NATIONAL UNIVERSITY OF COMPUTER & EMERGING SCIENCES ISLAMABAD CAMPUS

Programming Fundamentals (CS1002) | FALL2021

Due Date: Monday 29th November, 2021 (04:00 PM)

Assignment 4– Loops, Loops, and More Loops

[125 marks] + 10 bonus marks

Instructions

Submission: Combine all your work (solution folder) in one .zip file. Use a proper naming convention for your submission file. Name the .zip file as SECTION_ROLL-NUM_04.zip (e.g. A_20i0412_04.zip). Submit .zip file on Google Classroom within the deadline. Failure to submit according to the above format would result in a deduction of 10% marks. Submissions on the email will not be accepted.

Plagiarism: Plagiarism cases will be dealt with strictly. If found plagiarized, both the involved parties will be awarded zero marks in this assignment, all of the remaining assignments, or even an F grade in the course. Copying from the internet is the easiest way to get caught!

Deadline: The deadline to submit the assignment is **29th November 2021 at 04:00 PM**. **Late submissions (even 1 minute) will not be considered**. Correct and timely submission of the assignment is the responsibility of every student; hence no relaxation will be given to anyone.

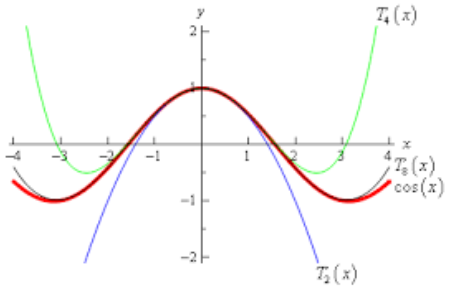
Modular Programming: Your code for each question should be modular. Use functions to decompose your problem into smaller subproblems. This is also mentioned in rubrics.

Comments: Comment your code properly. Write your name and roll number (as a block comment) at the beginning of the solution to each problem.

Tip: For timely completion of the assignment, start as early as possible.

Note: Follow the given instructions to the letter, failing to do so will result in a zero

Question 1: Writing the cos(x) method (20 Marks)



The Math(cmath) library contains a method that allows you to calculate the trigonometric function cos(x). You can execute this method by writing cos(x) for some expression x of type double. Using properties of a Maclaurin series or Taylor series, you can approximate the function cos(x) using the following formula:

$$\cos(x) \approx \sum_{n=0}^{a} (-1)^n \frac{x^{2n}}{(2n)!} = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \frac{x^8}{8!} - \ldots\ldots \frac{x^{2a}}{2a!}$$

Write your own approximation of cos(x) without using **any library** methods (such as pow( ) etc.). Your program should take x (in radians) and an as input. You should make two functions calcTerm( ) and **sumTerms( )** to calculate the value of cos(x). **calcTerm( )** function will compute nth term in the sequence. It should take x and term number as arguments and the return value of the term. sumTerms( ) takes a single argument term value and should be used to calculate the sum of all terms. Finally, your program will have a function testFunction( ), you will call it from main to verify whether your function is working correctly or not.

Question 2: Implementing a calculator using bitwise operators (20 Marks)

Implement a calculator using bitwise operators and for loops only. Your program should run correctly for positive integers only. You calculator should be able to perform following operations.
Basic arithmetic functions (***Addition***, ***subtraction***, ***division*** and ***multiplication***); ***Square*** – to compute the square of the given value and ***Power*** – to compute the power of an integer. It should take two int arguments number and its power.
<u>Any operation that uses operators other than bitwise operators will be awarded zero marks.</u>

Question 3: Perfect Number (5 Marks)

A perfect number is an integer that is equal to the sum of its factors. For example, 6 is a perfect number as 6 = 3+2+1. Write down a program that takes an integer x as an input and prints all perfect numbers from 1 to x.

Question 4. Plotting function (15 Marks)

Write a program to plot a graph of function f(x) = $x^n + x^{n-1}$. Your program should take a maximum absolute value of x as well as a positive integer n as input. You will plot a graph for the range [0, x]. You should label the y-axis according to the maximum value of x.
Sample Output: For x = 3 and n = 2, you should have the following output

```
12            *
10
08
06        *
04
02    *
00  *
    0   1   2   3
```

Question 5: Number System Conversions (10 Marks)

Write a program that outputs the binary, octal, and hexadecimal equivalent of the integer in the range from 1 to 256.

Question 6: Occupancy rate for a hotel (10 Marks)

Write a program that calculates the occupancy rate for a hotel. The program should start by asking the user how many floors the hotel has. A ***for*** loop should then iterate once for each floor. In each iteration, the loop should ask the user for the number of rooms on the floor and how many of them are occupied. After all the iterations, the program should display how many rooms the hotel has, how many of them are occupied, how many are unoccupied, and the percentage of rooms that are occupied. The percentage may be calculated by dividing the number of rooms occupied by the number of rooms.

Input Validation: Do not accept a value less than 1 for the number of floors. Do not accept a number less than 10 for the number of rooms on a floor.

Question 7: Projectile Motion (20 marks) + (10 bonus marks)

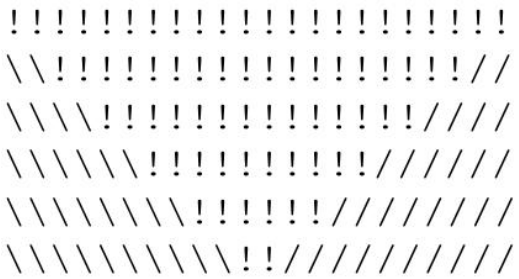
A cannon ball is thrown straight into the air with a starting velocity v_0. The position of the cannon ball after t seconds is given by the equation, $s = v_i t - (1/2) gt^2$, where g = 9 8. m/sec^2 is the gravitational force. You are required to confirm the equation by a simulation. In this simulation, you will consider how the ball moves in very short time interval Δt.

In a short time interval, the velocity v is nearly constant, and we can compute the distance the ball moves using Δs = vΔt. You will assume that Δt is a constant double with a value of 0.01. You can get the updated position using s = s + v * Δt. The velocity changes constantly—in fact, it is reduced by the gravitational force of the earth. In a short time interval, Δv = –gΔt, you must keep the velocity updated as v = v - g * Δt; In the next iteration, the new velocity is used to update the distance.

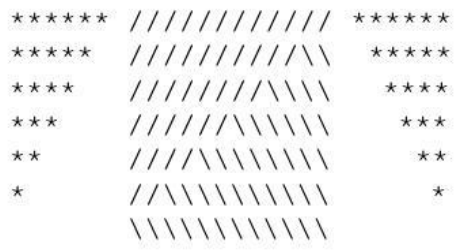
You need to run the simulation until the cannon ball falls back to the ground. Your program should take the initial velocity as an input. Update the position and velocity 100 times per second, but print out the position only every full second. Also, printout the values from the exact formula $s = v_i t - (1/2) gt^2$ for the comparison. Lastly plot the path that cannon ball will take to get the bonus marks.

Question 8: Write a program that produces the following output. (25 Marks)

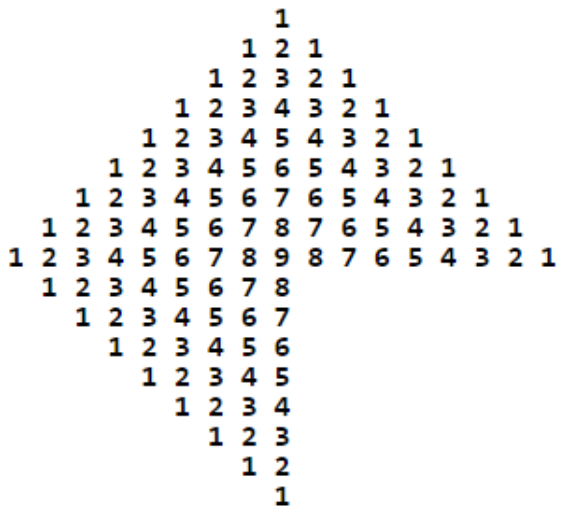
(a) Use nested *While* loops to produce the following pattern:



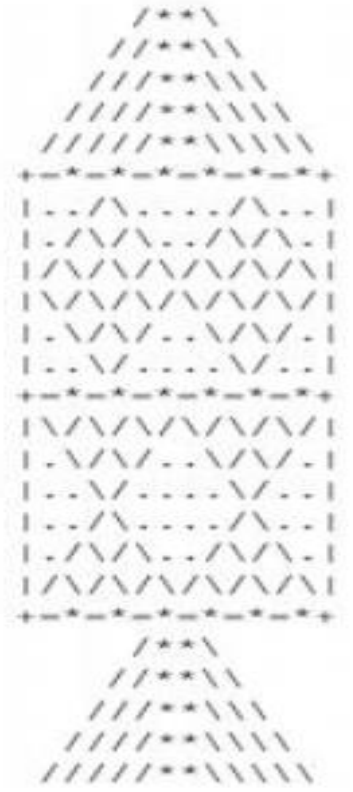
(b) Write nested *for l*oops to produce the following output.



(c) Write nested *for l*oops to produce the following output.



(d) Write nested *for l*oops to produce the following output.



(e) Write nested *l*oops to produce the following output.