TP 3

Les fichiers doivent être rendu par email à l'adresse suivante
herve.chaminaud@univ-tours.fr

Votre code doit être formaté pour être lisible (VsCode format document)
Pas de warning avec l'option de compilation `-Wall`

Dans tous les exercice vous devez traiter les cas d'erreurs éventuels. (argument erroné, argument vide etc

Exercice 1 (8 points)

Notes:

- Créer un fichier ex1.c

Réaliser un programme qui récupère les arguments en paramètre et les affiches à l'envers

Exemple:

```
$ ./a.out abcd toto machine
enihcam
otot
dcba
```

Exercice 2 (6 points)

Les programmes prennent 2 paramètres. Le premier est un nombre qui correspond a une valeure entière. Le deuxième paramètre est un entier qui correspond à un numéro de bit. (indexé à partir de zéro)
Ecrire un programme pour lire la valeur du bit correspondant au deuxième argument.
Ecrire un programme qui met la valeur du bit correspondant au 2 ème argument à 1.
Ecrire un programme qui met la valeur du bit correspondant au 2 ème argument à 0.

Notes:

- Créer 3 fichiers ex2_rd.c, ex2_set.c, ex2_reset.c

```
$ ./rd_bit  42 3
1

$ ./set_bit  42 0
43

$ ./reset_bit 42 5
10
```

Exercice 3 (3 points)

Afficher la valeur binaire d'un entier.
Ecrire un programme qui prend en paramètre un entier et affiche la valeur binaire. Le bit de poid faible doit-être à droite. En cas d'erreur le programme retourne -1

```
$ ./ex3 42
101010
```

Exercice 4 (3 points)

On define la st

```
typedef struct s_complex {
     double r
     double i
} t_complex;
```

Ecrire un programme qui créer 2 structures de nombres complexes, les valeurs sont laissée à votre discrétion.

Ecrire une fonction c_add qui réalise l'addition 2 structures complexe et qui crée une nouvelle structure pour le résultat.

```
t_complex *c_add(t_complex *op1, t_complex *op2);
```

Le programme doit afficher le résultat de l'addition.