CS3230: Assignment 2
Galactus :The Great Attractor

1 Objective

The purpose of this assignment is for you to practice the following skills which you are expected to master in this course:

- Algorithm design;
- Correctness analysis;
- Efficiency analysis;
- Implementation of algorithms.

2 Problem

Background

You are Galactus, the conqueror of Galaxies. You've fallen for Gravitina, enchanted by her omnipresent gravitational pull. You want to go all the way and make a ring out of all your galaxies to bedazzle her.
You have galaxies of all shapes spiral(S), elliptical(E), lenticular(L), and irregular(I). The irregular galaxies do not fit in a ring formation and cannot be a part of the ring. Hence the ring must be made of all spiral, elliptical and lenticular galaxies such that no two galaxies of the same shape are adjacent.
The ring arrangement is specified by the shape of the *Crown galaxy* and the arrangement of shapes of the rest of the galaxies. Hence, $(S)LE$ and $S(L)E$ are considered different ring arrangements as the designated Crown galaxy$(paranthesized)$ is different in each case. Two ring arrangements are the same if they are identical when the Crown positions are superimposed. Hence, $(S)LE$ and $LE(S)$ are considered the same ring arrangement as the Crown galaxy is the same and the shapes at ring positions relative to the Crown Galaxy are identical. On the other hand, $L(S)E$ and $LE(S)$ are not the same ring arrangement.
Given a list of the shape of all your galaxies, find the number of all unique ring formations that you could make. This number could be astronomically large, so print the number modulo 10^9

For example if your galaxies are of shape $SLSIL$, the list of possible ring arrangements is $\{(S)LSL, LS(L)S\}$ assuming the first shape in each arrangement is the *Crown galaxy*. Similarly for the shapes $LSIE$, we have the arrangements $\{(L)SE, (S)LE, (S)EL, (L)ES, (E)LS, (E)SL\}$

Your Task

Write a program that computes the number of possible ring arrangements modulo 10^9.

You can write your program either in C/C++ or in Java. The task has been set up in the **CodeCrunch** programming environment, which we are using for the submission of your program. You can access CodeCrunch for either C/C++ or Java through the following two links:

Note:Use the correct link for your submissions! Incorrectly submitted code will not be graded.

`https://codecrunch.comp.nus.edu.sg/task_view.php?tid=3364`

`https://codecrunch.comp.nus.edu.sg/task_view.php?tid=3365`

Input

The input contains several test cases, each of which is a list of the shapes of all your galaxies. Your program must read from the standard input in the following format. Each line is a string where each character represents the shape of one of your galaxies.

Your program needs to read the input from STDIN. Several test cases have been set up on **CodeCrunch** and when you compile and execute your program there the system will tell you whether your program generates the correct results and thus passes the tests.

Note: The final evaluation will be based on separate test cases.

Output

For each test case, your program needs to print a newline with a single integer representing the number of ring arrangement of galaxies (modulo 10^9).

Sample Input - STDIN	**Output for the Sample Input - STDOUT**
ISESIE SESESI LSIE SELS	2 0 6 4

Your program needs to generate a plain text output written to STDOUT. The output must only contain a single line with a single integer for each test case.

Note that your program's output will be scored **automatically** on **CodeCrunch**. It is therefore very important that your output format is as described above. Otherwise your output may be mis-judged.

3 Deliverables

Note All files must follow the proper naming and should be in the format specified. Archives or incorrectly named files will not be graded.

- A Java or C/C++ program with the name "Main.java" or "main.c" (or "main.cpp") submitted via **CodeCrunch**. Please make sure that your program reads its input from STDIN and writes its results to STDOUT.
- A short report (12pt font, at most 3 A4 pages – max. 2 pages are preferred), containing the following:
  - Your name and student number;
  - A description of your algorithm (you can use pseudo-code or plain English);
  - A correctness proof for your algorithm;
  - A time and space complexity analysis for your algorithm;
  - (Optional) Any other information that you feel is important about your algorithm and implementation.

  Your report should be in PDF format and named as U080887X_Ex2.pdf, where U080887X is to be replaced by your student number. Remember to write your name and student number at the beginning of your report. Submit your report to IVLE (see below).

4 Submission

- Submit your program, Main.java, main.c or main.cpp via CodeCrunch before **23:59** on Monday **15 April 2017**. (We will know your program from your login information.)
- Submit your U080887X_Ex2.pdf report to the IVLE Workbin folder called **Ex2** before **23:59** on Saturday **15 April 2017**.
- Late submission: submit to the folder **Ex2Late** before **23:59** on **16 April 2017**, but 30% marks will be deducted for such a late submission. The same applies for the submission of your program.
- Late submission: submit to the folder **Ex2Latest** before **23:59** on **17 April 2017**, but 60% marks will be deducted for such a late submission. The same applies for the submission of your program.
- No submissions will be accepted after **23:59** of **17 April 2016**.

5 Grading Policy

This exercise is worth 15% of your overall grade, and is graded on a 60 point basis, with 30 pts for your program and another 30 pts for your report.

Grading of Program

The program will be automatically graded on the **CodeCrunch** server. Therefore, please make sure that your program is compilable on CodeCrunch (with Java or C/C++). Before you submit your program, please test it on **CodeCrunch** server. **Note that we do not have time to debug your program! No marks will be given for this part if your program does not compile and run on CodeCrunch.**

Note that we will let your program run for a maximum amount of time during scoring. Some programs never finish and hence our automatic testing procedure will terminate such programs after what we think is sufficient time to compute the output.

A single input file with 5 hidden test cases will be used for grading. Another input file with a different set of test cases is already loaded for testing on the **CodeCrunch** server. The final evaluation will be based on the 5 hidden test cases. Each correct output is worth 6 pts. You will gain all the 30 pts of this part only when your program can handle all five input cases.

Grading of Report

The 30 pts of this part are distributed as follows:

- An algorithm that produces the count of ring arrangements(14 pts): The report should include a well-structured pseudo-code explaining the algorithm.
- Correctness proof (8 pts) : A sketch of derivation of the algorithm based on the problem statement should provide an argument for the correctness.
- Complexity analysis (8 pts) : In your report include an analysis of the **time** and **space** complexity of the implementation of your algorithm as submitted on **CodeCrunch**

The main grading criteria are correctness, clarity and rigor.

Hint

You may want to think about overlapping subproblems to get the required result.

Please note: Plagiarism will not be tolerated!