Lab 06: Loops

CSE 4108

Structured Programming I Lab

September 2022

Lab Tasks

1. **Find Max:**

Write a program that finds the largest in a series of numbers entered by the user. The program must prompt the user to enter numbers one by one. When the user enters **0** or **a negative number**, the program must display the largest nonnegative number entered.

Sample run:

Enter a number: 60
Enter a number: 38.3
Enter a number: 4.89
Enter a number: 100.62
Enter a number: 75.2295
Enter a number: 0

The largest number entered was ***100.62***
Notice that the numbers aren't necessarily integers.

2. **GCD Finder:**

Write a program that asks the user to enter two integers, then calculates and displays their greatest common divisor (GCD).

Sample run:

Enter two integers: 12 28

Greatest common division: 4

3. **Reduce the Fractions:**

Write a program that asks the user to enter a fraction, then reduces the fraction to lowest terms.

Sample run:

Enter a fraction: 6/12

In lowest terms: 1/2

4. **Diamond Hands** 💎:

When stocks are sold or purchased through a broker, the broker's commission often depends upon the value of the stocks traded.

Suppose that a broker charges the amounts shown in the following table:

Transaction size	***Commission rate***
Under $2,500	$30 + 1.7%
$2,500–$6,250	$56 + 0.66%
$6,250–$20,000	$76 + 0.34%
$20,000–$50,000	$100 + 0.22%
$50,000–$500,000	$155 + 0.11%
Over $500,000	$255 + 0.09%

The minimum charge is **$39**

Add a loop to the program, so that the user can enter more than one trade and the program will calculate the commission on each. The program should terminate when the user enters ***0*** as the trade value.

Sample run:

Enter value of trade: 30000
Commission: $166.00
Enter value of trade: 20000
Commission: $144.00
Enter value of trade: 0

5. **Is it really Christmas?:**

Given an integer input **n**, build a christmas tree.

Sample run:

Input	Output
1	1 *
2	* *** * *
3	* *** ***** *** *** ***
4	* *** ***** ******* *** *** *** ***

6. **Compare Dates II:**

In *Lab 5 - Selection Statements (compare_dates.c)*, you were asked to write a program that determines which of two dates comes earlier on the calendar. Generalize the program so that the user may enter any number of dates. The user will enter **0/0/0** to indicate that no more dates will be entered.

Sample run:

Enter a date (mm/dd/yy): 3/6/18
Enter a date (mm/dd/yy): 5/17/17
Enter a date (mm/dd/yy): 6/3/17
Enter a date (mm/dd/yy): 0/0/0
5/17/17 is the earliest date

Homework:

1. **Calendars:**

Alex is working under his uncle Mr. Sakamoto. Mr. Sakamoto has asked him to print calendars. Specifically, he wants one-month calendars. Mr. Sakamoto will specify the number of *days* in a *month* and the *day* of the *week* on which the *month* begins.
Alex, being busy with other tasks, has asked you to help.

Sample run:
Enter number of days in month: 31
Enter starting day of the week (1=Mon, 7=Sun): 3

```
Mo Tu We Th Fr Sa Su
        1  2  3  4  5
 6  7  8  9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 31
```

Hint: This program isn't as hard as it looks. The most important part is a for statement that uses a variable i to count from 1 to n, where n is the number of days in the month, printing each value of i. Inside the loop, an if statement tests whether i is the last day in a week; if so, it prints a new-line character

2. **E:**

The value of the mathematical constant e can be expressed as an infinite series:

$e = 1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \ldots$

Write a program that approximates e by computing the value of:

$1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \ldots + \frac{1}{n!}$

where n is an integer entered by the user.

3. **Never Too Small:**

Modify the last task so that the program continues adding terms until the current term becomes less than **ε**, where **ε** is a small (floating-point) number entered by the user.