COP 3014: Spring 2022
Homework 3

Total Points: 150 (plus 50 points extra credit)
Due: Monday, 03/07/2022 11:59:00 PM EST through Canvas

1 Objective

The purpose of this assignment is to

- Familiarize yourself with simple looping constructs.
- Declare, define and use functions in C++.
- Use fundamental algorithmic error checking to check the validity of user input.
- Familiarize yourself with selection statements (if . . . else and switch).
- Use variables and arithmetic operators in C++ to do basic math.
- Practice using CLion to create, edit, compile, debug, and run a c++ program.
- Use the `iostream` library to perform console input and output.
- Use the cmath library to perform certain mathematical operations.

From this assignment, please make sure you conform to Output requirements. So far, we have allowed you to write your own output statements, print out intermediate steps, etc. You should now be familiar enough with cout statements and output formatting that you should be able to EXACTLY match the sample output (other than whitespace and floating point precision when it is not specified).

This assignment requires you to submit 3 (or 4) files on Canvas. Please do so in a SINGLE submission. Canvas allows you to turn in multiple flies in one submission. Once you have uploaded your first file, click "Attach Another File" to upload your second file, and so on.

Turn in your files `letters.cpp`, fractions.cpp, `flipnums.cpp` and `evaluator.cpp` to Canvas.

Program 1 - Printing Letters

Following the success of you cake slicing program, Patrick has offered you a state in his new startup. He's not quite sure what the startup is going to sell. However, he has been reliably informed that startup names should never have vowels and has decided to go with PTRK. He now needs to label all things in his new office with the letters P, T, R, and K in various sizes. Since you are the CTO of this new company,it has fallen upon you to write a C++ program to print the letters in the same ASCII Art style that we practiced in class. We will use the same rules - a square Canvas of odd size of at least 7.

Please ensure your program conforms to the following requirements.

1. This program should be called `letters.cpp`
2. Write functions called `printP, printT, printR` and `printK`. Each of these functions will take 1 parameter - the letter's size and return nothing. The functions will each print the corresponding letter (7x4 = 28 points)
3. Write a function called `callLetters`. This function will take 2 parameters - the letter and the size, and return nothing. In this function, use a selection statement (if...else or switch) to call the function to print the given letter. (5 points)
4. In main, accept the size of the letter (in the number of lines) from the user. This number should be an odd number greater than or equal to 7. If the value entered is invalid, tell the user so, and ask for another one. Repeat until you get a valid size. (4 points)
5. Accept the letter to be printed from the user. If the letter is P, T, R, or K, go to the next step. If not, tell the user that the letter is invalid, and ask for another one. Repeat until you get a valid letter. (4 points)
6. Call the `callLetters` function for the given letter and size.
7. Repeat the entire process (steps 4, 5, and 6) if the user indicates they wish to continue. (4 points)
8. Make sure you add comments to explain the logic. (5 points)
9. You may only use the iostream library.

Sample Run

```
Enter the size: 8
Invalid size. Enter the size again: -7
Invalid size. Enter the size again: 3
Invalid size. Enter the size again: 9
Enter the letter: X
Invalid letter: Enter the letter again: 4
Invalid letter: Enter the letter again: &
Invalid letter: Enter the letter again: P

*********
*       *
*       *
*       *
*********
*
*
*
*

Do you want to continue? (Y or N): Y
Enter the size: 12
Invalid size. Enter the size again: 11
Enter the letter: E
Invalid letter: Enter the letter again: T
```

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

Do you want to continue? (Y or N): Y
Enter the size: 11
Enter the letter: K

*         *
*       *
*     *
*   *
* *
*
* *
*   *
*     *
*       *
*         *

Do you want to continue? (Y or N): Y
Enter the size: -9
Invalid size. Enter the size again: 13
Enter the letter: L
Invalid letter: Enter the letter again: R

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

```
Do you want to continue? (Y or N): N
```

Program 2 - Superpartient Fractions

Patrick Star is convinced you a genius. He is now on a mission to become the next big viral musical sensation in town. Patrick has, in his infinite wisdom, decided that everyone talking to him shall only count in musically relevant superpartient fractions. He will randomly yell out a number and you need to show him all the superpartient fractions less than 2 with that number in the denominator.

Write a C++ program to print all the superpartient fractions less than 2 for the given denominator. A superpartient fraction is a fraction of the form $\frac{n+a}{n}$, where n and a are co-prime. Co-prime numbers are numbers that do not share any common factors besides 1. Make sure your program conforms to the following requirements:

1. This program should be called fractions.cpp
2. Accept the denominator from the user (as an integer). (5 points)
3. Make sure the number is at least 3. If it is not, the program stops. (5 points).
4. Go from 1 to the number. If the resulting fraction formed is superpartient, print it. Otherwise, print the common factors between the numerator and denominator. (35 points)
5. Add comments wherever necessary. (5 points)
6. You may only use the iostream library.

Sample Runs

It is OK if you have a comma at the end of the list of common factors.

Sample Run 1

```
Enter the denominator: 8
The possible numbers are:
9 - 9/8
10 - 2 common
11 - 11/8
12 - 2, 4 common
13 - 13/8
14 - 2 common
15 - 15/8
```

Sample Run 2

```
Enter the denominator: 30
The possible numbers are:
31 - 31/30
32 - 2 common
33 - 3 common
34 - 2 common
35 - 5 common
```

```
36 - 2, 3, 6 common
37 - 37/30
38 - 2 common
39 - 3 common
40 - 2, 5, 10 common
41 - 41/30
42 - 2, 3, 6 common
43 - 43/30
44 - 2 common
45 - 3, 5, 15 common
46 - 2 common
47 - 47/30
48 - 2, 3, 6 common
49 - 49/30
50 - 2, 5, 10 common
51 - 3 common
52 - 2 common
53 - 53/30
54 - 2, 3, 6 common
55 - 5 common
56 - 2 common
57 - 3 common
58 - 2 common
59 - 59/30
```

Sample Run 3

```
Enter the denominator: -7
Number is too small. Goodbye.
```

Problem 3: Number Flipping

Unfortunately, Patrick's music career didn't pan out. Now, Patrick has decided that he needs to make a name for himself has a math tutor with services offered through his startup. To demonstrate his abilities, Patrick has put together a very complex equation. When given a number, Patrick will calculate the difference between the number and its reverse, and subtract from that a random number at most as large as the original number. After confusing a lot of people, he gets you to design a program to do it for him instead.

Write a C++ program that, given a number 'num' will calculate the required value. For example, if the number were 1234, you need to calculate

$$|| 1234 - 4321 | - random\%1234 |$$

Please make sure you conform to the following requirements:

1. Call this program `numberflip.cpp`
2. Write a function called `reverseDiff` that takes an integer as a parameter and returns the difference between the number and its reverse. Please note the "difference" is always positive. (20 points)

3. Write a function called `calcVal` that accepts a number as a parameter, calculates the required value for that number, and returns it. This function should call the `reverseDiff` function and the Random Number Generator (RNG) to get a random number. (10 points)
4. In main, first get a seed from the user to seed the RNG. (3 points)
5. Then, accept a series of numbers from the user. Stop if the number entered is 0. Use the `calcVal` function to find the required value for each of the numbers as they are entered, print them. (10 points)
6. Finally, calculate the average value and print it. (2 points)
7. Make sure you add comments to explain your logic. (5 points)
8. You are only allowed the `iostream, cmath`, and `cstdlib` libraries for this problem.

Sample Run

Please note that your output might be different due to the use of the RNG.

```
Enter the seed for the RNG: 6746464
Enter a number: 501
The required value is 152
Enter the next number: 874
The required value is 542
Enter the next number: 1236
The required value is 1025
Enter the next number: 984
The required value is 769
Enter the next number: 25478
The required value is 19227
Enter the next number: 0
The average is 4343
```

Problem 4 - Extra Credit: Evaluating Business Performance

Mr. Krabs, now aware of Patrick's increasing reputation as a entrepreneur (that is willing to consult for his offered fees), has hired him to calculate if business at the Krusty Krab is at the optimum level. This can be modeled on a Poisson distribution. Patrick needs to calculate the cumulative Distribution function of the Poisson distribution using the formula:

$$F(x,\lambda) = e^{-\lambda}\sum_{k=0}^{x}\frac{\lambda^k}{k!} = e^{-\lambda}\left(\frac{\lambda^0}{0!}+\frac{\lambda}{1!}+\frac{\lambda^2}{2!}+\frac{\lambda^3}{3!}+\ldots+\frac{\lambda^x}{x!}\right)$$

Patrick has offered you a bonus to do this for him. Write a C++ program to evaluate the cumulative distribution function of the Poisson Distribution. For this program, you may assume the value of e is 2.71828. Make sure you conform to the following requirements.

1. Call this program `evaluator.cpp`
2. Write a function called `factorial` that returns the factorial of a number. (10 points)

3. Write a function called `sum` that takes ‘x’ and ‘λ’ as parameters and calculates F(x,λ) (25 points)
4. Oversimplified definition: ‘x’ can be considered to represent the number of customers in the restaurant at a time, and ‘λ is analogous to the expected number of customers. The CDF then can represent the probability having less than the expected number of customers at a time.
5. You may use the `pow` function to raise a number to a power.
6. In the main function, accept ‘λ’ and various values of ‘x’ from the user, use the `sum` function to calculate the required CDF probability for each ‘x’, and print it. Stop if the user enters a negative value for x. (8 points)
7. If the CDF for a set of inputs was less than 0.5, then the restaurant is not at optimum (either due to too few or too many customers). If is is so, print it. (2 points).
8. You may assume that ‘x’ will always be an integer.
9. You may assume that λ will always be a positive number.
10. You may use the `iostream`, `iomanip` and `cmath` libraries.
11. Make sure you add comments to explain your logic. (5 points)

1.1 Sample Run

```
Enter the expected number of customers: 25
Enter the actual number of customers : 10
Probability of less than 10 customers was 0.0006
Under optimum performance
Enter the actual number of customers : 35
Probability of less than 35 customers was 0.9776
Enter the actual number of customers : 19
Probability of less than 19 customers was 0.1336
Under optimum performance
Enter the actual number of customers : 27
Probability of less than 27 customers was 0.7002
Enter the actual number of customers : -5
```

General Requirements

1. Include the header comment with your name and other information on the top of your files.
2. Please make sure you’re only using the concepts already discussed in class. Please restrict yourself to variables, operators,selection statements, loops and functions. Using arrays, strings, C++ 11 and up features or anything more advanced will result in a loss of 10 points per program.
3. Each program is worth 50 points.
4. If we have listed a specification and allocated point for it, you will lose points if that particular item is missing from your code, even if it is trivial.

5. **This is individual work. You may NOT collaborate with other students in the course, former students, hire tutors to "help", copy solutions off the internet, or use pay-for solution websites, including but not limited to Chegg, CourseHero, WiseAnt, Bartelby, tutor.com, assorted Social Media groups, whatever GroupMe students might have created, etc.) This includes posting the problem statements on these websites even if you do not use the answer obtained. Doing so is a violation of the Academic Honor Code. Violation of the Honor Code will result in a 0 grade on the program, a reduced letter grade in the course and potentially more serious consequences.**

6. No global variables (variables outside of main())

7. No `goto` statements or the `auto` keyword.

8. All input and output must be done with streams, using the library `iostream`

9. You may only use the `iostream, cstdlib, cmath` and `iomanip` libraries (you do not need any others for these tasks) as indicated in the individual problem requirements.

10. NO C style printing is permitted. (Aka, don't use printf). Use cout if you need to print to the screen.

11. Functions have to be declared above the main function and defined below the main function (as demonstrated in class). Not doing so would result in a loss of 2 points per function.

12. Do not use `break` or `continue` statements in loops. break is allowed only for preventing the fall-though in a `switch` statement. Use of these would result in a loss of 10 points per statement.

13. When you write source code, it should be readable and well-documented (comments).

14. Make sure you either develop with or test with JetBrains CLion (to be sure it reports no compile errors or warnings) before you submit the program.

15. Testing your program thoroughly is a part of writing good code. We give you sample runs to make sure you match our output requirements and to get a general idea of how we would test your code. Matching your outputs for JUST the sample runs is not a guarantee of a 100. We have several extensive test cases.

16. Please make sure you've compiled and run your program before you turn it in. Compilation errors can be quite costly. We take 5 points off per compiler error for the first 9 errors. The 10th compiler error will result in a grade of 0.

17. Only a file turned in through Canvas counts as a submission. A file on your computer, even if it hasn't been edited after the deadline, does not count.

18. The student is responsible for making sure they have turned in the right file(s). We will not accept any excuses about inadvertently modifying or deleting files, or turning in the wrong files.

19. **Program submissions** should be done through the Canvas class page, under the assignments tab (if it's not there yet I'll create it soon.) Do not send program submissions through e-mail – e-mail attachments will not be accepted as valid submissions.

20. The ONLY files you will submit via Canvas are `fractions.cpp, letters.cpp, numberflip.cpp`, and `evaluator.cpp`

21. **General Advice** - always keep an untouched copy of your finished homework files in your email. These files will have a time-stamp which will show when they were last worked on and will serve as a backup in case you ever have legitimate problems with submitting files through Canvas. Do this for ALL programs.