91) String Processing - V

Write a program to read a string and also a number N. Form a new string made up of n repetitions of the last n characters of the String. You may assume that n is between 1 and the length of the string.

Include a class UserMainCode with a static method **returnLastRepeatedCharacters** which accepts the string and the number n. The return type is the string as per the problem statement.

Create a Class Main which would be used to accept the string and integer and call the static method present in UserMainCode.

Input and Output Format:
Input consists of a string and integer.
Output consists of a string.
Refer sample output for formatting specifications.

Sample Input 1:
Hello
2
Sample Output 1:
lolo

Sample Input 2:
Hello
3
Sample Output 2:
llollollo

```
public class Main {
	public static void main(String[] args) {
		Scanner sc=new Scanner(System.in);
		String s1=sc.next();
		int n=sc.nextInt();
		System.out.println(User.returnLastRepeatedCharacters (s1,n));
	}
}

public class User {
	public static String returnLastRepeatedCharacters (String s1,int n)
	{
		StringBuffer sb=new StringBuffer();
		for(int i=0;i<n;i++)
		sb.append(s1.substring(s1.length()-n));
		 return sb.toString();
	}
}
```

92) Regular Expression - III

Given a string (s) apply the following rules.
1. String should not begin with a number.
If the condition is satisifed then print TRUE else print FALSE.
Include a class UserMainCode with a static method **validateString** which accepts the string. The return type is the boolean formed based on rules.
Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:
Input consists of a string.
Output consists of TRUE or FALSE .
Refer sample output for formatting specifications.

Sample Input 1:
ab2
Sample Output 1:
TRUE

Sample Input 2:
72CAB
Sample Output 2:
FALSE

```
public class Main {
	public static void main(String[] args) {
		Scanner sc=new Scanner(System.in);
		String s1=sc.next();
		boolean b=User.validateString(s1);
		System.out.println(b);
	}
}
```

```
public class User {
	public static boolean validateString(String s1)
	{
		boolean b= false;
		if(!Character.isDigit(s1.charAt(0)))
			b= true;
		else
			b= false;
		return b;

	}
}
```

93) 3String Processing - TrimCat

Write a program to read a string and return a new string which is made of every alternate characters starting with the first character. For example NewYork will generate Nwok, and Samurai will generate Smri.

Include a class UserMainCode with a static method getAlternateChars which accepts the string. The return type is the modified string.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:
Input consists of a string.
Output consists of string.
Refer sample output for formatting specifications.

Sample Input 1:
Hello

Sample Output 1:
Hlo

```
public class Main {


	public static void main(String[] args) {
		Scanner sc=new Scanner(System.in);
		String s1=sc.next();
		System.out.println(User.fetchUserName (s1));
	}
}

public class User {
	public static String fetchUserName (String s1)
	{
		StringBuffer sb=new StringBuffer();
		for( int i=0;i<s1.length();i=i+2)
			sb.append(s1.charAt(i));
		return sb.toString();

	}
}
```

94) 2 String Processing - Username

Write a program to read a valid email id and extract the username.

Note - user name is the string appearing before @ symbol.

Include a class UserMainCode with a static method fetchUserName which accepts the string. The return type is the modified string.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:
Input consists of a string.
Output consists of string.
Refer sample output for formatting specifications.

Sample Input 1:
admin@xyz.com

Sample Output 1:
admin

```
public class Main {


	public static void main(String[] args) {
		Scanner sc=new Scanner(System.in);
		String s1=sc.next();
		System.out.println(User.fetchUserName (s1));
	}
}
public class User {
	public static String fetchUserName (String s1)
	{
		boolean b=false;
		StringTokenizer st=new StringTokenizer(s1,"@");
		String name=st.nextToken();

		return name;

	}
}
```

95) 1 String Processing - VII

Write a program to read a two strings and one int value(N). check if Nth character of first String from start and Nth character of second String from end are same or not. If both are same return true else

return false.
Check need not be Case sensitive
Include a class UserMainCode with a static method **isEqual** which accepts the two strings and a integer n. The return type is the TRUE / FALSE.
Create a Class Main which would be used to read the strings and integer and call the static method present in UserMainCode.

Input and Output Format:
Input consists of two strings and an integer.
Output consists of TRUE / FALSE .
Refer sample output for formatting specifications.

Sample Input 1:
AAAA
abab
2

Sample Output 1:
TRUE

Sample Input 2:
MNOP
QRST
3

Sample Output 2:
FALSE

```
public class Main {


	public static void main(String[] args) {
		Scanner sc=new Scanner(System.in);
		String s1=sc.next();
		String s2=sc.next();
		int n=sc.nextInt();
		boolean b=User.isEqual(s1,s2,n);
		System.out.println(b);

	}

}


public class User {
	public static boolean isEqual(String s1,String s2, int n)
```

```
	{
		boolean b=false;
		String i= s1.toLowerCase();
		String j= s2.toLowerCase();
		if(i.charAt(n-1)==j.charAt(n))
			b=true;
		else
			b=false;

		return b;

	}
}
```

96) Largest Difference

Write a program to read a integer array, find the largest difference between adjacent elements and display the index of largest difference.

EXAMPLE:

input1: {2,4,5,1,9,3,8}

output1: 4 (here largest difference 9-1=8 then return index of 9 ie,4)

Include a class UserMainCode with a static method **checkDifference** which accepts the integer array. The return type is integer.

Create a Class Main which would be used to accept the integer array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of an integer n which is the number of elements followed by n integer values.

Output consists of integer.

Refer sample output for formatting specifications.

Sample Input 1:

7
2
4
5
1
9
3
8

Sample Output 1:

4

```
public class Main {
```

```
	public static int getDiffArray(int[] n1){
		int n2,n3=0,n4=0,i;
		for(i=0;i<n1.length-1;i++){
			n2=Math.abs(n1[i]-n1[i+1]);
			if(n2>n3){
				n3=n2;
				n4=i+1; }}
		return n4;
	}
	public static void main(String[] args) {
		int[] n1={2,4,5,1,9,3,8};
		System.out.println(getDiffArray(n1));
	}
}
```