

1 Instructions

When you are finished submit all your work through the MyClasses page for this class. Create a directory called Lab2, put each programming exercise into its own subdirectory of this directory, zip the entire Lab2 directory up into the file `Lab2.zip`, and then submit this zip file to Lab #2.

You will be working pairs on this, called paired programming. In paired programming you work together on each exercise, bounce ideas off of each other, fix errors together, etc.. It DOES NOT MEAN that you divide the exercises between you and do them separately.

2 Programming Exercises

1. Write a program that will,

- Ask the user for the size of an array which will be populated with random numbers between 1 and an input number n .
- Then have the program ask the user for the maximum random value for n .
- Finally ask the user if they wish to print the array out to the screen.
- Have the program create a dynamic array of the user specified size and populate it with random numbers between 1 and the input number n .
- If the user specified to display the array have the array printed to the screen twice, one time using a for loop and array bracket notation for array access and the second time with a while loop that uses a pointers and dereferencing to print out the array, no bracket notation can be used here.
- Then have the program create another dynamic array called `counts` that will hold the counts of each of the values between 1 and n that are in the array.
- Have the program display these counts.
- If the user inputs a number that is 0 or less for either the array size or the maximum random number have the program end and return a 1 from the main.

Note: Make sure that all the memory is cleaned up before the end of the program so that there are no memory leaks, memory access errors, and no multiple frees.

As a challenge, and a little extra credit, when you populate the `counts` array with the counts of the random values from the array do it with a single line body to a standard for loop that uses only pointer arithmetic with references and dereferences. That is, no bracket notation, no conditional statements, no additional loops.

The following are two separate runs of the program.

```
Input size of array: 25
Input n > 0 for upper bound on range of random numbers, [1, n]: 5
Do you want a printout of the array? (Y/N):y
```

```
Array
4 1 2 4 1 2 4 3 2 2 2 3 2 1 4 4 1 4 4 4 5 1 2 2 2
4 1 2 4 1 2 4 3 2 2 2 3 2 1 4 4 1 4 4 4 5 1 2 2 2

Counts
1: 5
2: 9
3: 2
4: 8
5: 1
```

```
Input size of array: 1000000
Input n > 0 for upper bound on range of random numbers, [1, n]: 7
Do you want a printout of the array? (Y/N):n

Counts
1: 142358
2: 142522
3: 142841
4: 142551
5: 143399
6: 142933
7: 143396
```

2. Create a program that is an update of the previous program. This one will simulate die rolls of standard (fair) six-sided dice, store the sum of the die rolls in an array for each roll, and then displays the counts of each roll value.
 - The program will ask the user for the number of rolls to do and the number of dice to roll each time.
 - It will also ask the user if they wish to print the rolls to the screen.
 - As with the previous exercise all of the arrays are to be dynamically allocated.

Make sure that all the memory is cleaned up before the end of the program so that there are no memory leaks and no multiple frees.

Note: Be careful on how you calculate the value of rolling several dice. As you know, for generating random numbers you start by setting the seed of the random number generator using `srand(time(0))`. Then you can calculate a single die roll with,

```
int roll = 1 + rand() % 6;
```

Now rolling 2 dice would be

```
int roll1 = 1 + rand() % 6;
int roll2 = 1 + rand() % 6;
int total = roll1 + roll2;
```

This is not the same as

```
int roll = 2 + rand() % 11;
```

That is, to roll multiple dice, do single die rolls and add their results. The following are two separate runs of the program.

```
Input the number of rolls: 25
Input the number of dice: 3
Do you want a printout of the array? (Y/N):y

Rolls
12 11 10 18 9 13 9 12 8 8 14 9 11 12 6 12 18 9 4 8 7 7 8 5 9

Counts
3: 0
4: 1
5: 1
6: 1
7: 2
8: 4
9: 5
10: 1
11: 2
12: 4
13: 1
14: 1
15: 0
16: 0
17: 0
18: 2
```

```
Input the number of rolls: 1000000
Input the number of dice: 5
Do you want a printout of the array? (Y/N):n

Counts
5: 124
6: 621
7: 1886
8: 4561
9: 9127
10: 16255
11: 26231
12: 39483
13: 54172
14: 69097
15: 83277
16: 94555
17: 100770
18: 100195
19: 94571
20: 83593
21: 69527
22: 53990
23: 38846
24: 26536
25: 16344
26: 9040
27: 4487
28: 1957
29: 628
30: 127
```