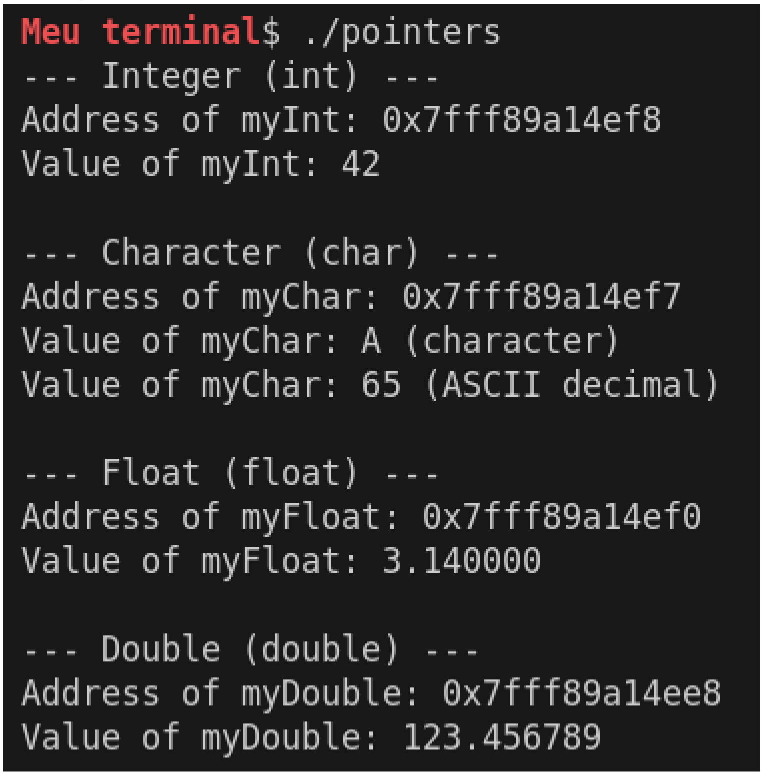
Exercise 01

Write a program that creates an `int` variable and prints its memory address using the pointer format.
Then, print its value in decimal format.

Test with other types to see what happens.

Example:



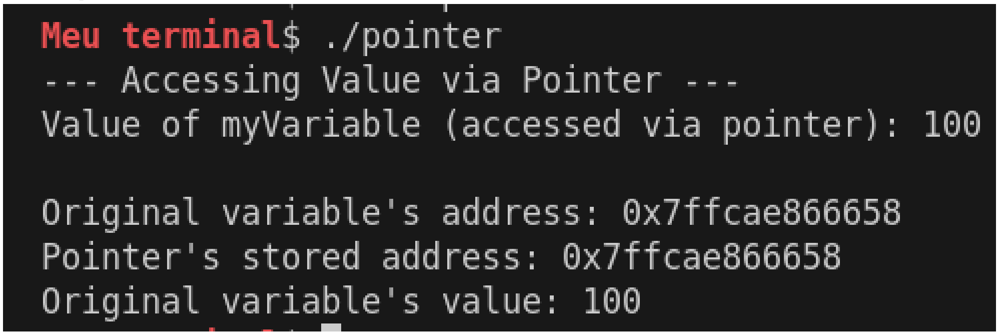
```
Meu terminal$ ./pointers
--- Integer (int) ---
Address of myInt: 0x7fff89a14ef8
Value of myInt: 42

--- Character (char) ---
Address of myChar: 0x7fff89a14ef7
Value of myChar: A (character)
Value of myChar: 65 (ASCII decimal)

--- Float (float) ---
Address of myFloat: 0x7fff89a14ef0
Value of myFloat: 3.140000

--- Double (double) ---
Address of myDouble: 0x7fff89a14ee8
Value of myDouble: 123.456789
```

Allowed functions: printf

Exercise 02

Create a program having a var with the type you want, insert a value in it. Then, create a pointer with the same type that receives the address of the var and print the value of the var using only the *.

Example:



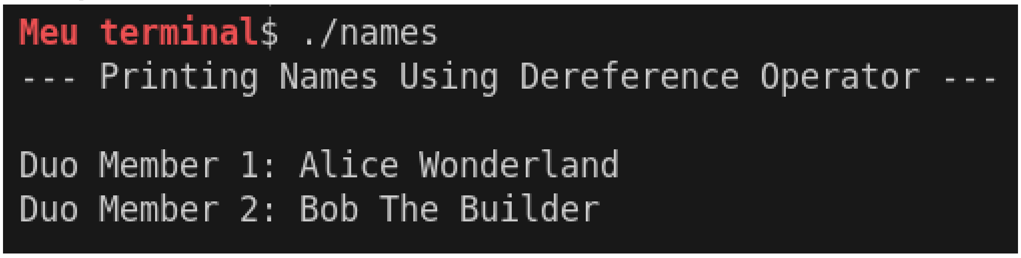
```
Meu terminal$ ./pointer
--- Accessing Value via Pointer ---
Value of myVariable (accessed via pointer): 100

Original variable's address: 0x7ffcae866658
Pointer's stored address: 0x7ffcae866658
Original variable's value: 100
```

Allowed functions: printf

Exercise 03

Create a program that will have 2 arrays, each containing the full name of a member of the duo. Print each name using only the dereference operator (*).

Example:



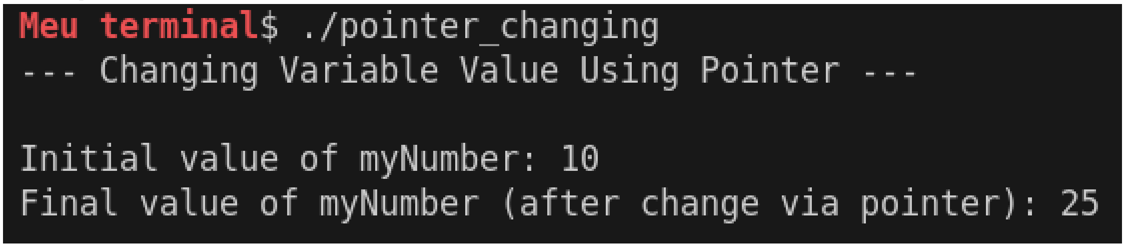
```
Meu terminal$ ./names
--- Printing Names Using Dereference Operator ---

Duo Member 1: Alice Wonderland
Duo Member 2: Bob The Builder
```

Allowed functions: write

Exercise 04

Create a program with an `int` variable and assign it a value.
Then, use a pointer to change its value to another number.
Print the initial and the final values of the variable.

Examples:



```
Meu terminal$ ./pointer_changing
--- Changing Variable Value Using Pointer ---

Initial value of myNumber: 10
Final value of myNumber (after change via pointer): 25
```

Allowed functions: printf

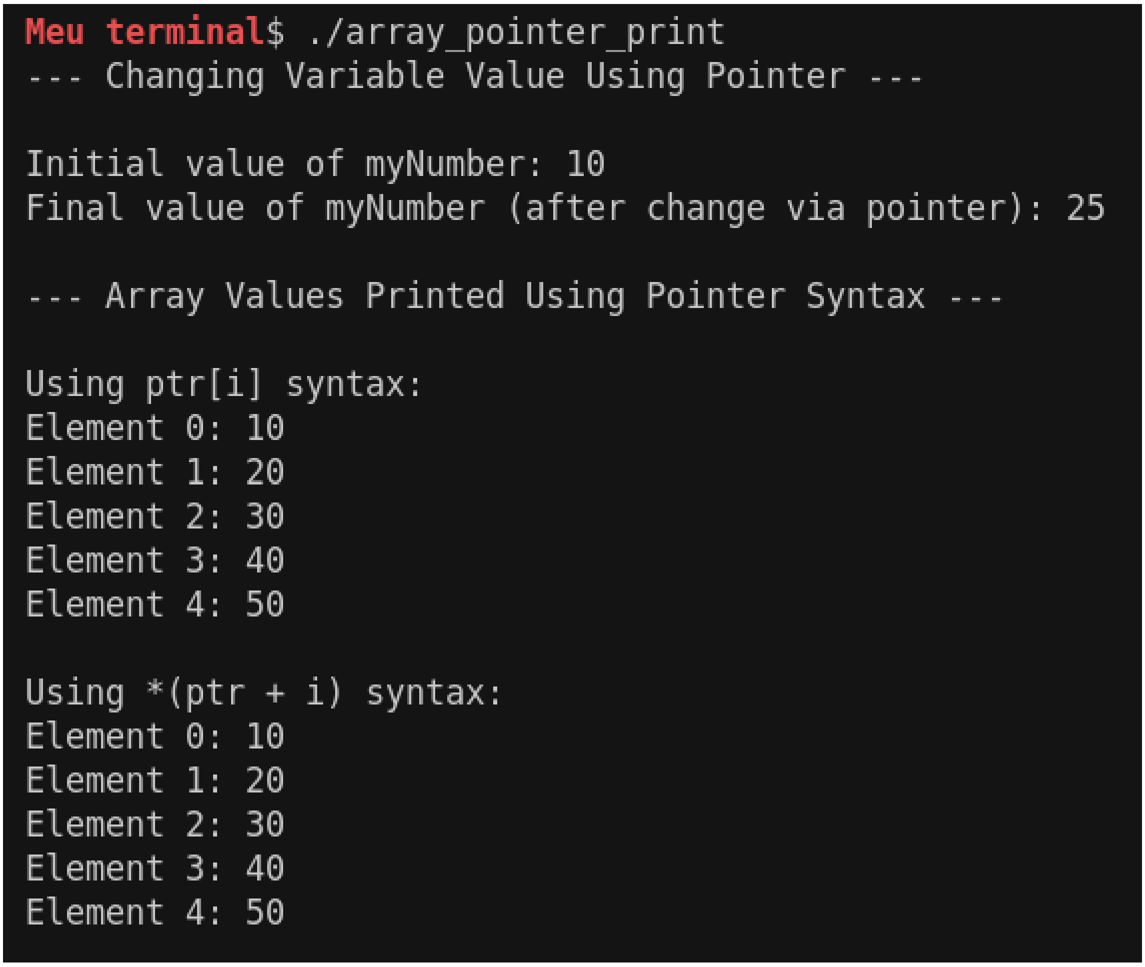
Exercise 05

Create a program with an array of 5 positions and initialize it with values of your choice. Use a pointer to print all the values using:

- `ptr[i]` syntax
- `*(ptr + i)` syntax

Do both in the same program.

Example:



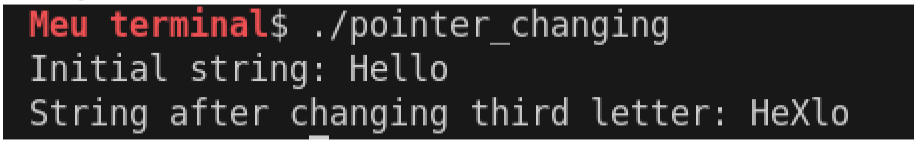
```
Meu terminal$ ./array_pointer_print
--- Changing Variable Value Using Pointer ---

Initial value of myNumber: 10
Final value of myNumber (after change via pointer): 25

--- Array Values Printed Using Pointer Syntax ---

Using ptr[i] syntax:
Element 0: 10
Element 1: 20
Element 2: 30
Element 3: 40
Element 4: 50

Using *(ptr + i) syntax:
Element 0: 10
Element 1: 20
Element 2: 30
Element 3: 40
Element 4: 50
```

Allowed functions: printf

Exercise 06

Create a program that has a char array with the string “Hello” as its value.
Print the string. Then, use a pointer to change a letter at the third position.
Print the string again.

Example:



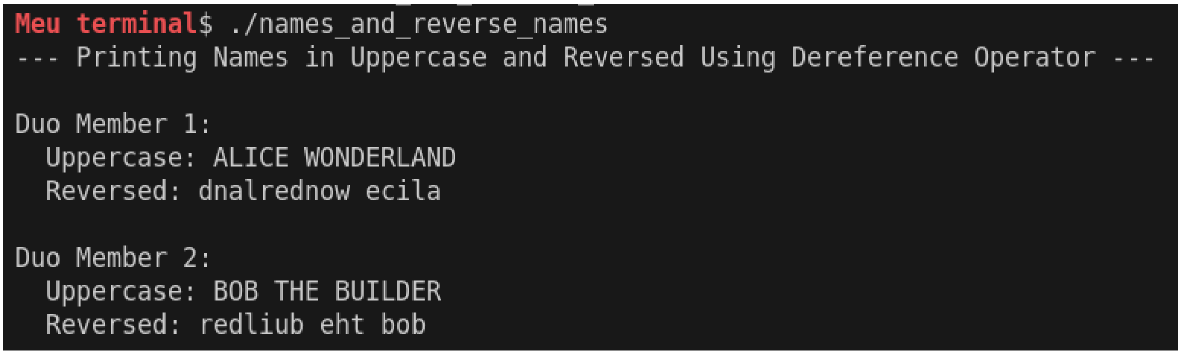
```
Meu terminal$ ./pointer_changing
Initial string: Hello
String after changing third letter: HeXlo
```

Allowed functions: write

Exercise 07

Create a program that will have 2 arrays, each containing the full name of a member of the duo, in lower case. Print each name in upper case using only the dereference operator (*).
Then, print each name reversed with the same operator.

Example:



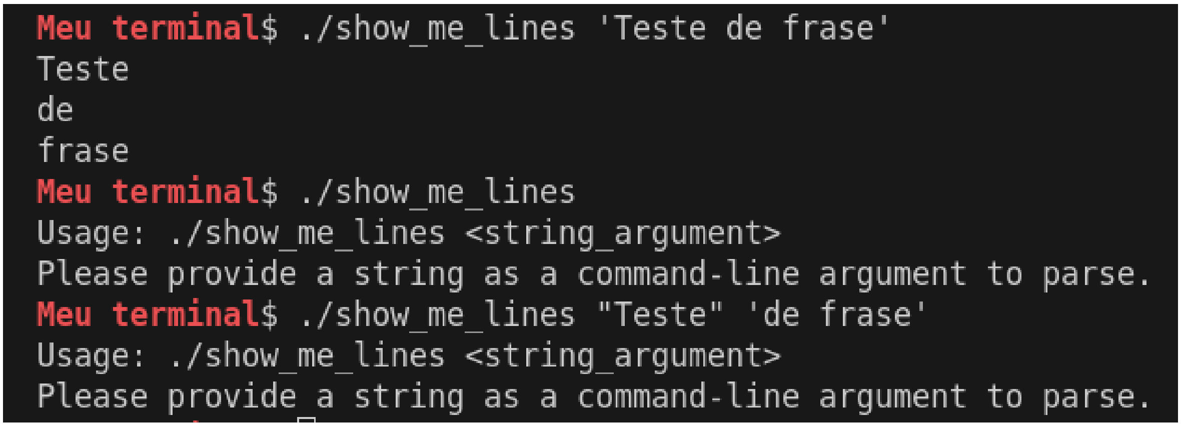
```
Meu terminal$ ./names_and_reverse_names
--- Printing Names in Uppercase and Reversed Using Dereference Operator ---

Duo Member 1:
  Uppercase: ALICE WONDERLAND
  Reversed: dnalrednow ecila

Duo Member 2:
  Uppercase: BOB THE BUILDER
  Reversed: redliub eht bob
```

Allowed functions: write

Exercise 08

Create a program that receives an argument.
Print each word on a new line by manually detecting spaces.
Use only pointer arithmetic and dereferencing.

Example:



```
Meu terminal$ ./show_me_lines 'Teste de frase'
Teste
de
frase
Meu terminal$ ./show_me_lines
Usage: ./show_me_lines <string_argument>
Please provide a string as a command-line argument to parse.
Meu terminal$ ./show_me_lines "Teste" 'de frase'
Usage: ./show_me_lines <string_argument>
Please provide a string as a command-line argument to parse.
```

Allowed functions: printf