Q1 CP Expression Evaluation

Problem | Submissions | Leaderboard | Discussions

Write a C program to calculate the result of n expressions written in the form of ***(string1)op(string2)*** where ***op*** represents the operators (+ , - , *, /, %) and ***string1*** and ***string2*** represent two strings. First you have to check whether the two strings are valid numeric constants or not.

If so, apply the operation *op* on both the strings and display the result in a format as per the following rules:

```
1. (string1)+(string2) - output in base 3 format
2. (string1)-(string2) - output in base 3 format
3. (string1)*(string2) - output in base 9 format
4. (string1)/(string2) - output in base 5 format
5. (string1)%(string2) - output in base 7 format
```

If not, print the following results

```
- if string1 is valid : print (decimal equivalent of string1)op(error)
- if string2 is valid : print (error)op(decimal equivalent of string2)
- if both strings are invalid: print (error)op(error)
```

You should define the following functions in your program for doing the calculations

1. **int check(char input[ ])** - checks whether the input strings represent integer constants or not.
2. **int type(char input[ ])** - checks the type of input string (decimal, octal or hexadecimal).
3. **void convert_to_base3(char input[ ], char input2[ ])** - convert input1[] to base 3 format and store the result in input2[].
4. **void convert_to_base5(char input1[], char input2[ ])** - convert input1[] to base 5 format and store the result in input2[].
5. **void convert_to_base7(char input1[], char input2[ ])** - convert input1[] to base 7 format and store the result in input2[].
6. **void convert_to_base9(char input1[], char input2[ ])** - convert input1[] to base 9 format and store the result in input2[].
7. **void add(char input1[], char input2[ ])** - adds the input strings, interpreting them as numeric constants, and stores the result in input1[].
8. **void subtract(char input1[], char input2[ ])** - subtracts input2 from input1, interpreting them as numeric constants, and stores the result in input1[].
9. **void multiply(char input1[], char input2[ ])** - multiplies the input strings, interpreting them as numeric constants, and stores the result in input1[].
10. **void divide(char input1[], char input2[ ])** - divide input1 by input2, interpreting them as numeric constants, and store the result in input1[].
11. **void remainder(char input1[], char input2[ ])** - find the remainder when input1 is divided by input2, interpreting them as numeric constants, and store the result in input1[].

Input Format

The first line represents the number of input expressions n. The following n lines contain one expression each in the format (string1)op(string2). where the maximum length of an expression is 50 characters.

Constraints

- (0 < n < 100)
- The operands ***string1*** and ***string2*** can be of different formats (decimal representation or octal representation or hexadecimal representation).
- Not allowed to use any standard library files other than stdio.h

Output Format

n lines containing the corresponding output for the input expressions.

Sample Input 0

```
3
(100)+(25)
(100)/(25)
(100)%(25)
```

Sample Output 0

```
11122
4
0
```

Sample Input 1

```
4
(0xA)+(12)
(12)+(0xA)
(077)+(100L)
(100KG)+(0XFUL)
```

Sample Output 1

```
211
211
20001
(error)+(15)
```

Sample Input 2

```
1
(NIT)/(Calicut)
```

Sample Output 2

```
(error)/(error)
```

Q2 CP Convert Names

Problem | Submissions | Leaderboard | Discussions

The Department of Computer Science and Engineering conducted a survey among the students to evaluate the learning facilities provided by the Institute in the campus. As part of the survey, the students entered their name along with other details. While analysing the data, it was found that the format of the names entered are not unique. There are irregularities like a few students used only capital letters, some used only small letters, some used a mix of both, some used extra symbols/characters etc. The survey co-ordinator would like to have the names of students in a specific format as given below:

```
1. All names should contain only letters from the English alphabet and
spaces. However there should not be more than one space between any two
words in the name and the name should not be starting with a space. The
extra spaces and other invalid letters/characters should be removed.

2. All names should be in Title Case Format, i.e., all the letters should
be in small cases except the first letter of each word in the name, which
is in capital case. [eg. Suraj Kumar P Yadav]
```

As a volunteer of this survey, you are requested to write a C program to convert all the names into the required format. You will be given the total count of the students and their names as input. You may take each name and convert it to the required format and display its corrected form on the screen.

Input Format

First line contains the total number of students, say N. The next N lines will contain the name of each student. There will not be any name containing more than 50 letters.

Constraints

```
0 < N < 105
```

Output Format

N lines containing the formatted names in the same order as given in the input.

Sample Input 0

```
3
Vyshak D Vijay
ARCHANA KP
Deepak Gupta C. P.
```

Sample Output 0

```
Vyshak D Vijay
Archana Kp
Deepak Gupta C P
```

Explanation 0

1. No changes required
2. Only first letter need to be in capital case
3. Remove the invalid character '.'

Sample Input 1

```
2
Archana K.P.
  Deepak Gupta    C P
```

Sample Output 1

```
Archana Kp
Deepak Gupta C P
```

Explanation 1

1. Remove '.' and make the second letter of the second word to small case
2. Remove spaces in the beginning and extra spaces between the words Gupta & C

Q3 CP Count Grades

Problem | Submissions | Leaderboard | Discussions

Write a C program to calculate the count of each grade in a class of N students. When the total marks obtained by each student are given, then the grades will be calculated as follows:

Suppose **max** is the highest mark among the entered marks. Then the grades of the students are calculated as per the following

Grade	Upper bound	Lower bound
S	<= max	>= 0.90 * max
A	< 0.90 * max	>= 0.80 * max
B	< 0.80 * max	>= 0.70 * max
C	< 0.70 * max	>= 0.60 * max
D	< 0.60 * max	>= 0.50 * max
E	< 0.50 * max	>= 0.40 * max
F	<0.40 * max	>= 0

One student will be awarded a particular grade if his total marks is greater than or equal to the lower bound value and less than or equal to the upper bound value of that grade.

Input Format

```
First line contains total number of students, N.
The following n lines contain the total marks, M.
```

Constraints

0 < N <= 1000

0 <= M <= 1000

Output Format

```
First line : Total number of students who obtained the grade ‘S’
Second line : line : Total number of students who obtained the grade ‘A’
Third line : Total number of students who obtained the grade ‘B’
4th line : Total number of students who obtained the grade ‘C’
5th line : Total number of students who obtained the grade ‘D’
6th line : Total number of students who obtained the grade ‘E’
7th line : Total number of students who obtained the grade ‘F’
```

Sample Input 0

```
6
50
44
80
70
10
20
```

Sample Output 0

```
1
1
0
1
1
0
2
```

Explanation 0

```
max = 80
The grades of the students are
50 -> C
44 -> D
80 -> S
70 -> A
10 -> F
20 -> F
```

Sample Input 1

```
4
97
98
100
99
```

Sample Output 1

```
4
0
0
0
0
0
0
```