Module 6:

Arrays

COP2274

In-class Assignments

UF UNIVERSITY of FLORIDA

M6A Functions with an array parameter

1. Write a void function called **fill()** to fill an array with doubles with the size given by the user, where an entire array and its size are the parameters of **fill()**.
2. Write a function called **mean()** to take in an entire array and its size as parameters and return the mean of the given array of doubles.
3. Write a function called **min()** to take in an entire array and its size as parameters and return the minimum value of the given array of doubles.
4. Write a function called **max()** to take in an entire array and its size as parameters and return the maximum value of the given array of doubles.

M6A Functions with an array parameter

5. In main, get the size of an array from the user and fill the array with your **fill()**. Output the mean, minimum, and maximum values of the array using your **mean()**, **min()** and **max()** as shown in the test case.

Note:

- *You may assume that the user will not request to create an array of more than 50 elements.*

Test case

```
Enter the size you'd like to make your array, up to 50: 4
Enter value for arr[0]: 4.5
Enter value for arr[1]: 19.8
Enter value for arr[2]: 5.12
Enter value for arr[3]: 9.2
The mean of the array is: 9.655
The min value of the array is: 4.5
The max value of the array is: 19.8
```

M6B Functions with an array parameter

1. Write a function called **fill()** function to fill an array with integers with the size given by the user, where an entire array and its size are the parameters of **fill()**.
2. Write a function called **isRepeat()** to take in an entire array, its size, and a target number as parameters and return true or false after checking if the target number is repeated in the given array of integers.
3. Write a function called **deleteNumbers()** to take in an entire array and its size as parameters and overwrite the array with only the elements after deleting the repeated elements of the array. Use/call **isRepeat()** to check if there is a repeated element or not in the given array.

M6B Functions with an array parameter

4. Write a function called **print()** to take in an entire array and its size as parameters and print all the elements of the array.
5. In main, get the size of an array from the user and fill the array with your **fill()**. Print the user-entered array with your **print()**. Output the array with repeated elements deleted by **deleteNumbers()**as shown in the test case.

Note:

- *The user should not request to create an array of more than 25 elements. If the user requests an array of more than 25 elements, your program should display an error message and prompt the user input again as shown in the test case.*

M6B Functions with an array parameter

Test case

```
Enter the number of elements in your array (1-25): 0
Invalid entry!
Enter the number of elements in your array (1-25): 26
Invalid entry!
Enter the number of elements in your array (1-25): 10
Enter 10 numbers:
1 1 2 2 3 3 4 4 5 6

Array Entered: 1 1 2 2 3 3 4 4 5 6
Array with repeated numbers deleted: 1 2 3 4 5 6
```

M6C Functions with an array parameter

1. Using the **fill()** and **print()** functions from M6B, write a new function called **sort()** that takes an entire array, the size of the array, and a bool as the parameters. If the bool is true, the **sort()** will sort the array from smallest to largest. If the bool is false, it will sort from largest to smallest.
2. To aid in writing your **sort()**, you should also write a void function called **swap()** that swaps 2 integers with 2 call-by-reference parameters. After every swap you make in your **sort()**, call your **print()** to display the current state of the array. Your **sort()** should return an integer which contains the number of swaps you made. *Note: the number of swaps may vary depending on how you define your sort function.*

M6C Functions with an array parameter

3. Test your sort function in main according to the test cases

Note:

- *You may assume that the user will not want to enter an array with more than 10 elements.*

Test case 1

```
How many elements in your array [up to 10]: 5
Enter 5 integers separated by spaces:
20 -21 123 -77 3
You entered the array:
20 -21 123 -77 3
Sort in [i]ncreasing or [d]ecreasing? d
123 -21 20 -77 3
123 20 -21 -77 3
123 20 3 -77 -21
123 20 3 -21 -77
Our array was sorted in 4 swaps
```

Test case 2

```
How many elements in your array [up to 10]: 5
Enter 5 integers separated by spaces:
20 -21 123 -77 3
You entered the array:
20 -21 123 -77 3
Sort in [i]ncreasing or [d]ecreasing? i
-21 20 123 -77 3
-77 20 123 -21 3
-77 -21 123 20 3
-77 -21 20 123 3
-77 -21 3 123 20
-77 -21 3 20 123
Our array was sorted in 6 swaps
```