ITU Computer Engineering Department
BLG 223E Data Structures, Fall 2020
Homework #1
Due November 11, 2020 11:59pm

"Big goals start with small steps"

Problem Defintion

π is the irrational mathematical constant obtained by dividing a circle's circumference by its diameter. Initially, mathematicians expected that, after some point, the digits repeated their previous values, thus they could writte π as a repeating decimal number. Perhaps, that is the reason why so many decimal parts of π is calculated. But, that did not happen. Finally, in 1761, mathematician Lambert proved that was irrational. In other words, the circumference of the circle and its diameter had no common measure in rational number. The World's Pi number record was broken by Google employee Emma Haruka Iwao, who managed to calculate the 31.4 trillion digits of the number.

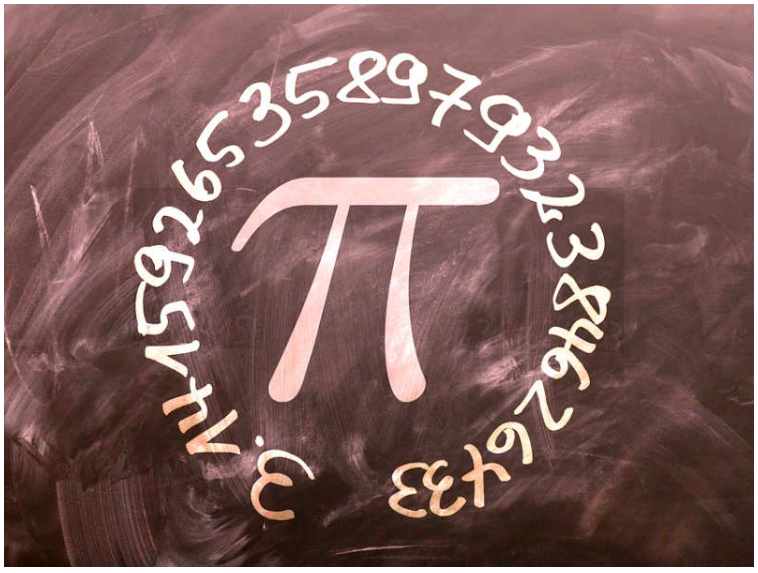

Figure 1: First 25 decimal number of Pi

In this homework, you should write an app that counts how many times you encounter a 2 digits number (let's call it as token) in decimal part of pi. Also, the most common three tokens should be found. A file that contains approximate pi number is shared with you. For example; how many times some of 2 digit numbers repeat in first 100 number of pi's decimal part are given below.

'65' repeats 1 times
'53' repeats 2 times
'35' repeats 1 times
'58' repeats 2 times
'89' repeats 2 times
'97' repeats 3 times

in 3.1415926535 8**97**9323846 2643383279 5028841**97**1 6939937510 5820**97**4944 5923078164 0628620899 8628034825 3421170679

Implementation

1. The necessary libraries and structs for the solution have been given you and explained below. Please pay attention **'do not change this file'** warning in code files. You have to change just `counter.cpp` file and if you need you can add some stuff in `counter.h` file. You are not allowed to include any Standard Template Library (STL) container.

2. Use the below data structure `Token` that is described in **token.h** file. The struct contains a char array `token` that stores the 2 digit number and an integer value `count` that stores the count of repeating in decimal part of π.

```
struct Token{
        char token[TOKEN_LENGTH];
        int count=0;
};
```

3. The header file of the source code that you will implement is given below. You can add new functions or variables if you need, but you are not allowed to drop any of these functions or variables that are declared in the given header files. You have an array `token_array` in `Token` type. This array stores the tokens that you encounenter in decimal part of pi.

```
#define ARRAY_SIZE 100

#include "token.h"

struct Counter{
        Token token_array[ARRAY_SIZE];
        int token_count=0;
        bool contains(char *,char);
        int findindex(Token [],char[]);
        Token *get_most_common_three();
        void read_and_count();
};
```

4. Implement `bool contains(char *,char);`:

   This function checks whether the char array (the first argument) contains the char (the second argument) or not. If it contains, the function returns **true**, otherwise; **false**.

5. Implement: `int findindex(Token [],char[])`:

   This function returns the index of desired char array (the second argument) in Token array (the first argument). If there is no matching, it returns -1.

6. Implement `void read_and_count()`:

   This function reads file that is named as **'pi_approximate'**, that includes approximate pi number, and assign each encountered two digits tokens to `token_array` with its count. **Note that, tokens have to consist of two digits and `token_array` have to contains tokens in only decimal part of pi. For example; '3.' is '.1' are not accepted forms**.

7. Implement: `Token *get_most_common_three()`:

   This function returns a pointer of an `Token` array that includes the most common three tokens.

Main Program, Compilation and Screen Output

- A main program is given in **main.cpp** file. In this way, you can use it to test your implementations. **PLEASE DO NOT CHANGE THIS FILE, OTHERWISE YOUR CODE WILL NOT BE EVALUATED AND WILL BE GRADED AS 0**.

- To develop your implementation, you can use any IDE (Integrated Developmet Environment) such as Visual Studio etc. But before submitting, your program should be compiled and ran on Linux environment using **g++ (version 4.8.5 or later)**. You can test your program on ITU's Linux Server using SSH protocol. To compile the code, use the following command:

```
g++ -std=c++11 -Wall -Werror counter.cpp main.cpp -o hw1
```

- Main program prints out the following messages for a given approximate pi:

```
COUNTER APPLICATION OF PI'S DECIMAL
Choose an operation
D: Display
M: Print Most Common Three
E: Exit

Enter a choice {D,M,E}: D
Please enter the token that you want to see how many times it is
encountered in decimal part in pi number (press'*'for full list):
Token : 05
05 1


COUNTER APPLICATION OF PI'S DECIMAL
Choose an operation
D: Display
M: Print Most Common Three
E: Exit

Enter a choice {D,M,E}: M
The most common three tokens;
62 4
97 3
79 3
```

Your program will be checked by using **Calico(https://bitbucket.org/uyar/calico)** automatic checker. Therefore, make sure you **not to change main.cpp file** You get zero marks if the automatic checker fails.

Submission Rules

- If explanations for this homework is not clear, you can ask your question under the thread that is specially started for homework 1 (About HW1) on the message board for BLG 223E on NINOVA. Please check before writing your question whether your question is asked by someone else.

  Do not share any code or text that can be submitted as a part of an assignment (discussing ideas is okay).

- Make sure you write your name and number in all of the files of your project, in the following format:

  /* @Author
  Student Name: <student_name>
  Student ID : <student_id>
  Date: <date> */

- Only electronic submissions through Ninova will be accepted no later than deadline.

- You may discuss the problems at an abstract level with your classmates, but you should not **share or copy code** from your classmates or from the Internet. You should submit your **own, individual** homework.
- Academic dishonesty, including cheating, plagiarism, and direct copying, is unacceptable.
- Use comments wherever necessary in your code to explain what you did.
- Note that **YOUR CODES WILL BE CHECKED WITH THE PLAGIARISM TOOLS!**