SYDE 121
Lab Number 5

Exercise 1: Diamond of Asterisks

Learning Objectives: Practice developing algorithms for more complicated loops.

Read This First

The objective of this exercise is to print out a diamond of asterisks. For example, given an input value of seven rows, the following would be displayed:

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

What to Do

Write a program, *lab0501.cpp*, which reads in an odd positive number from 3 to 19 and draws a diamond with that many number of rows (do not simply exit if the user does not enter a proper value). Use only output statements that print either a single asterisk (*) or a single blank space, and use only 'for' structures for looping.

A conventional (brute-force) approach would be to write two separate loops, one to print the top half of the diamond, and another to print the bottom half of the diamond. However, a more elegant approach would use a *single* loop structure to print the entire diamond. For full marks, design an algorithm that uses a *single* loop structure (that would count through all the rows), rather than two separate loops. Note that the single loop structure can contain nested loops.

Hint: You may find the abs() predefined function useful. This function takes an integer argument and returns an integer, i.e.

```
int c = abs( a );
```

where integer c is assigned the absolute value of integer a.

Create a *"README_Lab0501.txt"* file, and create detailed instructions on how to run your *lab0501.cpp* program, including what input should be entered.

What to Hand In

Submit both your *lab0501.cpp* files and *README_Lab0501.txt* files to the Lab Assignment 5 LEARN dropbox.

Exercise 2: Payroll

Learning Objectives: Practice using the switch statement.

Read This First

A company pays its employees as managers (who receive a fixed weekly salary), hourly workers (who receive a fixed hourly wage for up to the first 40 hours they work and "time-and-a-half" i.e. 1.5 times their hourly wage, for overtime hours worked), commission workers (who receive $200 plus 6.6% of their gross weekly sales), or pieceworkers (who receive a fixed amount of money per item for each of the items they produce – each pieceworker in this company works on only one type of item).

What to Do

Write a program, *lab0502.cpp*, which computes the weekly pay for each employee. You do not know the number of employees in advance. Each type of employee has its own code – ensure this information is conveyed to the user. Use a **switch** to compute each employee's pay based on the user entered employee's paycode. (Note: enum types are optional but keep in mind that not all compilers allow **cin** and **cout** to directly handle enum datatypes; type casting may be needed.) Within the **switch**, prompt the user to enter the appropriate facts your program needs to calculate each employee's pay based on that employee's paycode. Ensure that the default case for the switch statement is an error condition.

After all the employees have been entered, display a well formatted table indicating the total number of employees and the total salary at each position as well as the overall totals across all positions.

Create a "*README_Lab0502.txt"* file, and create detailed instructions on how to run your *lab0502.cpp* program, including what input should be entered.

What to Hand In

Submit both your *lab0502.cpp* files and *README_Lab0502.txt* files to the Lab Assignment 5 LEARN dropbox.

Exercise 3: Determining pi

Learning Objectives: Practice exiting a loop on a flag condition.

Read This First

The value of pi can be determined using the following infinite series:

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

Infinite series are a means of determining mathematical terms such as pi, sine, cosine, etc. algorithmically, for example, within a calculator. You will learn more about these types of mathematical expressions in your calculus courses.

What to Do

Display a table that shows the value of pi approximated by the first term of this series, followed by the pi approximated by the first two terms, etc. Indicate both the term number and the approximation to pi on the same line. Set your approximation of pi to 12 decimal places for display purposes.

For example, the following output represents the pi approximation using the first 3 terms:

```
1     4.000000000000
2     2.666666666667
3     3.466666666667
```

Since this is an infinite series, you will need a stopping criterion to avoid an infinite loop. Each additional term in this infinite series becomes smaller in magnitude relative to its previous term. This fact means that at some point the numbers begin to settle out (i.e. remain the same from iteration to iteration.) Set up a flag condition that stops the pi calculation when the n^{th} decimal place has settled to a consistent number after three iterations. Allow the user to select the stopping condition, within the range of 1 to 6 decimal places.

Create a "*README_Lab0503.txt"* file, and create detailed instructions on how to run your *lab0503.cpp* program, including what input should be entered.

Hint: You may find the pow(a,b) and floor (a) predefined functions useful. The pow(a,b) function can take in two integer arguments and return an integer, i.e.

```
int c = pow( a, b );
```

where integer c is assigned the value a^b, or `a` to the power of `b`.

The floor(a) function can take in an integer argument and return an integer, i.e.

```
int b = floor( a );
```

where integer `b` is assigned the value of a rounded down, e.g. floor(3.1) == 3.

What to Hand In

Submit both your *lab0503.cpp* files and *README_Lab0503.txt* files to the Lab Assignment 5 LEARN dropbox.

Due Date

All materials for this lab are due **Friday, October 17** by **1:30pm.**