Laboratory 05B

CS-102

Spring 2022

Lab 05B – Part 1

- **Problem 1:** Write a program that inputs a number and prints all of the divisors or factors of that number (including 1 and itself).
  - Enter a positive integer.
  - Validate that it is 2 or greater.
  - Then determine and print all the divisors of that number including 1 and itself.
  - This means you will test all the possible divisors USING THE MOD OPERATOR (%). (Make sure you understand how to use it)
  - Call your program: *YourName*-Lab05B-1.cpp

Test Case 1 for Part 1

```
Enter a number and I will tell you its divisors: -7
Must be 2 or greater, re-enter: 1
Must be 2 or greater, re-enter: 77
The divisors of 77 are
1
7
11
77
```

Do the following Test Cases for Part 1

- **Test case 2:** 11
- **Test case 3:** 100
- If you are doing this Lab synchronously, demonstrate *YourName*-Lab05B-1.cpp to the instructor for credit.
- If you are doing this Lab asynchronously, submit the program to Canvas.

Lab 05B – Part 2

- **Problem 2:** Write a program that calculates and displays the sum of the integers from 1 to a given number.
  - The user inputs an integer number of 2 or more.
  - The program should calculate and display the sum of all of the integers from 1 to this number.
  - (For example, if you input 4 as the number, the program should calculate the sum: 1 + 2 + 3 + 4 and display the answer as 10.)
  - <u>Use a loop to validate the input,</u> i.e. make sure that the number input is 2 or greater before proceeding with the calculation.
  - Call your program: *YourName*-Lab05B-2.cpp

Test Case 1 for Part 2

```
Enter a positive integer:      -4
Must   be greater than 2, re-enter:          1
Must   be greater than 2, re-enter:          4
The sum of the integers from 1 to 4 is:  10
```

Do Test Cases 2 & 3:

- **Test Case 2:** 22
- **Test Case 3:** 50
- If you are doing this Lab synchronously, demonstrate *YourName*-Lab05B-2.cpp to the instructor for credit.
- If you are doing this Lab asynchronously, submit the program to Canvas.

Lab 05B – Part 3

- **Problem 3:** Write a program to see how your money grows in an investment account.
  - You will enter the initial amount deposited, the interest rate, and the number of years the money will remain in the account.
  - Assume no withdrawals are made during this time period.
  - You may assume that the interest is compounded annually.
  - Calculate and display the balance at the end of each year based on the interest rate.
  - Also display the total amount of interest earned.
  - Use loops to validate that the deposit amount, interest rate, and number of years are greater than 0.
  - Call your program: *YourName*-Lab05B-3.cpp

Test Case 1 for Part 3

```
What is inital deposit? -1000
Invalid, must be greater than 0. Re-enter: 1000
Annual interest rate (as percent)? 0
Invalid, must be greater than 0. Re-enter: 10
Number of years on deposit? -3
Invalid, must be greater than 0. Re-enter: 3
Let's see how your money grows!
Year    Balance
1       1100.00
2       1210.00
3       1331.00
Total interest earned is $ 331.00
```

Do the following Test Cases for Part 3

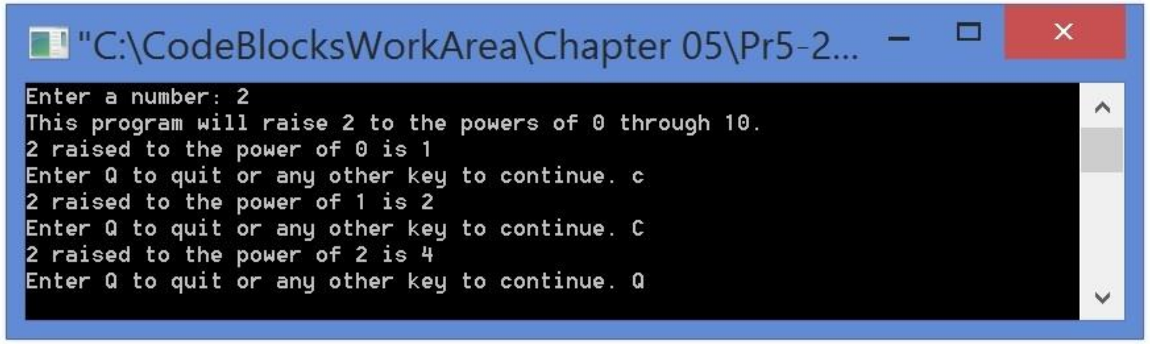
- **Test case 2:** $5000 deposited for 10 years at 7% interest
- **Test case 3:** $10000 deposited for 20 years at 5% interest
- If you are doing this Lab synchronously, demonstrate *YourName-*Lab05B-3.cpp to the instructor for credit.
- If you are doing this Lab asynchronously, submit the program to Canvas.

Laboratory 5B – Part 4

- In this Lab we are going to look at a C++ command which violates the rules of Structured Programming when used in loops, yet is tempting to use in certain situations.
- As the **break** command is not necessary, we're going to show you where it might appear and how you can avoid its use altogether.
  - In Structured Programming, a loop is allowed to have one entrance and one exit.
  - The **break** command allows the loop to have more than one exit point.
  - This makes the loops more difficult to follow as well as debug.
- Program 5-25 shows the use of the **break** command.
- Go ahead and enter this program and get it working. Test it so that you are clear about how it works.

Program 5-25 Breaking a Loop Example

```
// This program raises the user's number to the powers of 0 through 10.
#include <iostream>
#include <cmath>
using namespace std;
int main()
{
        int value;
        char choice;
        cout << "Enter a number: ";
        cin >> value;
        cout << "This program will raise " << value;
        cout << " to the powers of 0 through 10.\n";
        for (int count = 0; count <= 10; count++)
        {
                cout << value << " raised to the power of ";
                cout << count << " is " << pow(value, count);
                cout << "\nEnter Q to quit or any other key to continue. ";
                cin >> choice;
                if (choice == 'Q' || choice == 'q')
                        break;
         }
        return 0;
}
```



"C:\CodeBlocksWorkArea\Chapter 05\Pr5-2...

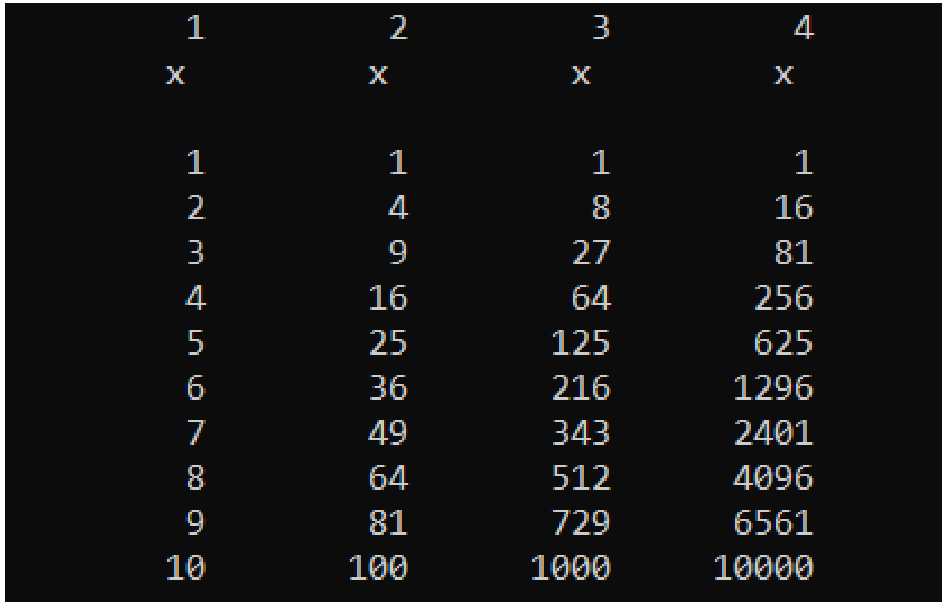
```
Enter a number: 2
This program will raise 2 to the powers of 0 through 10.
2 raised to the power of 0 is 1
Enter Q to quit or any other key to continue. c
2 raised to the power of 1 is 2
Enter Q to quit or any other key to continue. C
2 raised to the power of 2 is 4
Enter Q to quit or any other key to continue. Q
```

Laboratory 5B – Part 4

- As you can see, you can exit the loop either through the **while ()** statement, or you can exit through the **break** statement.
- Your job is to rewrite the program so that the **break** statement is no longer needed.
- Tip 1: You will need to replace the **for ()** loop with a **while()** loop.
- Tip 2: The argument for the **while ()** loop will contain both the condition from the **for ()** loop as well as the condition that would have caused the **break** to occur. These two conditions should be combined with a logical operator.
- Call your program: *YourName*-Lab05B-4.cpp
- After you have worked out the new argument, rewrite the relevant lines and after you have it working:
    - If you are doing this Lab synchronously, demonstrate *YourName*-Lab05B-4.cpp to the instructor for credit.
    - If you are doing this Lab asynchronously, submit the program to Canvas.

Laboratory 5B – Part 5
Nested For Loop

- Suppose we wish to make a table where we want to print out x to the first, second, third and fourth powers.
    - Supposing we want to do this for n = 1 up till n =10.
- Write a program with a nested loop where n indicates the number of the row and x indicates the number of the column.
- You will want to want to put a header on the table so it looks like this:



```
      1          2          3          4
     x          x          x          x

      1          1          1          1
      2          4          8         16
      3          9         27         81
      4         16         64        256
      5         25        125        625
      6         36        216       1296
      7         49        343       2401
      8         64        512       4096
      9         81        729       6561
     10        100       1000      10000
```

Laboratory 5B – Part 5

- Call your program: *YourName*-Lab05B-5.cpp
- If you are doing this Lab synchronously, demonstrate *YourName*-Lab05B-5.cpp to the instructor for credit.
- If you are doing this Lab asynchronously, submit the program to Canvas.