Problem: STRONG PASSWORD

Louise joined a social networking site to stay in touch with her friends. The signup page required her to input a *name* and a *password*. However, the password must be *strong*. The website considers a password to be *strong* if it satisfies the following criteria:

- Its length is at least .
- It contains at least one digit.

- It contains at least one lowercase English character.

- It contains at least one uppercase English character.

- It contains at least one special character. The special characters are: `!@#$%^&*()-+`

She typed a random string of length in the password field but wasn't sure if it was strong. Given the string she typed, can you find the minimum number of characters she must add to make her password strong?

Note: Here's the set of types of characters in a form you can paste in your solution:

```
numbers = "0123456789"
lower_case = "abcdefghijklmnopqrstuvwxyz"
upper_case = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
special_characters = "!@#$%^&*()-+"
```

Input Format

The first line contains an integer denoting the length of the string.
The second line contains a string consisting of characters, the password typed by Louise. Each character is either a lowercase/uppercase English alphabet, a digit, or a special character.

Constraints

-

Output Format

Print a single line containing a single integer denoting the answer to the problem.

Sample Input 0

```
3
Ab1
```

Sample Output 0

```
3
```

Explanation 0

She can make the password strong by adding characters, for example, `$hk`, turning the password into `Ab1$hk`which is strong.
 characters aren't enough since the length must be at least .

Sample Input 1

11
#HackerRank

Sample Output 1

1

Explanation 1

The password isn't strong, but she can make it strong by adding a single digit.

Solution:

static int minimumNumber(int cases, String password) {

    //returns number of characters to be added to make the

    //password secure

    int flag=0, flag2=0, flag3=0, flag4=0, add=0;

    String number = "0123456789";

    String lower_case = "abcdefghijklmnopqrstuvwxyz";

    String upper_case = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";

    String special = "!@#$%^&*()-+";

    int length = password.length();

    for(int i=0; i<length; i++)

      {

        for(char c: lower_case.toCharArray())

          {

```java
            if(c==password.charAt(i))

              {flag=1;}

          }

       for(char c: upper_case.toCharArray())

         {

           if(c==password.charAt(i))

             {flag2=1;}

         }

       for(char c: number.toCharArray())

         {

           if(c==password.charAt(i))

             {flag3=1;}

         }

       for(char c: special.toCharArray())

         {

           if(c==password.charAt(i))

             {flag4=1;}

         }


    }


if(flag==0)

  {add+=1;}

if(flag2==0)

  {add+=1;}

if(flag4==0)
```

```
        {add+=1;}

    if(flag3==0)

        {add+=1;}

    if(length+add <6)

        {add+=6-length-add;}

    return add;

}
```

Elegant Solution

```
public class Solution
    {
        static int minimumNumber(int cases, String password)
            {
                int add = 0;

                if(!Pattern.compile("[a-z]").matcher(password).find())
                    add+=1;
                if(!Pattern.compile("[A-Z]").matcher(password).find())
                    add+=1;
                if(!Pattern.compile("\\d").matcher(password).find())
                    add+=1;
                if(!Pattern.compile("\\W").matcher(password).find())
                    add+=1;

                if(add+password.length() < 6)
                    {add+= 6 - password.length() - add;}
                return add;
            }
    }
```

`'Anshul AgGarwal