```
1.import java.util.Scanner;
public class Main{
   public static void main(String args[])
   {
   Scanner in=new Scanner(System.in);
   System.out.println("enter the string: ");
   String s=in.nextLine();
   int num=0;
   char temp=' ';
   StringBuilder res=new StringBuilder();
   for(int i=0;i<s.length();i++)
   {
      if(Character.isLetter(s.charAt(i)))
      {
         temp=s.charAt(i);
      }
      else
      {
         num=num*10+Character.getNumericValue(s.charAt(i));
         if(i==s.length()-1 || Character.isLetter(s.charAt(i+1)))
         {
            for(int j=0;j<num;j++)
            {
               res.append(temp);
            }
              num=0;
         }
      }
   }

   for(int i=0;i<res.length();i++)
   {
      System.out.print(res.charAt(i));
   }
   }
}

2.import java.util.Scanner;
public class Main
{
        public static void main(String[] args) {
                Scanner in=new Scanner(System.in);
                System.out.println("enter the string: ");
                StringBuilder res=new StringBuilder();
                String s=in.nextLine();
                int count=1;
                for(int i=1;i<s.length();i++)
                {
```

```
                    if(s.charAt(i-1)==s.charAt(i))
                    {
                        count++;
                    }
                    else{
                        res.append(s.charAt(i-1)).append(count);
                        count=1;
                    }
                }
                 res.append(s.charAt(s.length()-1)).append(count);
                for(int i=0;i<res.length();i++)
                {
            System.out.print(res.charAt(i));
                }

        }
}
3.import java.util.HashMap;
import java.util.Scanner;
public class NumberToWords {

    private static final HashMap<Character, Integer> digitMap = new HashMap<>();
    static {
        digitMap.put('0', 0);
        digitMap.put('1', 1);
        digitMap.put('2', 2);
        digitMap.put('3', 3);
        digitMap.put('4', 4);
        digitMap.put('5', 5);
        digitMap.put('6', 6);
        digitMap.put('7', 7);
        digitMap.put('8', 8);
        digitMap.put('9', 9);
    }

    private static final String[] ones = {"", "one", "two", "three", "four", "five", "six", "seven",
"eight", "nine"};
    private static final String[] teens = {"ten", "eleven", "twelve", "thirteen", "fourteen", "fifteen",
"sixteen", "seventeen", "eighteen", "nineteen"};
    private static final String[] tens = {"", "", "twenty", "thirty", "forty", "fifty", "sixty", "seventy",
"eighty", "ninety"};
    private static final String[] thousands = {"", "thousand", "million", "billion", "trillion"};

    public static String numberToWords(int num) {
        if (num == 0)
            return "zero";

        String numStr = String.valueOf(num);
```

```
        int groups = (numStr.length() + 2) / 3;
        numStr = String.format("%" + (groups * 3) + "s", numStr).replace(' ', '0');

        StringBuilder result = new StringBuilder();
        for (int i = 0; i < groups * 3; i += 3) {
            int h = digitMap.get(numStr.charAt(i));
            int t = digitMap.get(numStr.charAt(i + 1));
            int o = digitMap.get(numStr.charAt(i + 2));

            String group_name = thousands[groups - (i / 3) - 1];

            if (h >= 1) {
                result.append(ones[h]).append(" hundred ");
            }
            if (t > 1) {
                result.append(tens[t]).append(" ");
                if (o >= 1) {
                    result.append(ones[o]).append(" ");
                }
            } else if (t == 1) {
                result.append(teens[o]).append(" ");
            } else if (o >= 1) {
                result.append(ones[o]).append(" ");
            }

            if (!group_name.isEmpty()) {
                result.append(group_name).append(" ");
            }
        }

        return result.toString().trim();
    }

    public static void main(String[] args) {
        Scanner in=new Scanner(System.in);
        int number=in.nextInt();
        System.out.println(numberToWords(number));
    }
}
4.import java.util.Scanner;
public class Main
{
        public static void main(String[] args) {
            Scanner in=new Scanner(System.in);
                System.out.println("enter string1: ");
                String str1=in.nextLine();
                System.out.println("enter string2: ");
                String str2=in.nextLine();
```

```
            for(int i=0;i<str1.length();i++)
            {
                if(str1.charAt(i)!=str2.charAt(i))
                {
                    System.out.println(str1.charAt(i)+","+str2.charAt(i));
                }
            }
        }
}
5.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();
 }
}

6.import java.util.Scanner;
import java.util.*;
```

```
public class Main
{
        public static void main(String[] args) {
            Scanner in=new Scanner(System.in);
            System.out.print("enter string: ");
            String s=in.nextLine();
            s=s.toLowerCase();
            String res="";
            for(int i=0;i<s.length();i++)
            {
                if(s.charAt(i)>='a' && s.charAt(i)<='z')
                {
                    res+=s.charAt(i);
                }
                else{
                    continue;
                }
            }


           StringBuilder rev=new StringBuilder(res).reverse();


              if(res.equals(rev.toString()))
              {
                 System.out.print("true");
              }
              else{
                 System.out.print("false");
              }

        }
}


7..import java.util.Scanner;
import java.util.ArrayList;
public class Main{
    public static void main(String args[]){
        Scanner in=new Scanner(System.in);
        System.out.print("enter the string: ");
        String s=in.nextLine();
        char ch[]=s.toCharArray();
        ArrayList<String> res=new ArrayList<>();
        permutation(ch,0,res);
        ArrayList<String> uniqueList = new ArrayList<>();
       ArrayList<String> uniqueRes = removeDuplicates(res);
        for (String str : uniqueRes) {
```

```
            System.out.print(str + " ");
        }
    }
        static void permutation(char ch[],int fi,ArrayList<String> res)
        {
            if(fi==ch.length-1)
            {
                res.add(new String(ch));
            }
            for(int i=fi;i<ch.length;i++)
            {
                swap(ch,i,fi);
                permutation(ch,fi+1,res);
                swap(ch,i,fi);
            }
        }
        static void swap(char ch[],int i,int fi)
        {
            char temp=ch[i];
            ch[i]=ch[fi];
            ch[fi]=temp;
        }
    static ArrayList<String> removeDuplicates(ArrayList<String> list) {
        ArrayList<String> uniqueList = new ArrayList<>();
        for (String str : list) {
            if (!uniqueList.contains(str)) {
                uniqueList.add(str);
            }
        }
        return uniqueList;
    }
}
8.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) {
 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;
 }
 }
 }
}
```

```
9.import java.util.Scanner;
import java.util.*;
public class Main
{
	public static void main(String[] args) {
	   Scanner in=new Scanner(System.in);
	   System.out.print("enter string: ");
	   String s=in.nextLine();
	   s=s.toLowerCase();
	   String vow="aeiou";
	   char ch[]=vow.toCharArray();
	   HashMap<Character,Integer> map=new HashMap<>();
	   for(int i=0;i<ch.length;i++)
	   {
	      map.put(ch[i],0);
	   }
	   for(int i=0;i<s.length();i++)
	   {
	      if(map.containsKey(s.charAt(i)))
	      {
	         map.put(s.charAt(i),map.get(s.charAt(i))+1);
	      }
	   }
	   for(Map.Entry<Character,Integer> entry:map.entrySet())
	   {
	      System.out.println(entry.getKey()+" "+entry.getValue());
	   }
	}
}
```

```
10.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;
}
}
```