Programming Fundamentals – Spring 2022

(BS-IT-F21 Morning)

Assignment # 1

Assigned on: Thursday, September 15, 2022
Submission Deadline: Thursday, September 22, 2022 (till 11:49 PM)

Instructions

- This is an individual assignment. You are NOT allowed to work/submit in the form of group. Absolutely NO collaboration is allowed. Any traces of plagiarism/cheating would result in an "F" grade in this course.
- Do **NOT** copy even a single line of code from any other person or book or Internet or any other source.
- Named your cpp file as task1.cpp, task2.cpp, task3.cpp, …., task15.cpp
- Compress all 15 cpp files only (no screen shorts are required) and named the zipped file as **BITF21M001-A1.zip/BITF21M001-A1.rar** (change according to your roll no)**.**
- Late submissions will NOT be accepted, whatever your excuse may be.

Task 1: Multiplication without *

Write a program that asks the user to enter two positive integers. Then you have to calculate their product without using the symbol (*) and print the result on the screen.

Input validation: Your program has to ask the user, again and again, to enter positive integers unless the user enters valid integers.

Hint: Think of repetitive addition.

Sample Run:

```
Enter first number: -1
Invalid Input! Enter first number
again: 12
Enter second number: 3

Product is:  36
```

Task 2: Division without /

Write a program that asks the user to enter a positive dividend. After that, the program asks the user to enter a positive divisor and you have to validate whether the divisor is less than or equal to the dividend or not. Then you have to calculate their remainder without modulus symbol (%) and quotient without divide symbol (/) and print the result on the screen.

Input Validation: Your program has to ask the user, again and again, to enter positive dividend and divisor unless the user enters valid integers. And also validate that the divisor must be less than or equal tothe dividend.

Hint: Think of repetitive subtraction.

Sample Run:

```
Enter Dividend: -3
Invalid Input! Enter Dividend
Again: 14
Enter Divisor: 18
Invalid Input! Enter Divisor
Again: 5

Quotient: 2
Remainder: 4
```

Task 3: Addition by Hand

What is addition by Hand: All of you have used this method of adding two numbers in your childhood. In this method, we start from the right side of both numbers and add their last digit and then add the carry of their sum to the second last digits sum and this process goes on unless both numbers end.

Write a program that asks the user to enter two positive numbers. Then you have to calculate their sumby using the above-explained method and print the result on the screen.

Input validation: Your program has to ask the user, again and again, to enter positive integers unless the user enters valid integers.

Note: Both numbers may not be of the same length.

Sample Run:

```
Enter First Number: -7
Invalid Input! Enter Number
Again: 14567
Enter Second Number: 1234
  14567
+  1234
-----------
  15801
```

Post Assignment Work: You can also Perform the multiplication by Hand by using the same procedure.

Task 4: Binary Addition

Write a program that asks the user to enter two binary numbers. Then you have to calculate their sum and print the result on the screen.

Assumption: The user always enters a valid binary number (combination of only 0's and 1's)

Note: Both binary numbers may not be of the same length.

Hint: Binary Addition can be done the same as we did the above task (Sum by Hand) with some ruleswhich are explained below.

1) If digit1 is 0 and digit2 is 0 and the carry of the previous addition is 0 then the sum is 0 and the carry is also0.

2) If digit1 is 1 and digit2 is 0 and the carry of the previous addition is 0 then the sum is 1 and the carry is 0.

3) If digit1 is 1 and digit2 is 1 and the carry of the previous addition is 0 then the sum is 0 and the carry is1.

4) If digit1 is 1 and digit2 is 1 and the carry of the previous addition is 1 then the sum is 1 and the carry isalso 1.

5) If digit1 is 1 and digit2 is 0 and the carry of the previous addition is 1 then the sum is 0 and the carry is 1.

6) If digit1 is 0 and digit2 is 1 and the carry of the previous addition is 1 then the sum is 0 and the carry is 1.

Sample Run:

```
Enter First Binary Number:
1110011
Enter Second Number: 111010

   1110011
+   111010
----------------------
 10101101
```

Task 5: Prime Number

Write a program that asks the user to enter a positive integer and prints whether the integer is prime or not a prime.

Note: An integer is said to be prime if it has exactly 2 divisors. For example, 5 is a prime number because it has only two divisors 1 and 5.

Hint: Use a loop.

Sample Run:

```
Enter a Positive Integer: -1
Invalid Input! Enter Positive
Integer Again: 17

17 is a Prime Number.
```

Task 6: Twin Primes

Write a program that asks the user to enter a positive integer n. Then you have to print all the twin primes from 2 to till the n(inclusive).

Note: Prime numbers which have a difference of 2 are called twin primes. For example, (3,5) is a twin prime.

Hint: You can use the above task (Prime Number) to do this task. Use a nested Loop.

Sample Run:

```
Enter a Positive Integer: 19
Twin Primes between 2 and 19 are:
(3,5)
(5,7)
(11,13)
(17,19)
```

Task 7: LCM (least Common Multiple)

Write a program which ask the user to enter two positive integers. Then you have to calculate their LCM and print the result on screen.

Note: LCM is the least number which is exactly divisible by both numbers.

Hint: Use a loop.

Sample Run:

```
Enter first number: -1
Invalid Input! Enter first number
again: 8
Enter second number: 12

LCM is: 24
```

Task 8: Digit, Alphabet, Others

Write a program that asks the user to enter a character. Then your display the message on the screen whether it is a digit, small alphabet, or capital. If the character is not a digit or alphabet then you simply haveto print It as some other Character.

Hint: Think of **ASCII** Values of Characters.

Sample Run:

```
Enter a Character: 8
It is a Digit!
Do you want to enter again(Y/N): Y

Enter a Character: T
It is a Capital Alphabet!
Do you want to enter again(Y/N): Y

Enter a  Character:  a
It is a small Alphabet!
Do you want to enter again(Y/N): Y

Enter a Character: $
It is some other Character!
Do you want to enter again(Y/N): N
```

Task 9: CGPA Calculator

Write a program which ask the user to enter his roll number (integer value) and his number of semesters completed. Now ask the user to enter GPA of thegiven number of semesters. Calculate the CGPA and displays it as shown in the sample output.

Sample Run:

```
Enter your Roll Number: 12
Enter Number of Semesters: 2
Enter Your Semester 1 GPA: 3.5
Enter Your Semester 2 GPA: 3.8
********************* Result Card***************
Roll Number: 12
CGPA: 3.65
```

Note: Assume that Semester 1 has Credit hours 12, Semester 2 has 13, Semester 3 has 14 and so on. Number of Semesters can't be less than 1 and greater than 8, Roll Number Can't be negative, GPA can't be less than 0 or greater than 4. If any of these constraints are broken indicated error and prompt the user again and again until he enters the value in the correct range.

Task 10: Corresponding Number Finder

In this task you have to write a program which finds out the corresponding number of 1 series to other. Let's say there's a series.

Example:
Series 1:

5,7,9,11,13,15

Here 5 is a1, and the difference between values is d which is 2.
Both a1 and d will be entered by the user.

Series 2:
The first number of Series 2 which is n1 will be entered by the user, if n1=7 then series 2 will be,

7,8,9,10,11,12

Note that Series 2 starts with a user-specified value and goes on with the increment of 1.
Now ask the user to enter the number from series 1 and then print the corresponding value from series 2.
If a user enters 11 You should print 10 on the screen, for 5 it's 7 and for 15 you should print 12.

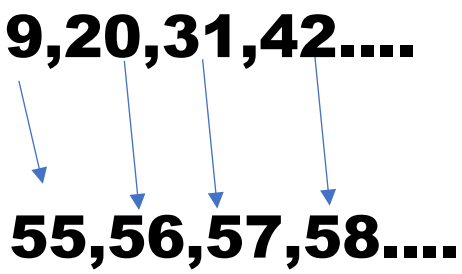
Sample Run 1:

```
Enter First Number of Series 1: 5
Enter the difference of series 1: 2
Enter First Number of Series 2: 7
Enter Number from the Series 1: 13
Your Corresponding Number is 11.
```

Sample Run 2:

```
Enter First Number of Series 1: 9
Enter the difference of series 1: 11
Enter First Number of Series 2: 55
Enter Number from the Series 1: 42
Your Corresponding Number is 58.
```

Explaination:



You can Assume that user will always enter a legal value from series 1 e.g in the sample run above user will not enter 40 as it's not in the series.

Hint:

Make Such series and work on their correspondence.

Task 11: Printing Sequence

Write a program which takes two integers from the user and displays the following sequence in the specified format. (***Hint:*** *Use nested loops*)

Two sample runs of your program will look like as follows. User's input is **underlined**.

Sample Run 1:

```
Enter starting integer: 1
Enter ending integer: 5
(1,1)(1,2)(1,3)(1,4)(1,5)
(2,2)(2,3)(2,4)
(3,3)
```

Sample Run 2:

```
Enter starting integer: 11
Enter ending integer: 18
(11,11)(11,12)(11,13)(11,14)(11,15)(11,16)(11,17)(11,18)
(12,12)(12,13)(12,14)(12,15)(12,16)(12,17)
(13,13)(13,14)(13,15)(13,16)
(14,14)(14,15)
```

Task 12: printing diamond-like shape

Write a program that takes a *positive odd integer* (greater than or equal to 3) from the user and prints the following diamond-like shape on the screen. For example, if the input provided by the user is 9, the following figure should be displayed by your program:

```
    *
   ***
  *****
 *******
*********
 *******
  *****
   ***
    *
```

In order to display the above pattern, your program should use ONLY following THREE print statements:

```
printf ("*");     printf (" ");     Printf ("\n");
```

Input validation: Your program should make sure that the size is ***an odd integer greater than or equal to 3****. In case of invalid input, your program should keep prompting the user again and again till the user provides valid input.*