CHANDIGARH UNIVERSITY

UNIVERSITY INSTITUTE OF NGINEERING

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Submitted By:	**Submitted To:**
Vivek Kumar(21BCS8129)	Mamta Punia(E12337)
Subject Name	Competitive Coding - I
Subject Code	20CSP-314
Branch	Computer Science and Engineering
Semester	5th

Experiment No. - 7

Student Name: Vivek Kumar
Branch: BE-CSE(LEET)
Semester: 5th
Subject Name: Competitive coding - I

UID: 21BCS8129
Section/Group: WM-20BCS-616/A
Date of Performance: 14/10/2022
Subject Code: 20CSP-314

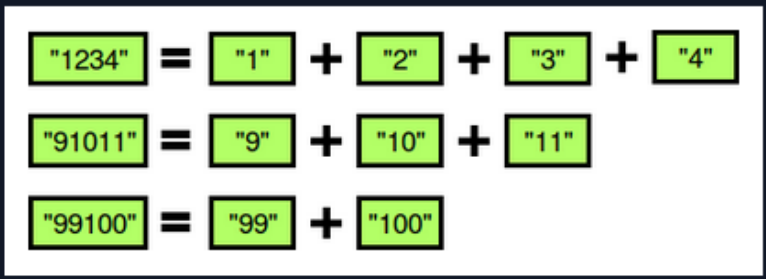
Separate the Numbers

1. Aim/Overview of the practical:

A numeric string, s, is beautiful if it can be split into a sequence of two or more positive integers, $a[1], a[2], \ldots, a[n]$, satisfying the following conditions:

1. $a[i] - a[i-1] = 1$ for any $1 < i \leq n$ (i.e., each element in the sequence is 1 more than the previous element).
2. No $a[i]$ contains a leading zero. For example, we can split $s = 10203$ into the sequence $\{1, 02, 03\}$, but it is not beautiful because 02 and 03 have leading zeroes.
3. The contents of the sequence cannot be rearranged. For example, we can split $s = 312$ into the sequence $\{3, 1, 2\}$, but it is not beautiful because it breaks our first constraint (i.e., $1 - 3 \neq 1$).

The diagram below depicts some beautiful strings:



Perform q queries where each query consists of some integer string s. For each query, print whether or not the string is beautiful on a new line. If it is beautiful, print `YES x`, where x is the first number of the increasing sequence. If there are multiple such values of x, choose the smallest. Otherwise, print `NO`.

2. Task to be done/ Which logistics used:

Function Description

Complete the separateNumbers function in the editor below.

separateNumbers has the following parameter:

- s: an integer value represented as a string

Prints

- string: Print a string as described above. Return nothing.

Input Format

The first line contains an integer q, the number of strings to evaluate.
Each of the next q lines contains an integer string s to query.

Constraints

- $1 \leq q \leq 10$
- $1 \leq |s| \leq 32$
- $s[i] \in [0-9]$

Sample Input 0

```
7
1234
91011
99100
101103
010203
13
1
```

Sample Output 0

```
YES 1
YES 9
YES 99
NO
NO
NO
NO
```

Explanation 0

The first three numbers are beautiful (see the diagram above). The remaining numbers are not beautiful:

- For $s = 101103$, all possible splits violate the first and/or second conditions.
- For $s = 010203$, it starts with a zero so all possible splits violate the second condition.
- For $s = 13$, the only possible split is $\{1, 3\}$, which violates the first condition.
- For $s = 1$, there are no possible splits because s only has one digit.

Sample Input 1

```
4
99910001001
7891011
9899100
999100010001
```

Sample Output 1

```
YES 999
YES 7
YES 98
NO
```

3. Hardware and Software Requirements (For programming-based labs)**:**

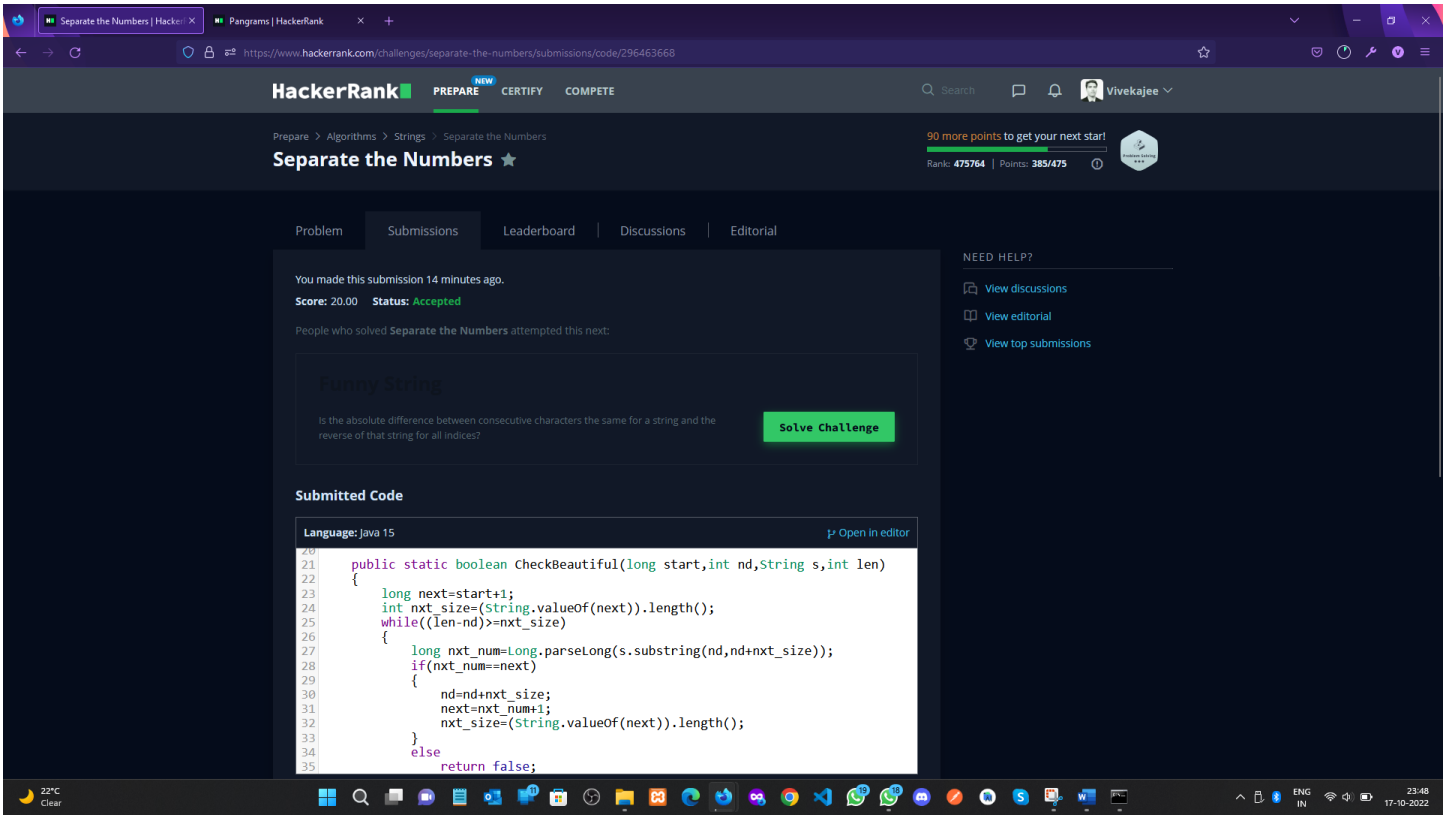
- Laptop or Desktop
- Hacker-Rank Account

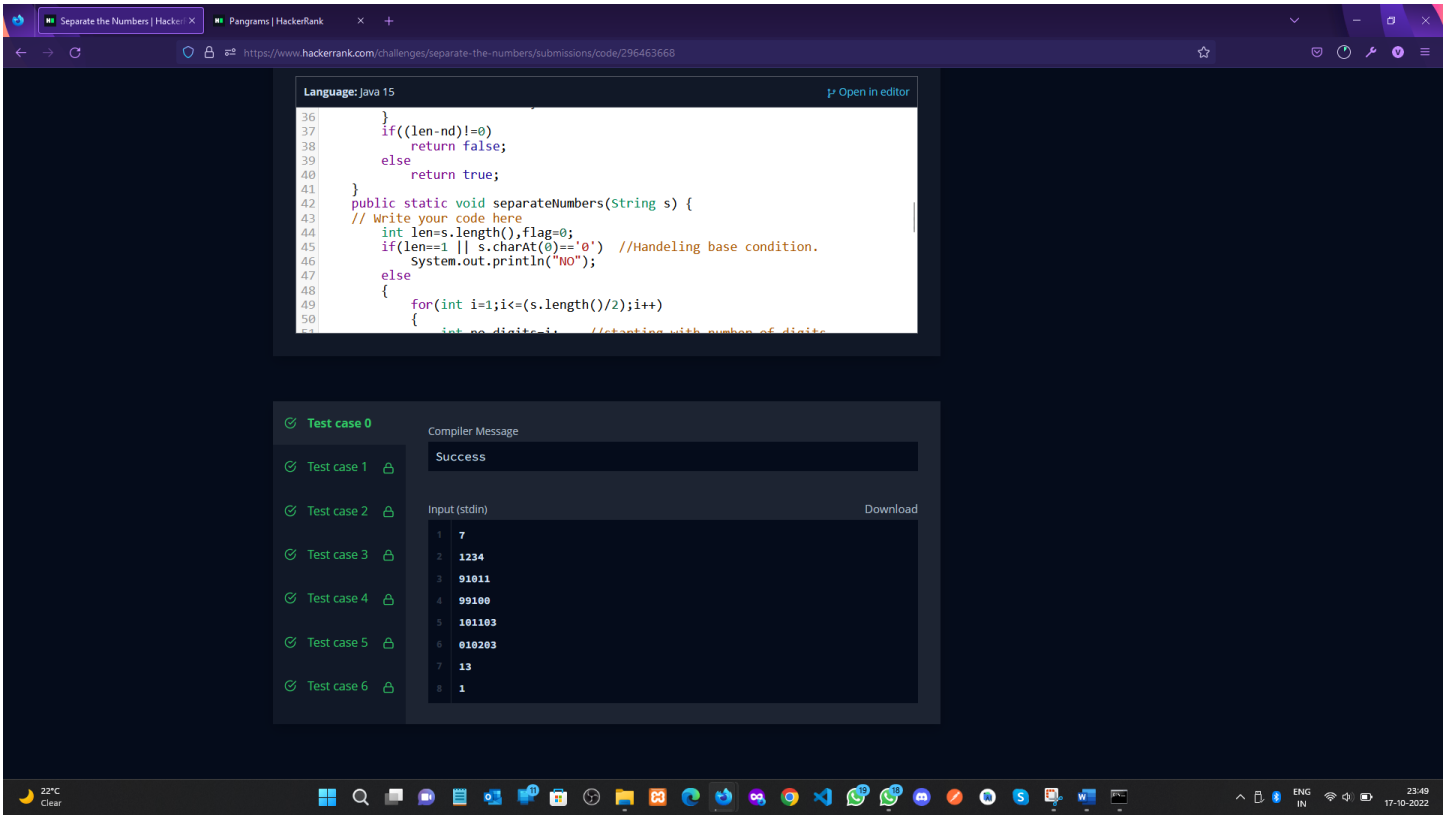
4. Steps for experiment/practical/Code:

```
public static boolean CheckBeautiful(long start,int nd,String s,int len)
{
    long next=start+1;
    int nxt_size=(String.valueOf(next)).length();
    while((len-nd)>=nxt_size)
    {
        long nxt_num=Long.parseLong(s.substring(nd,nd+nxt_size));
        if(nxt_num==next)
        {
            nd=nd+nxt_size;
            next=nxt_num+1;
            nxt_size=(String.valueOf(next)).length();
        }
        else
            return false;
    }
    if((len-nd)!=0)
        return false;
    else
        return true;
}
public static void separateNumbers(String s) {
// Write your code here
    int len=s.length(),flag=0;
    if(len==1 || s.charAt(0)=='0')  //Handeling base condition.
        System.out.println("NO");
    else
    {
        for(int i=1;i<=(s.length()/2);i++)
        {
            int no_digits=i;    //starting with number of digits.
            long start=Long.parseLong(s.substring(0,no_digits));
            boolean ans=CheckBeautiful(start,no_digits,s,len);
            if(ans==true)
            {
                System.out.println("YES"+" "+start);
                flag=1;
                break;
```

```
            }
        }
        if(flag==0)
            System.out.println("NO");
    }
}
```

5. Result/Output/Writing Summary:





Pangrams

1. Aim/Overview of the practical:

A *pangram* is a string that contains every letter of the alphabet. Given a sentence determine whether it is a pangram in the English alphabet. Ignore case. Return either pangram or not pangram as appropriate.

2. Task to be done/ Which logistics used:

Example

s = 'The quick brown fox jumps over the lazy dog'

The string contains all letters in the English alphabet, so return `pangram`.

Function Description

Complete the function pangrams in the editor below. It should return the string `pangram` if the input string is a pangram.

Otherwise, it should return `not pangram`.

pangrams has the following parameter(s):

- string s: a string to test

Returns

- string: either `pangram` or `not pangram`

Input Format

A single line with string s.

Constraints

$0 <$ length of $s \leq 10^3$

Each character of s, $s[i] \in \{a-z, A-Z, space\}$

Sample Input

Sample Input 0

```
We promptly judged antique ivory buckles for the next prize
```

Sample Output 0

```
pangram
```

Sample Explanation 0

All of the letters of the alphabet are present in the string.

Sample Input 1

```
We promptly judged antique ivory buckles for the prize
```

Sample Output 1

```
not pangram
```

Sample Explanation 0

The string lacks an `x`.

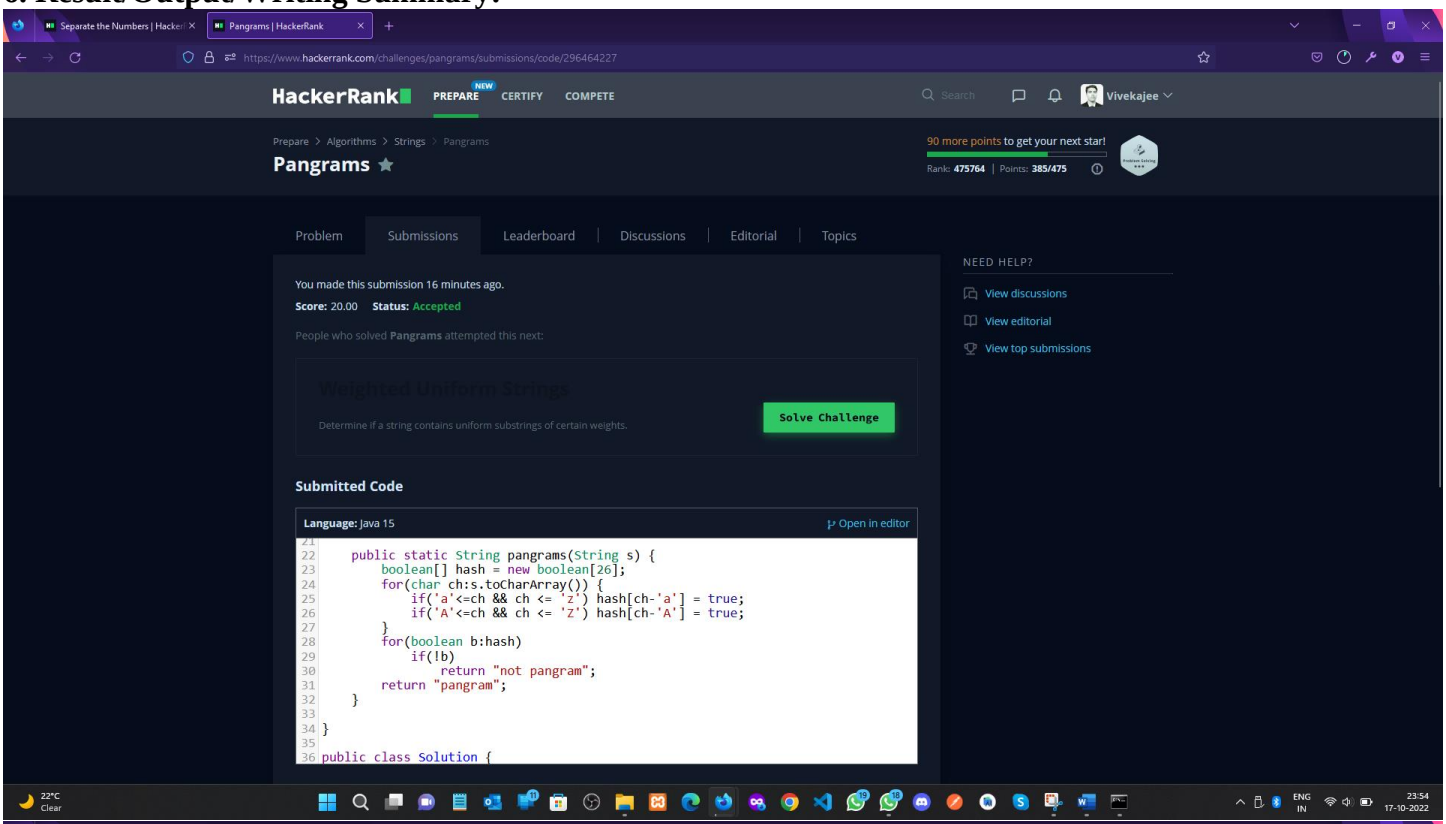
3. Hardware and Software Requirements (For programming-based labs):

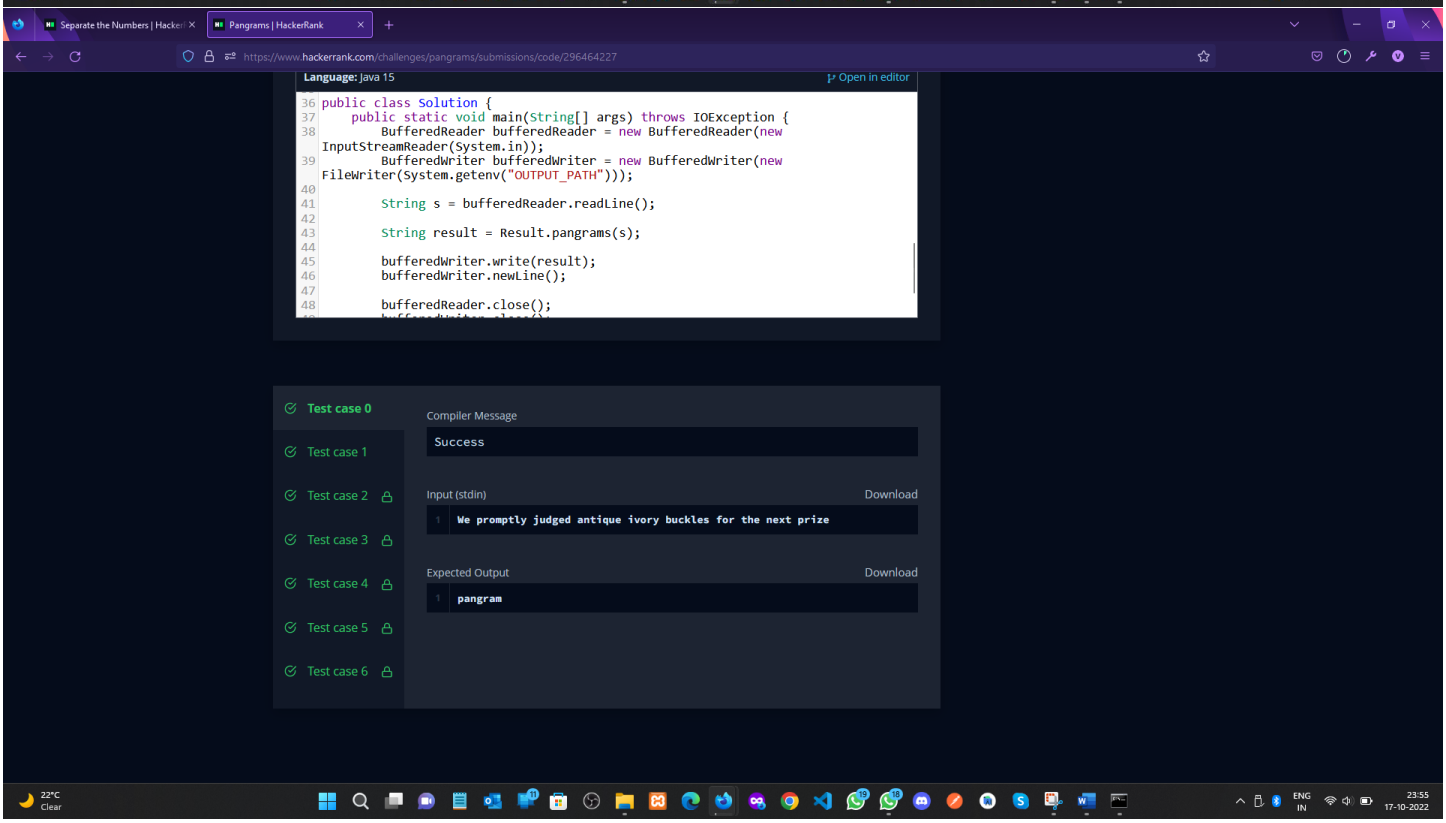
- Laptop or Desktop
- Hacker-Rank Account

4. Steps for experiment/practical/Code:

```
public static String pangrams(String s) {
        boolean[] hash = new boolean[26];
        for(char ch:s.toCharArray()) {
            if('a'<=ch && ch <= 'z') hash[ch-'a'] = true;
            if('A'<=ch && ch <= 'Z') hash[ch-'A'] = true;
        }
        for(boolean b:hash)
            if(!b)
                return "not pangram";
        return "pangram";
    }
```

6. Result/Output/Writing Summary:





Learning outcomes (What I have learnt):

a. Learnt about String concept.

b. Learnt about Number separation concept from string.

c. Learn about the pangram concept.

Evaluation Grid (To be created per the faculty's SOP and Assessment guidelines):

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1.	Worksheet completion including writing learning objectives/Outcomes. (To be submitted at the end of the day).		
2.	Post-Lab Quiz Result.		
3.	Student Engagement in Simulation/Demonstration/Performance and Controls/Pre-Lab Questions.		
	Signature of Faculty (with Date):	Total Marks Obtained:	