Core Java Assessment 2018 (Programming)

General Instructions:

- There are **5** questions with total of **45** points
- Maximum time allowed is **3 hours**
- Internet is strictly **NOT** allowed
- It is highly recommended to use **Eclipse** or **IntelliJ** IDE
- **Exception handling, code efficiency and coding standards** contain weightage

Question 1 (5 points)

Write a program to create and sort a Stack of integers, without converting it into any other data structure.

Question 2 (10 points)

Problem Statement:
Given a string of characters, find the length of longest proper prefix which is also a proper suffix. For eg, **S = abab.** Here, the required length is 2 because, **ab..** is prefix and **..ab** is also a suffix. (Proper prefix/suffix has length which is strictly less than the length of the original string)

Input Details:

- First line of the input contains an integer T denoting the number of test cases. (1<=T<=100).
- Each test case has one line denoting the string of length less than 100000.

Output Details:

- Print the length of longest proper prefix which is also a proper suffix.

Example Test Case:

Input:

```
2
abab
aaaa
```

Output:

```
2
3
```

Question 3 (10 points)

Problem Statement:

Given an array of integers that might contain duplicates, return all possible subsets. Such that:

- Elements in a subset must be in non-descending order.
- The solution set must not contain duplicate subsets.
- The subsets must be sorted lexicographically (as you would see in a regular dictinoalry).

For example, if S= [1,2,2], is the given array, the solution is :

```
[ ],
[1],
[1,2],
[1,2,2],
[2],
[2,2]
```

Input Details:

- First is T, no of test cases. 1<=T<=500
- Every test case has two lines.
- First line is N, size of array. 1<=N<=12
- Second line contains N space separated integers(x). 1<=x<=9.

Output Details:

- One line per test case, every subset enclosed in () and in every set integers should be space separated. (See example test case below)

Example Test Case:

Input:

```
2
3
1 2 2
4
1 2 3 3
```

Output:

```
()(1)(1 2)(1 2 2)(2)(2 2)
()(1)(1 2)(1 2 3)(1 2 3 3)(1 3)(1 3 3)(2)(2 3)(2 3 3)(3)(3 3)
```

Question 4 (10 points)

Problem Statement:
The problem is to count all the possible paths from top left to bottom right of an M*N matrix with the constraint that from each cell you can either move towards right or towards down.

Input Details:

- The first line of input contains an integer T denoting the number of test cases. 1<=T<=30
- Each test case has one line with two integers M and N, where M is the number of rows and N is the number of columns. 1<-M,N<=1000

Output Details:

- Print the number of paths possible.

Example Test Case:

Input:

```
2
3 3
2 8
```

Output:

```
6
8
```

Question 5 (10 points)

Problem Statement:
Given an input string and a dictionary of words, find out if the input string can be segmented into a space-separated sequence of dictionary words.
For example, consider the following dictionary -
{I,like,sam,sung,samsung,mobile,ice,cream,icecream,man,go,mango}.

- If the input is ‘ilike’, it can be segmented as like “i like”.
- If the input is ‘ilikesamsung’, it can again be segmented as ‘i like samsung’ or ‘i like sam sung’.
- If the input is ‘idontlike’, it cannot be segmented, since the dictionary does not contain the workd ‘dont’.

Input Details:

- First line is integer T denoting number of test cases. 1<=T<=100.
- Every test case has 3 lines.
- First line is N number of words in dictionary. 1<=N<=12.
- Second line contains N strings denoting the words of dictionary. Length of
- Each word is less than 15.
- Third line contains a string S of length less than 1000.

Output Details:

- Print 1 is possible to break words, else print 0.

Example Test Case:

Input:

```
2
12
i like sam sung samsung mobile ice cream icecream man go mango
Ilike
12
i like sam sung samsung mobile ice cream icecream man go mango
Idontlike
```

Output:

```
1
0
```