```c
// la fonction ft_div_mod prend en argument
// deux entiers (a et b)
// et deux pointeurs vers des entiers (div et mod)
// qui stockeront respectivement
// le quotient de la division entière de a par b
// et le reste de la division entière de a par b

void    ft_div_mod(int a, int b, int *div, int *mod)
{
        // on divise a par b (avec l'opérateur /)
        // (division entière, des 2 entiers
        // car a et b sont de type int)
        // et on stocke le résultat (le quotient)
        // dans la variable
        // pointée par div
        *div = a / b;

        // on calcule le reste de la division entière
        // de a par b (le modulo de a par b)
        // (avec l'opérateur %)
        // (division entière, des 2 entiers
        // car a et b sont de type int)
        // et on stocke le résultat (le reste)
        // dans la variable
        // pointée par mod
        *mod = a % b;
}

// main.c :

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

int     main(void)
{
        // on déclare 4 variables entières
        // i et j stockeront les nombres à diviser
```

```
// d stockera le résultat de la division
// de i par j
// m stockera le reste de la division
// de i par j

int     i;
int     j;
int     d;
int     m;

// on initalise i à 14
i = 14;

// et j à 8
j = 8;

// on exécute notre fonction
// en lui passant les deux nombres à diviser
// (14 et 8)
// et les adresses de d et de m
// où seront stockés le quotient et le reste
// de la division
// après cet appel, d et m contiendront
// les résultats du calcul
ft_div_mod(i, j, &d, &m);

// on affiche i
// résultat : 14
ft_putnbr(i);
write(1, " ", 1);

// on affiche j
// résultat : 8
ft_putnbr(j);

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

```
        // on affiche d
        // résultat : 1
        ft_putnbr(d);
        write(1, " ", 1);

        // on affiche m
        // résultat : 6
        ft_putnbr(m);
        write(1, "\n", 1);
        return (0);
}

// RESULTAT :
// 14 8
// 1 6
```