Question - 1

What will be the output of the program?

```
public class X
{
    public static void main(String [] args)
    {
        try
        {
            badMethod(); /* Line 7 */
            System.out.print("A");
        }
        catch (Exception ex) /* Line 10 */
        {
            System.out.print("B"); /* Line 12 */
        }
        finally /* Line 14 */
        {
            System.out.print("C"); /* Line 16 */
        }
        System.out.print("D"); /* Line 18 */
    }
    public static void badMethod()
    {
        throw new RuntimeException();
    }
}
```

○ AB

○ BC

○ ABC

○ BCD

Question - 2

The WHERE clause - WHERE city LIKE '%a_%b' cannot display

○ ab

○ abb

○ a_b

○ a_%b

Question - 3
Java : Collections

Which of the below statements are true about ArrayList and Vector in Java ?

○ Vector can be resized while ArrayList cannot be

○ Vector is synchronized while ArrayList is not

○ ArrayLists can grow but cannot shrink in size, while Vector can both grow and shrink

○ Vectors allow duplicate values while ArrayList doesnot

Question - 4

To delete all pairs of keys and values in a given HashMap, which of the following methods should be used ?

○ clearAll()

○ empty()

○ remove()

○ clear()

Question - 5

Which of the following is true about iterators ?

○ Iterator is an interface

○ Iterator is a member function of a class in the library

○ Iterators are used to traverse through the elements of a collection

○ Iterator is an abstract class used for iterating all the elements of the class

Question - 6
Unit Testing

Select one or more advantages of writing *unit tests*:

○ Simplifies debugging by uncovering bugs early in development

○ Speeds development by simplifying integration

○ Improves design as part of test-driven development

○ Detects new bugs during regression testing

Question - 7
Power Company

During a surge in demand, an electric company temporarily activated a number of generators to provide additional energy. Now that the demand has gone back to normal, they need to shut down at least half of the generators to save money. All of the generators of a particular model are similar and can be controlled as a single unit. It takes *1* minute to deactivate all generators of any one model. Determine the amount of time it will take to deactivate at least half of the generators, that is, the ceiling of *n/2* of them if *n* is the number of generators. The *ceiling* is a float value rounded up to the next higher integer if there are significant values after the decimal, e.g. *6/4 = 1.5, ceiling(1.5) = 2 and 6/3 = 2, ceiling(2) = 2.*

For example, there are *n = 14* generators*, model = [3, 4, 6, 11, 9, 9, 9, 9, 8, 8, 8, 8, 8, 8]*. The ceiling of *n/2 = 14/2 = 7,* so at least *7* generators must be deactivated. One of the optimal solutions is to deactivate two types of generators, models *9* and *8*. The sum of the frequencies of models *9* and *8* is *4 + 6 = 10*, which is *≥ 7*. The answer is *2*.

Function Description
Complete the function *reduceCapacity* in the editor below. The function must return an integer.

reduceCapacity has the following parameter(s):
model: an array of integers, the model numbers of each generator.

Constraints

- $1 \le n \le 10^5$
- $1 \le model[i] \le 10^6$

▼ Input Format For Custom Testing

The first line contains an integer *n*, the size of array *model*.
Each of the next *n* lines contains an integer, *model[i]*, that denotes the model of the i^{th} generator.

▼ Sample Case 0

Sample Input For Custom Testing

```
7
7
10
1
2
7
7
1
```

Sample Output

```
2
```

Explanation

One of the optimal ways is to deactivate the generators having model numbers *7* and *2*. The sum of the frequencies of models *7* and *2* is *4*, which is *≥ ceiling(7/2) = ceiling(3.5) = 4*.

▼ Sample Case 1

Sample Input For Custom Testing

```
9
6
6
6
6
7
7
7
7
5
```

Sample Output

```
2
```

Explanation

One of the optimal ways is to deactivate generators having model numbers *7* and *6*. The sum of their frequencies is *8*, which is *≥ ceiling(9/2) = ceiling(4.5) = 5*.

▼ Sample Case 2

Sample Input For Custom Testing

```
14
3
4
6
11
9
9
9
9
8
8
8
8
8
8
```

Sample Output

```
2
```

Explanation

One of the optimal ways is to deactivate generators having model numbers *9* and *8*. The sum of their frequencies is *10*, which is *≥ ceiling(14/2) = 7*.

Question - 8
Coin Gift

Its Alex's birthday today and Sam has brought a string as a gift. Alex is not happy with the gift and comes up with a plan to exchange it. First, generate one string of size *3* and ask Sam to give as many coins as the number of times the string occurs as a subsequence in Sam's string.

Determine the number of coins Alex will get, i.e. given two strings *alexString* and *samString*, the goal is to find the count of the number of times *alexString* appears as a subsequence in the *samString*. A subsequence is created by eliminating an number of characters from a string, from *0* to the length of the string, without changing the order of the characters retained.

For example :
alexString=ABC
samString=ABCBABC
alexString appears *5* times as subsequence in *samString* at 1-indexed positions *(1, 2, 3), (1, 2, 7), (1, 4, 7), (1, 6, 7), (5, 6, 7).*

Function Description

Complete the function *getSubsequenceCount* in the editor below. This function should return the number of times Alex's string appears as the subsequence in Sam's string.

getSubsequenceCount has the following parameter(s):
alexString **:** string of length 3
samString **:** a string

Constraints

- 1 ≤ | *samString* | ≤ $5*10^5$
- | *alexString* | = 3
- *alexString* and *samString* consist of uppercase English letters, *ascii[A-Z]*

▼ Input Format For Custom Testing

The first line of input contains a string *alexString* of length 3.
The second line of input contains a string *samString.*

▼ Sample Case 0

Sample Input For Custom Testing

```
HRW
HERHRWS
```

Sample Output

```
3
```

Explanation

HRW appears as a subsequence of *HERHRWS* at positions *(1, 3, 6), (1, 5, 6), (4, 5, 6)*

▼ Sample Case 1

Sample Input For Custom Testing

```
ELO
HELLOWORLD
```

Sample Output

```
4
```

Explanation

ELO appears as a subsequence of *HELLOWORLD* at positions *(2, 3, 5), (2, 3, 7), (2, 4, 5), (2, 4, 7)*