```
import java.util.Scanner;
public class Main {
    public static void main(String[] args){
        Scanner in=new Scanner(System.in);
        System.out.print("Enter the string: ");
        String str=in.nextLine();
        char[] ch=new char[str.length()];
        for(int i=0;i<str.length();i++){
            ch[i]= str.charAt(i);
        }
        for(int i=0;i< ch.length;i++){
            int num=0;
            if(ch[i]>='0'&&ch[i]<='9'){
                char temp=ch[i-1];
                for(int j=i;j<ch.length;j++){
                    if(ch[i]>='0'&&ch[i]<='9'){
                        num=(num*10)+ch[i]-48;
                    }
                    else{
                        break;
                    }
                    i++;
                }
                for (int k=0;k<num;k++){
                    System.out.print(temp);
                }
            }
        }
    }
}
```

```
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
```

```java
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter input string: ");
        String input1 = scanner.nextLine();
        String compressed1 = compressString(input1);
        System.out.println("Input: " + input1);
        System.out.println("Output: " + compressed1);
        scanner.close();
    }
    private static String compressString(String input) {
        StringBuilder compressed = new StringBuilder();
        int count = 1;
        for (int i = 0; i < input.length() - 1; i++) {
            if (input.charAt(i) == input.charAt(i + 1)) {
                count++;
            } else {
                compressed.append(input.charAt(i));
                if (count > 1) {
                    compressed.append(count);
                }
                count = 1;
            }
        }
        compressed.append(input.charAt(input.length() - 1));
        if (count > 1) {
            compressed.append(count);
        }
        return compressed.toString();
    }
}
```

```java
import java.util.Scanner;
public class NumberToWords {
    private final String[] units = {"", "One", "Two", "Three", "Four", "Five", "Six", "Seven",
            "Eight", "Nine"};
```

```
private final String[] teens = {"", "Eleven", "Twelve", "Thirteen", "Fourteen", "Fifteen",
    "Sixteen", "Seventeen", "Eighteen", "Nineteen"};
private final String[] tens = {"", "Ten", "Twenty", "Thirty", "Forty", "Fifty", "Sixty",
    "Seventy", "Eighty", "Ninety"};
public static void main(String[] args) {
  Scanner scanner = new Scanner(System.in);
  System.out.print("Enter a number: ");
  int input = scanner.nextInt();
  scanner.close();
  NumberToWords converter = new NumberToWords();
  String words = converter.convertToWords(input);
  System.out.println("Input: " + input);
  System.out.println("Output: " + words);
}
private String convertToWords(int number) {
  if (number == 0) {
    return "Zero";
  }
  return convertToWordsHelper(number);
}
private String convertToWordsHelper(int number) {
  if (number < 10) {
    return units[number];
  } else if (number < 20) {
    return teens[number - 10];
  } else if (number < 100) {
    return tens[number / 10] + " " + convertToWordsHelper(number % 10);
  } else if (number < 1000) {
    return units[number / 100] + " Hundred " + convertToWordsHelper(number % 100);
  } else if (number < 10000) {
    return convertToWordsHelper(number / 1000) + " Thousand " +
        convertToWordsHelper(number % 1000);
  } else {
    return convertToWordsHelper(number / 10000) + " Ten Thousand " +
                convertToWordsHelper(number % 10000);
            }
        }
    }
```

```
import java.util.Scanner;
public class StringComparator {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the first string: ");
        String str1 = scanner.nextLine();
        System.out.print("Enter the second string: ");
        String str2 = scanner.nextLine();
        scanner.close();
        compareStrings(str1, str2);
    }
    private static void compareStrings(String str1, String str2) {
        if (str1.length() != str2.length()) {
            System.out.println("Input strings must be of equal length.");
            return;
        }
        System.out.println("Output:");
        for (int i = 0; i < str1.length(); i++) {
            if (str1.charAt(i) != str2.charAt(i)) {
                System.out.println(String.format("%-5s%s", str1.charAt(i), str2.charAt(i)));
            }
        }    }
}
```

```
import java.util.Scanner;
public class TextJustification {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Text: ");
        String text = scanner.nextLine();
        System.out.print("Padding: ");
        int desiredLength = scanner.nextInt();
        scanner.close();
        String justifiedText = justifyText(text, desiredLength);
        System.out.println("Input: " + text);
        System.out.println("Output: " + justifiedText);
    }
    private static String justifyText(String text, int desiredLength) {
        String[] words = text.split("_");
        int numberOfSpaces = words.length - 1;
        int totalSpacesToAdd = desiredLength - text.length();
        if (numberOfSpaces == 0) {
            return text;
        }
        int spacesToAddPerWord = totalSpacesToAdd / numberOfSpaces;
        int extraSpaces = totalSpacesToAdd % numberOfSpaces;
        StringBuilder justifiedText = new StringBuilder(words[0]);
        for (int i = 1; i < words.length; i++) {
            for (int j = 0; j < spacesToAddPerWord; j++) {
                justifiedText.append(' ');
            }
            if (extraSpaces > 0) {
                justifiedText.append(' ');
                extraSpaces--;
            }
            justifiedText.append(words[i]);
        }
        return justifiedText.toString();
    }
}
```

```
import java.util.Scanner;
public class PalindromeChecker {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String input = scanner.nextLine();
        scanner.close();
        System.out.println("Input: " + input);
        System.out.println("Output: " + isPalindrome(input));
    }
    private static boolean isPalindrome(String str) {
        String cleanedStr = str.replaceAll("[^a-zA-Z0-9]", "").toLowerCase();
        int left = 0;
        int right = cleanedStr.length() - 1;
        while (left < right) {
            if (cleanedStr.charAt(left) != cleanedStr.charAt(right)) {
                return false;
            }
            left++;
            right--;
        }
        return true;
    }
}
```

```
import java.util.HashSet;
import java.util.Scanner;
import java.util.Set;

public class StringPermutations {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String input = scanner.nextLine();
        scanner.close();
        System.out.println("Input: " + input);
        Set<String> permutations = generatePermutations(input);
        System.out.println("Output: " + permutations);
    }
    private static Set<String> generatePermutations(String str) {
        Set<String> result = new HashSet<>();
        generatePermutationsHelper("", str, result);
        return result;
    }
    private static void generatePermutationsHelper(String prefix, String remaining, Set<String> result) {
        int n = remaining.length();
        if (n == 0) {
            result.add(prefix);
        } else {
            for (int i = 0; i < n; i++) {
                String newPrefix = prefix + remaining.charAt(i);
                String newRemaining = remaining.substring(0, i) + remaining.substring(i + 1);
                generatePermutationsHelper(newPrefix, newRemaining, result);
            }
        }
    }
}
```

```
import java.util.Scanner;
public class StringMismatch {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the first string: ");
        String str1 = scanner.nextLine();
        System.out.print("Enter the second string: ");
        String str2 = scanner.nextLine();
        scanner.close();
        System.out.println("Input: " + str1 + ", " + str2);
        findMismatchedSubstrings(str1, str2);
    }
    private static void findMismatchedSubstrings(String str1, String str2) {
```

```java
        int minLength = Math.min(str1.length(), str2.length());
        for (int i = 0; i < minLength; i++) {
            if (str1.charAt(i) != str2.charAt(i)) {
                int j = i + 1;
                while (j < minLength && str1.charAt(j) != str2.charAt(j)) {
                    j++;
                }
                System.out.println(str1.substring(i, j) + "," + str2.substring(i, j));
                i = j - 1;
            }
        }
    }
}
```

```java
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;
public class VowelCount {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String input = scanner.nextLine();
        scanner.close();
        System.out.println("Input: " + input);
        Map<Character, Integer> vowelCount = countVowels(input);
        System.out.println("Output:");
        for (char vowel : "aeiouAEIOU".toCharArray()) {
            System.out.println(vowel + ": " + vowelCount.getOrDefault(vowel, 0));
        }
    }
    private static Map<Character, Integer> countVowels(String str) {
        Map<Character, Integer> vowelCount = new HashMap<>();
        for (char ch : str.toCharArray()) {
            if ("aeiouAEIOU".indexOf(ch) != -1) {
```

```
            vowelCount.put(ch, vowelCount.getOrDefault(ch, 0) + 1);
          }
        }
        return vowelCount;
    }
}
```

```
import java.util.Scanner;
public class NextPalindrome {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int input = scanner.nextInt();
        scanner.close();
        System.out.println("Input: " + input);
        System.out.println("Output: " + findNextPalindrome(input));
    }
    private static int findNextPalindrome(int number) {
        char[] digits = Integer.toString(number).toCharArray();
        int n = digits.length;
        if (allDigitsAreNine(digits)) {
            return (int) Math.pow(10, n) + 1;
        }
        int mid = n / 2;
        boolean leftSmaller = false;
        int i = mid - 1;
        int j = (n % 2 == 0) ? mid : mid + 1;
        while (i >= 0 && digits[i] == digits[j]) {
            i--;
            j++;
        }
        if (i < 0 || digits[i] < digits[j]) {
            leftSmaller = true;
        }
        while (i >= 0) {
            digits[j] = digits[i];
            i--;
            j++;
        }
        if (leftSmaller) {
            int carry = 1;
            mid = (n % 2 == 0) ? mid - 1 : mid;
            while (mid >= 0 && carry > 0) {
                int num = digits[mid] - '0' + carry;
                digits[mid] = (char) ('0' + num % 10);
                carry = num / 10;
                mid--;
            }
        }
        return Integer.parseInt(new String(digits));
    }
    private static boolean allDigitsAreNine(char[] digits) {
        for (char digit : digits) {
            if (digit != '9') {
                return false;
            }
        }
        return true;
    }
}
```