr.h"
#include "ft_strlcpy.h"
#include "ft_putnbr.h"
#include <unistd.h>
#include <bsd/string.h>

int     main(void)
{
        // on déclare le tableau de caractères src
        // qui contiendra la chaine de caractères source
        // il pourra contenir 13 caractères
        // + le caractère de fin de chaine
        char    src[14];
```

```
// on déclare le tableau de caractères dest1
// qui contiendra la chaine de caractères de destination
// résultant de l'exécution de strlcpy
// il pourra contenir 5 caractères
// + le caractère de fin de chaine
char    dest1[6];

// on déclare le tableau de caractères dest2
// qui contiendra la chaine de caractères de destination
// résultant de l'exécution de ft_strlcpy
// il pourra contenir 5 caractères
// + le caractère de fin de chaine
char            dest2[6];

// taille maximale de copie
// utilisé pour strlcpy et ft_strlcpy
unsigned int    size;

// on définit la taille maximale de la copie à 6
// même taille que dest1 et dest2
// (5 caractères + le caractère de fin de chaine)
size = 6;

// on crée la chaine source Hello World !
ft_strcpy(src, "Hello World !");

// on l'affiche
ft_putstr(src);

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

// on copie la chaine src dans dest1, en lui passant size
// avec la VRAIE fonction C strlcpy
ft_putnbr(strlcpy(dest1, src, size));

// on saute une ligne
```

```
        write(1, "\n", 1);

        // on affiche le résultat de la copie, dest1
        ft_putstr(dest1);

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

        // on copie la chaine src dans dest2, en lui passant size
        // avec NOTRE fonction ft_strlcpy
        ft_putnbr(ft_strlcpy(dest2, src, size));

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

        // on affiche le résultat de la copie, dest2
        ft_putstr(dest2);

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

        // fin du programme
        return (0);
}

// compiler avec :
// gcc -o a.out ft_strcpy.c ft_putstr.c ft_putnbr.c ft_strlcpy.c main.c -lbsd
```