**A Pattern Printing**

Write a program in C that will print a pattern based on the input odd integer **n**. Please see the sample input output.

**Conditions:**

**n is odd integer. And**

**1<=n<=49.**

|  |  |
| --- | --- |
| **Sample input** | **Sample output** |
| 9 | \_\_\_\_$\_\_\_\_  \_\_\_$$$\_\_\_  \_\_$\_$\_$\_\_  \_$\_\_$\_\_$\_  $$$$$$$$$  \_$\_\_$\_\_$\_  \_\_$\_$\_$\_\_  \_\_\_$$$\_\_\_  \_\_\_\_$\_\_\_\_ |
| 13 | \_\_\_\_\_\_$\_\_\_\_\_\_  \_\_\_\_\_$$$\_\_\_\_\_  \_\_\_\_$\_$\_$\_\_\_\_  \_\_\_$\_\_$\_\_$\_\_\_  \_\_$\_\_\_$\_\_\_$\_\_  \_$\_\_\_\_$\_\_\_\_$\_  $$$$$$$$$$$$$  \_$\_\_\_\_$\_\_\_\_$\_  \_\_$\_\_\_$\_\_\_$\_\_  \_\_\_$\_\_$\_\_$\_\_\_  \_\_\_\_$\_$\_$\_\_\_\_  \_\_\_\_\_$$$\_\_\_\_\_  \_\_\_\_\_\_$\_\_\_\_\_\_ |

**B Compact String**

John loves compacting strings. To make a sting compact John have to do the following simple things.

1. Count the consecutive identical symbols.
2. Replace the portion of the consecutive identical symbols with a single corresponding symbol and the count of the symbol.

For example: compact form of the string “**aaabefffg**” is “**a3b1e1f3g1**”.

**Input**

Input will consist of a line with a string. The string will consists of only alphabets. Length of the string will be at most **1000007**.

**Output**

A single line with the expected compact string.

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| aaabefffg | a3b1e1f3g1 |
| dfgrrrtssshgd | d1f1g1r3t1s3h1g1d1 |

**C Convex Polygon**

You will be given an ordered set of 2-D point coordinates. Your task is to detect whether these points form a convex polygon or not. You can skip at most M points but order of the points cannot be changed. First and second point will not be skipped.

You can visit this [link](https://www.mathopenref.com/polygonconvex.html" \l ":~:text=A convex polygon is defined,-gon) is always convex.) to learn about what a convex polygon is.

**Input**:

First line of the input is an integer **N** indicating the number of ordered 2-D points.

Each of the next **N** lines contains a pair of floating point numbers x, y denoting X and Y coordinate of the points respectively. All points are different.

Output:

A single line saying “**Convex polygon exists**” if the given sequential set of points can generate a convex polygon. Otherwise, your code should print: “**No convex polygon possible**”

**Conditions**

**3 <= N <= 100**

**-100000 <= x, y, <= 100000**

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| 7 2  1 2  3 4  12 9  -5 10  0 10  -10 0  -4 5 | No convex polygon possible |
| 5  1 2  3 4  0 10  -5 10  -10 0 | Convex polygon exists |

**D Dual Knight Moves**

Chess is a very popular game and the world is searching for a grand-master like you. In this problem, you will be provided a chess board with a number of chess pieces from the opponent and a single Knight piece from yourself. Each piece consists of some points according to the following table. You can achieve the point if you attack the corresponding pieces of the opponent.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Chess Piece | Pawn | Bishop | Knight | Rook | Queen | King |
| Symbol of the Piece | P | B | K | R | Q | A |
| Points | 1 | 3 | 5 | 3 | 8 | 10 |

Write a C program which can find the maximum number of points that you can achieve with two back-to-back moves of your Knight piece. Note that the opponent will not give any move in this game.

To know about the chess pieces and their moves, go to the link: <https://www.chess.com/terms/chess-pieces>.

**Input**

Input will consist of 8 lines.

*ith* line consists of 8 characters representing the chess pieces in *ith* row. A ’.’ will be used to represent an empty cell in the chess board.

Your Knight piece is represented by the digit 0.

**Output**

Maximum number of points you can achieve.

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| B.KAB...  P..P...K  P.R.P...  ......Q.  ...0..R.  ........  .P...P..  ........ | 13 |

Description

Maximum points can be got by moving from (5,4) to Rook at (3,3) and then to King at (1,4).

**E Array Operations**

Perera wanted to be a doctor, but he became a computer engineer. As he can’t do surgeries as doctor, he likes to have different operations on array. His favorite game is to ask his friends to do different operations on array. In this problem, the following operations are allowed to be done.

1. **I X**: It indicates that all the values in the array should be increased by **X**.
2. **D X**: It indicates that all the values in the array should be decreased by **X**.
3. **P**: It indicates that all the values in the array should be printed.

**Input**

First line contains an integer **N** indicating the size of the array.

Second line contains N space separated integers **a**.

Third line contains M which is the number of operations.

Next M lines will contain the operations asked by sequentially.

**Output**

The integers in arrays should be printed in a line separated by a single space when any print command is taken.

**Constraints**

**1 <= N <= 10^6**

**1 <= M <= 10^4**

**1 <= X <= 100**

**-100 <= a <= 100**

Note that the number of Print operations will be at most 10.

It is confirmed that no number in intermediate state will cross the 32-bit signed integer limit.

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| 5  1 2 3 4 5  5  I 5  D 2  P  I 10  P | 4 5 6 7 8  14 15 16 17 18 |

Clarification of Sample Case:

At the first Command (I 5), the array becomes → 6 7 8 9 10

At the Second Command (D 2), the array becomes → 4 5 6 7 8

At the Third Command (P), the array is printed

At the Fourth Command (I 10), the array becomes → 14 15 16 17 18

At the Fifth Command (P), the array is printed.

**F Catchy Name**

Suppose, Joe assigned you to propose a catchy name for his new website. You want to create a name depending on most frequently used english letters. So you have gathered some english text. Now you need to write a program to suggest a catchy name. To make the name pronounceable, Joe proposed you to add following conditions to your program:

1. Name will start with the most frequent consonant.
2. Second character should be the most frequent vowel.
3. Third character will be the most frequent letter except the most frequent vowel and the most frequent consonant.
4. Fourth character will be the next most frequent vowel or consonant depending on the third character. If the third character is a vowel, the fourth one will be consonant and vice versa.
5. All characters should be different.

If there is multiple possible solutions then you have to generate all the names satisfying Joe’s constraints from a given context in lexicographic order.

**Input**

Multiple lines of English text. Input will be terminated with EOF.

There will be at most **10000000** characters in input file.

**Output**

List of all the names satisfying Joe’s Constraints in block letters in lexicographic order.

If no names can be suggested, then print the following message.

“**Warning!! Insufficient Text!!**”

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| suppose, joe assigned you to propose a catchy name for his new website.  you wanted to create a name depending on most frequently used english letters.  so you have gathered some english text. now you need to write a program to auto-generate a catchy name. | TEON |
| aabc | Warning!! Insufficient Text!! |
| aaaeeeiiibbbcccdddukkkkki | KIAB  KIAC  KIAD  KIBA  KIBE  KICA  KICE  KIDA  KIDE  KIEB  KIEC  KIED |