DUE TUESDAY, OCTOBER 4, 2022 by 11:59pm

Introduction

ID Forgery has been a prevalent issue in the city of Einstakt, especially in bars. You are currently working in a government agency that tries to identify individuals sharing the same ID numbers and perform a formal investigation. Each day, your team travels to two random bars to collect the encoded IDs of each bar patron.

Those with the same ID numbers will be approached and invited for an interrogation. At the end of each mission, a report with listings of guilty and innocent individuals will be sent to the senior team members for further processing.

Your task is to write a program to decode the ID numbers, find the decoded ID duplicates (if there are any), and generate a report with decoded IDs of all the guilty and innocent individuals.

Assumptions:

- Each input file may contain a maximum of 100 encoded IDs.
- Bar names will always be “Bar1” and “Bar2”.
- Each encoded and decoded ID may vary in length (8293, 00087, 49, …).
- Each encoded ID contains only numbers and parentheses.
- Each encoded ID has *m* characters, where 1 <= *m* <= 100
- Two different encoded IDs that map to the same decoded ID number are considered duplicates. There will only be at most two people who share the same decoded IDs.
- All ID encodings will always have balanced parentheses.
- There will be no space in each encoded ID.

Rules and Operations

1. To decode each ID, reverse the numbers in each pair of the matching

parentheses, starting from the innermost pair.

2. Find the ID duplicates in both bar locations.
3. In your report, print the decoded IDs of both guilty and innocent individuals in ascending order. The output format is shown in the examples below.

<u>Note:</u> You must use stacks and linked lists to implement your solution. All linked list operations must be implemented using recursion!!! Points will be deducted for using iterative implementations.

Examples

<u>Example 1:</u>

Input11.txt

Bar1

10(01)

(4321)

Bar2

(20)02

Bar1

(20)21

Bar2

3(021)

(4321)

Output11.txt

```
Guilty:
1234            // All ID(s) sorted in ascending order
Innocent:
0202
0221            // All ID(s) sorted in ascending order
1010
3120
```

Command line:

./decode “input=input11.txt;output=output11.txt” **or** ./decode input=input11.txt output=output11.txt

Example 2:

Input12.txt

```
Bar2
10(01)

Bar1
(01)10

Bar2
20(12)

Bar1
2(20)1
(4(23)1)
```

```
Bar2
1(432)
```

Output12.txt

```
Guilty:
1010                    // All ID(s) sorted in ascending order
1234
2021
```

Command line:

./decode “input=input12.txt;output=output12.txt” **or** ./decode input=input12.txt output=output12.txt

Example 3:

Input13.txt

```




Bar2

(75(700)1)3

35(748648)

13(05)6(7(6(83)00)1)



Bar1

4(21)5(37(9600))78

1005(3304)
```

```
Bar2

64(42)00587611(47)

(3001)

(78)407

0000000000000526(543)

6(542)1
```

Output13.txt

```
Innocent:
1003
62451
87407
0000000000000526345
1700573                          // All ID(s) sorted in ascending order
10054033
35846847
135061638007
412596007378
64240058761174
```

Command line:

./decode “input=input13.txt;output=output13.txt” **or** ./decode input=input13.txt output=output13.txt

Important direction to submit your GA1:

We expect every student of each group will participate to solve the problem and discuss with each other. ***We will count the lowest grade of a group for every group member*** , so make sure everyone in your team is submitting the same copy. The purpose of this GA is to learn the basic functionalities of Stack and Linked List. If someone has trouble understanding Stack or Linked List, they should discuss with their friends to get a clear understanding. You are encouraged to help your friends, but do not copy paste from other groups. If you help, it will also help you gain a deeper understanding of the topics.

Every student of each group needs to submit ***the same copy*** of the solution. GA1 needs to be turned in to our Linux server. Make sure to create a folder under your root directory, name it ***"ga1"*** (name must be in lower case and exact), only copy your .cpp and .h files to this folder, no test case or other files needed. If you use ArgumentManager.h, don't forget to turn it in, too. ***GAs will be graded only once.*** You will not get the chance of resubmission after grading. So, make sure you are submitting the correct solution before the due date.