Lab 04
Nested Loops

Lab Objectives

The objective of this exercise is to integrate knowledge of printing, computation, data input, looping in a single program. During the lab, students shall develop programs that need loop(s) inside loop(s) (nested loops).

Task: 01

Write a C++ program that will prompt the user for a positive odd integer within a do-while construct. If the integer is 0, negative or even, prompt the user again. The program goes to the next step when an odd positive integer 'N' is received. The program will output an inverted triangle of asterisks having N '*' in the first row. Each following row has two *'s less than the one above it and is centered with respect to the row above.

Sample Run:

```
Input an odd positive int: 11
***********
 *********
  *******
   *****
    ***
     *
```

Task: 02

A double rectangle of stars as shown below (the user enters height and width)

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

Task: 03

Write a program that displays the following pattern

```
                    1
                  2 2
                3 3 3
              4 4 4 4
            5 5 5 5 5
6         6 6 6 6 6 6
```

Task: 04

Write a C++ program that converts any unsigned integer to any base between 2 and 9.

Sample Run:

Input the base to convert to between 2 & 9:

7
Input any positive unsigned integer:

126
Decimal 126 converted to base 7 is 240

Task: 05

(Special Activity)

In this problem, the program starts up assuming that it is following an object on a 2-dimensional plane as it takes many steps to perform a sort-of "random walk". The object starts off at the "origin" (x = 0 and y = 0). You will input two real numbers that will represent the next coordinates of the object. The computer will calculate the straight-line distance that it has to travel to get there and position the object at that point. Then you will input two more real numbers representing a move from that location, then another position (another pair of points) and another position (pair of points) and so on. To stop the program you have to input the identical new coordinates as the previous step—that is, there is no distance to be traveled for that final step. This is a "sentinel" indicating that the loop should terminate. This last zero-step should not be counted in the number of steps. When the program exits the loop at this point make the program print out the total distance traveled, the total number of moves and the average distance per move.

Here is the pseudo-code for this problem

(a) **START:**

Declare $N = 0$ (Counter for the number of steps), *xi, yi* (New coordinates), $x0 = 0, y0 = 0$ (Current coordinates), s (Distance of a single step) $and\ t = 0$ (Total distance).

(b) Start of the "input/stepping loop"

i. Read in the new coordinates: *xi* and *yi*
ii. Test to see if *xi* and *yi* are valid input. If they are not, go back to (i) else continue on to the next step
iii. Test to see if *xi* and *yi* are identical to *x0* and *y0*. If they are not, go to (iv) else go to step (c)
iv. Increment the step counter N
v. Calculate the distance of the step

$$s = \sqrt{(x_i - x_0)^2 + (y_i - y_0)^2}.$$

vi. Accumulate s in t, the total distance i.e. $t = t + s$
vii. Important! Update the current position $x0 = xi, y0 = yi$
viii. Try to take another step, that is go to (i)

(c) Report the total distance traveled

(d) Report the total number of steps.

(e) Report the average distance traveled per step. Important: make sure that the case of zero steps and zero distance traveled does not cause problems.

(f) **END**