National University of computer and emerging sciences

Foundation for Advancement of Science and Technology

Programming Fundamentals (CS 1002)

Cybersecurity Department

Fall 2022 ASSIGNMENT # 4

~~Due Date: Friday, November 18, 2022 (11:59 pm)~~

Due Date: Friday, November 13, 2022 (11:59 pm)

Instructions

1. Assignments are to be done individually. You must complete this assignment by yourself. You cannot work with anyone else in the class or with someone outside of the class. The code you write must be your own and you must understand each part of your code. You are encouraged to get help from the instructional staff through google classroom/Piazza.
2. Use appropriate data types and operations for each problem. You cannot use advanced topics not covered so far.
3. Your code must be **generic** i.e. it should work for different inputs where inputs are required.
4. The output should be properly displayed and well presented. 5% marks will be deducted in each question if appropriate comments and indentation not done in source code.
5. **Plagiarism:** Plagiarism of any kind (copying from others, copying from the internet, etc) is not allowed. If found plagiarized, you will be awarded zero marks in the assignment. Repeating such an act can lead to strict disciplinary actions and failure in the course.
6. **Submission Guidelines**: Dear students, we will be using auto-grading tools, so failure to submit according to the below format would result in zero marks in the relevant evaluation instrument.
    a. Run and test your program on a lab machine before submission. If there is a syntax error, zero marks will be awarded in that specific question.
    b. For each question in your assignment, make a separate .cpp file e.g. for question 1, make ROLL-NUM_SECTION_q1.cpp, and so on (e.g. **22i-0001_A_q1.cpp, 22i-0001_A_q2.cpp, 22i-0001_A_q3.cpp** and so on).
    c. In every .cpp file that you create (every question), you must write your name, student-id, and assignment # on the top of the file in the comments.
    d. Combine all your work (all questions files) in one folder. The folder must contain only .cpp files (no binaries, no exe files etc.,). If we unable to download your submission due to any reason you will be awarded zero mark.
    e. Rename the folder as ROLL-NUM_SECTION (e.g. 22i-0001_A) and compress the folder as a zip file. (e.g. **22i-0001_A.zip**). Only zip file will be acceptable.
    f. Submit the .zip file on Google Classroom within the deadline. Make sure that you have submitted the correct file.
    g. Submission other than Google classroom (e.g. email etc.) will not be accepted.
    h. The student is solely responsible to check the final zip files for issues like corrupt files, viruses in the file, mistakenly exe sent. If we cannot download the file from Google classroom due to any reason it will lead to zero marks in the assignment.
7. **Late submission:** 10% marks will be deducted for every hour of late submission, i.e. assignments submitted 10 hours late will get zero marks. 10% will be deducted with the start of hour so both submissions on 10:01 and 10:59 will get same deduction of 10%.

1. **(Calculating the Value of π):** Write a function which calculates the value of π from the infinite series given below. The function displays a table listing term number and value of π. 5 rows of the table are shown at a time and then stops; user enters a key "S" to stop printing it further or "C" to continue printing next 5 rows. This continues until user enters "S". We don't want to prompt any message to enter S or C.

   The function also returns the number of terms of this series before we first get the value 3.14. This returned value is then shown in the main program. However, it is also possible that the user stopped the series by pressing "S" before getting 3.14 value; in that case, the function will return -1 and the main program will display a message that we did not find the number of terms to get 3.14 value.

   Use manipulators to display properly aligned values. It would be much better if colours and bold text is used where needed.

   $$\Pi = 4 - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \frac{4}{9} - \frac{4}{11} \ldots\ldots$$

Term#	Value of Π
1	4
2	2.6
3	3.46
4	2.89
5	3.339

2. Write a function which takes a binary number as input and return its decimal equivalent. Use simple and accurate algorithm using while loop to produce the correct result. Then use this function in your program which asks user to enter a binary number in the main program, calls your conversion function to get the decimal value and show it in main program

   Here is an example of an algorithm illustrating how to convert from binary (base 2) to Decimal (Base 10): The binary or base-2 number 1 0 0 1 0 1 0 1 can be converted to a decimal number as follows:

   $$1*2^7+0*2^6+0*2^5+1*2^4+0*2^3+1*2^2+0*2^1+1*2^0$$

   = 128 + 16 + 4 + 1
   = 149

   The binary number 1 0 0 1 0 1 0 1 is written as 149 in the decimal system.

   Program Instructions:

   - Use bigger data types to be able to enter a long binary pattern. For example, 10010101 number is actually an integer "ten million ten thousand one hundred and one".
   - Input validation: Make sure that user enters only ones and zeros pattern. If user enters any other digit in the entered number then error must be displayed and user be asked to re-enter the value.

Enter a binary number: 10010101 The binary number 1 0 0 1 0 1 0 1 is written as 149 in the decimal system.
Enter a binary number: 100121003 Entered number cannot have any digit other than 0 and 1

3. Write a function to get the GCD of two integers (*m*, *n*). Use *Euclidean Algorithm* for finding the greatest common divisor of the two given positive integers. The function gets both integers (*m*, *n*) as parameters and transforms the pair (*m*, *n*) into a pair (*d*, 0) by repeatedly dividing the larger integer by the smaller integer and replacing the larger with the remainder. When the remainder is 0, the other integer in the pair will be the greatest common divisor of the original pair (and of all the intermediate pairs) which is returned by the function. For example, if *m* is 532 and *n* is 112, then your function reduces the pair (532,112) to (28,0) by (532,112) → (112,84) → (84,28)→ (28,0) using Euclidean Algorithm. So 28 is the greatest common divisor of 532 and 112 which is then returned.

Use the above written function in a program which asks user to enter two positive integers, pass these integers to your GCD function to get the GCD value in the main and display the value.

4. Write a function which takes an integer as parameter, reverses that integer number and returns the reversed value. Then use this function in your program to show all palindromes from 10 to 99999. Palindrome is a number which is same when its digits are reversed. For example, 12321 is a palindrome.

5. Write a function **highLow()** that takes an integer as an argument and returns whether or not the number has alternating “high” and “low” digits. 0 through 4 are “low” digits and 5 through 9 are “high” digits. Your function should return true if the number passed alternates between “high” and “low” digits, and false if not. Make sure that the number passed is positive. If the number passed consists of a single digit, **highLow()** should return true.

Note: **highLow()** returns true if the number alternates starting with a “high” digit or starting with a “low” digit. What is important is that the digits alternate. For example, both 9292 and 2929 passed to **highLow()** should return true; whereas, 6837 returns false.

6. Write a function which gets two integer values for “lines” and “cheers” and prints a series of "cheer" lines at increasing levels of indentation. The first parameter represents the number of lines of output to print, and the second represents the number of "cheers" per line. For example, if lines=2 and cheers =4 then you should print 2 lines of output, each containing 4 "cheers." A "cheer" is an occurrence of the word "Go" in the output. Neighbouring cheers are separated by the word "Buddy" (of course you can put name of your favourite political person here as well :)), so 1 cheer is printed as "Go", 2 cheers as "Go Buddy Go", 3 cheers are printed as "Go Buddy Go Buddy Go", and so on.
The lines you print should be displayed at increasing levels of indentation. The first line displayed should have no indentation, but each following line should be intended by 3 spaces more than the one before it. In other words, the 2nd line of output should be indented by 3 spaces, the 3rd line by 6 spaces, and so on.

Use this function in the main program which asks user to enter Lines and Cheers which are then passed to your function to display the required output.

Lines=2 Cheers=1 Go   Go

Lines=4
Cheers=3
 	
Go Buddy Go Buddy Go
  Go Buddy Go Buddy Go
  Go Buddy Go Buddy Go
  Go Buddy Go Buddy Go

Lines=2 Cheers=4 Go Buddy Go Buddy Go Buddy Go   Go Buddy Go Buddy Go Buddy Go

7. PakAsia is a country that has currency notes of 6, 9 and 20 PakAsia Rupees (PAR) only. Thus, it is possible, for example, to exchange exactly 15 Pakistani Rupees (PKR) (with one note of 6 PAR and a second note of 9 PAR will add up to make 15 PAR, since 1PKR == 1PAR), but it is not possible to exchange exactly 16 PKR, since no non-negative integer combination of 6's, 9's and 20's notes of PAR add up to make 16 PKR. To determine if it is possible to exchange exactly n PKR to PAR, one has to find non-negative integer (can be 0) values of a, b, and c such that:
6a+9b+20c=n

Now write a function that takes, 'n' PKR from user as input, and prints if it is possible to exchange it with a combination of 6, 9 and 20 PAR such that the total sum of PakAsia Rupees equals n, and otherwise prints it is not possible. Your function should return true or false which is then checked in the main program to print whether exchange is possible or not.

8. Implement functions **isAlpha(), isDigit(), isAlnum, isXdigit(), isLower(), isUpper(), isSpace().** Each of these functions receive one character as parameter and returns either true or false. These functions work as follows:

- **isAlpha()** returns true if character received by it is an alphabet (A-Z or a-z) otherwise it returns false.
- **isDigit()** returns true if character received by it is a decimal digit (0-9) otherwise it returns false.
- **isAlnum()** returns true if character received by it is a decimal digit (0-9) or an alphabet (A-Z or a-z) otherwise it returns false.
- **isXdigit()** returns true if character received is a hexadecimal digit (0-9 or A-F or a-f) otherwise it returns false.

- **isLower()** returns true if character received by it is a small case alphabet (a-z) otherwise it returns false.
- **isUpper()** returns true if character received by it is a Capital case alphabet (A-Z) otherwise it returns false.
- **isSpace()** returns true if character received by it is a space (“ ”) otherwise it returns false.

Write appropriate porotype and definition of above functions. Write appropriate implementation in main () function.