ft_putnbr.c

Cette fonction veut permettre d’afficher un nombre sur la sortie standard (pouvoir le write).

```
#include <unistd.h>

void    ft_putchar(char c)
{
    write(1, &c, 1);
/// write(0 = input / 1 = output / 2 = error, *Buffer, byte)
}

void    ft_putnbr(int n)
{
    long    nb; // <- il est plus grand qu'un int
// attention cependant, parfois il est nécessaire d'utiliser
// et de traiter le int min comme un cas particulier
/* Si on travail avec des int on doit faire le cas int min
if (n == -2147483648)
    write (1, "-2147483648", 11);
pourquoi ça ne marche pas directement avec un int ?
Parce qu'on converti en positif et l'int max c'est 2147483647
*/

    nb = n;
    if (nb < 0)
    {
        nb = -nb;// ici si on a un int min en passant positif
        write(1, "-", 1);
    }
    if (nb < 10) // tout ce qui est compris entre 0 et 9
        ft_putchar(nb + 48); // On converti en ascii 0 + 48 =
    else
    {
        // On peut convertir un chiffre directement mais pas
```

```
        // Donc on doit le traiter avant pour n'avoir que des
        // Un entier ne prends jamais de virgule. Et sur une
        // Résultat est toujours la dizaine inférieure avec l
        // Si on fait l'oppération un cas concret comme 42 :
        ft_putnbr(nb / 10);
        // 42 / 10 = 4,2 => donc 4
        ft_putnbr(nb % 10);
        // 42 % 10 = 2
    }
}
```

Note : ce code utilise la “récursivité”. Donc quand on veut faire une boucle dans notre fonction, on renvoie à la fonction une valeur. Dans le cas des opérations, on relance la fonction avec les résultats.

Avec l’exemple de 42, la première fois la fonction reçoit : ft_putnbr(42)

Une fois passée une première fois dans nos conditions, elle reçoit :

- ft_putnbr(4)
- ft_putnbr(2)

étant inférieurs à 10, ils sont écrits en passant dans ce bout de code :

```
    if (nb < 10)
        ft_putchar(nb + 48);
```

si le nombre avait été plus grand comme 420, le nombre aurait continué de tourner jusqu’à pouvoir revenir toujours à la partie qui écrit.

Cas concret d’utilisation d’un ft_putnbr bien comprit :

```
/*
on affiche tous les nombres de 1 à 100
Si le nombre est divisible par 3 on affiche fizz
Si le nombre est divisible par 5 on affiche buzz
Sinon on affiche le nombre
*/
#include <unistd.h>
```

```
void ft_putchar(char c)
{
    write(1, &c, 1);
}

void ft_putnbr(int n)
{
    if (n < 10)
        ft_putchar(n + 48);
    if (n > 9)
    {
        ft_putnbr(n / 10);
        ft_putnbr(n % 10);
    }
}

int fizzbuzz(int n)
{
    if (n <= 100)
    {
        if (n % 3 == 0)
                write(1, "fizz", 4);
        if (n % 5 == 0)
            write(1, "buzz", 4);
        if ((n % 3 != 0) && (n % 5 != 0))
            ft_putnbr(n);
        write(1, "\n", 1);
    }
    else
        return (0);
    return fizzbuzz(n+1);
}

int main(void)
{
    fizzbuzz(1);
    return (0);
}
```