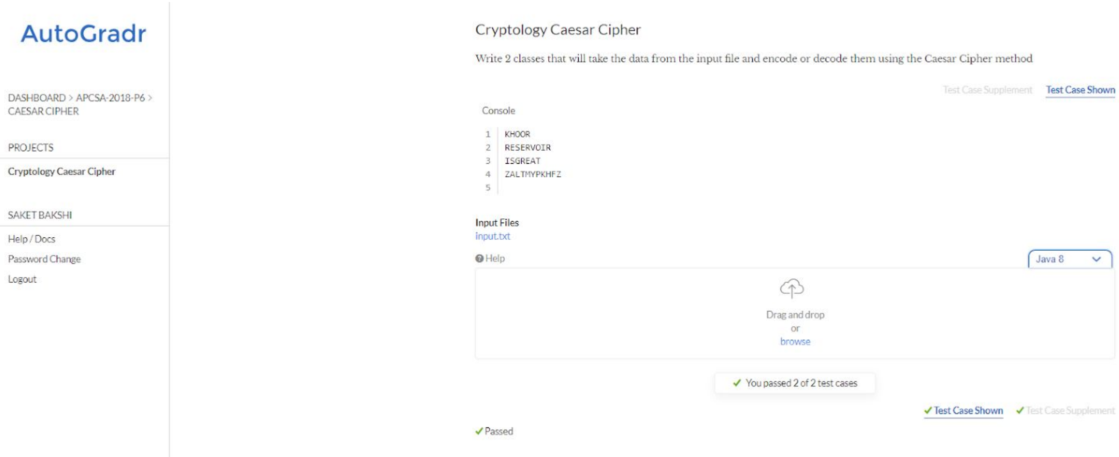


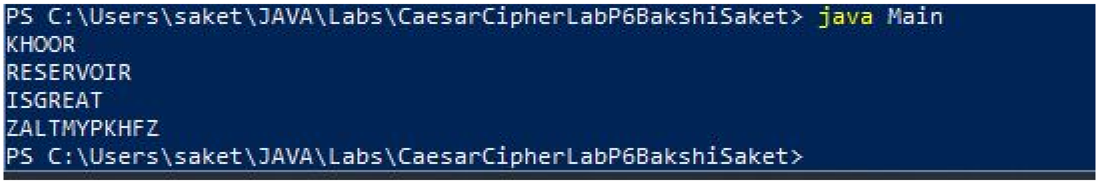
Cryptology Caesar Cipher

Write 2 classes that will take the data from the input file and encode or decode them using the Caesar Cipher method

```
KHOOR
RESERVOIR
ISGREAT
ZALTMYPKHFZ
```

✓ You passed 2 of 2 test cases

✓ Passed



```
PS C:\Users\saket\JAVA\Labs\CaesarCipherLabP6BakshiSaket> java Main
KHOOR
RESERVOIR
ISGREAT
ZALTMYPKHFZ
PS C:\Users\saket\JAVA\Labs\CaesarCipherLabP6BakshiSaket>
```

```
/* Saket Bakshi. Mr. Caces. AP Computer Science A. Due 15 November 2018.
This class makes the CaesarCipher class for decrypting or encrypting phrases with the Caesar method.
*/
public class CaesarCipherLabP6BakshiSaket
{
	//instance variables
	private String encodeOrDecode;
	private int movement;
	private String input;

	private String output;

	/** This creates a CaesarCipher class object that recognizes to encode or decode a phrase, how much to transfer letters, and what to transfer.
	There is one method to do the decryption/encryption, which also prints the output.
		@param encodeOrDecode the String that tells to encode or decode the input
		@param movement how much to encrypt or decrypt
		@param input what to decrypt or encrypt
	*/
	public CaesarCipherLabP6BakshiSaket(String encodeOrDecode, int movement, String input)
	{
		this.encodeOrDecode = encodeOrDecode;
		this.movement = movement;
		this.input = input.replace(" ",""); //deletes spaces in the input
```

```
        this.output = "";
    }

    /** This method decrypts or encrypts the input
    */
    public void runThroughInput()
    {
        int intTemp; //integer for value of the Unicode alphabet
        char temp; //character to convert
        String letterConvert=""; //once the character is converted, it is saved to this String
        int length = this.input.length(); //the length of the input to convert

        for (int i = 0; i < length; i++) //does a loop for each character of the input to
convert
        {
            // A is 65 and z is 90 in unicode
            temp = this.input.charAt(i); //takes character in input
            intTemp = (int)temp; //takes integer value of Unicode character

            if(intTemp >= 65 && intTemp <= 90) //disregards any non-Uppercase
alphabet
            {
                if(encodeOrDecode.equals("E")) //code for encrypting
                    intTemp = intTemp + movement;
                else if(encodeOrDecode.equals("D")) //code for decrypting
                    intTemp = intTemp - movement;
                //loops letter if it goes before a or after z
                if(intTemp < 65) //makes sure to realign letters if below the
Unicode value for A
                    intTemp += 26;
                else if(intTemp > 90) //makes sure to realign letters if above the
Unicode value for Z
                    intTemp -=26;
                //places the new integer value for the character into a char
variable and then converts the char into a String
                temp = (char)intTemp;
                letterConvert = Character.toString(temp);
                this.output = this.output + letterConvert; //adds the char's String to
the output
            }
        }
        System.out.println(this.output); //prints the end output
    }
}
/* Saket Bakshi. Mr. Caces. AP Computer Science A. Due 15 November 2018.
This class tests the CaesarCipher class
*/
import java.util.Scanner;
import java.io.File;
import java.io.FileNotFoundException;

public class CaesarCipherLabTester
{
    public static void main(String[] args) throws FileNotFoundException
    {
        //scans in file
        File inputFile = new File("input.txt");
        Scanner inputScanner = new Scanner(inputFile);

        do //runs through the loop until all lines of code have been converted
        {
            String encodeOrDecode = inputScanner.next(); //takes first String in the
line for knowing to encrypt or decrypt

            int movement = inputScanner.nextInt(); //takes next Integer for seeing net
movement of decryption or encryption
            if(movement > 26) //brings movements above 26 down to a range below
26
            {
                do
                {
                    movement -=26;
                } while(movement > 26);
            }
            else if(movement < -26) //brings movements below -26 up to a range
above -26
            {
                do
                {
                    movement +=26;
                } while(movement < -26);
            }

            String input = inputScanner.nextLine().toUpperCase(); //converts all
inputs to uppercase
```

```
            CaesarCipherLabP6BakshiSaket fml = new
CaesarCipherLabP6BakshiSaket(encodeOrDecode, movement, input); //creates
CaesarCipherLab objects
            fml.runThroughInput(); //runs through the encryption or decryption
        } while(inputScanner.hasNext());

    }
}
```