Assignment No 10

1. Write a Java program to get the character at the given index within the String

Program –

```
        import java.util.Scanner;
        public class CharIndex
        {
        public static void main(String[] args)
        {
                Scanner sc = new Scanner(System.in);
                String str = "Edubridge";
                System.out.println("Original String " +str);
                System.out.println("Enter the index number which you want to print:");
                int n = sc.nextInt();
                System.out.println("Character at "+n+" Position is "+str.charAt(n));
                }
        }
```

2. Write a Java program to get the character (Unicode code point) at the given index within the String.

Program -

```
        import java.util.Scanner;
        public class CharUnicode
        {
        public static void main(String[] args)
        {
                Scanner sc = new Scanner(System.in);
                String str = "Edubridge Learning";
                System.out.println("Original String " +str);
                System.out.println("Enter index which u want to println");
                int n = sc.nextInt();
                System.out.println("Character Unicode at point at the "+n+" index:
                "+str.codePointAt(n));
                System.out.println("Character character before the "+n+" index:
                "+str.codePointBefore(n));
                System.out.println("Enter ending index :");
                int m=sc.nextInt();
                System.out.println("number of Unicode code points in the specified text
                range of this String"+str.codePointCount(n, m));
        }
}
```

3. Write a Java program to compare two strings lexicographically. Two strings are lexicographically equal if they are the same length and contain the same characters in the same positions.

Program –

```
        public class CompareString
        {
                public static void main(String[] args)
                {
                        Scanner sc = new Scanner(System.in);
                        System.out.println("Enter First String");
                        String str1 = sc.next();
                        System.out.println("Enter Second String");
                        String str2 = sc.next();
                        System.out.println("String 1 is " +str1+ " " +"String 2 is "
                        +str2);
                        //comparing two strings together
                        int compare = str1.compareTo(str2);
                        //result of comparision
                        if(compare<0)
                        {
                        System.out.println(str1+ " is less than" +str2);
                        }
                        else if(compare == 0)
                        {
                        System.out.println(str1+ " is equals to" +str2);
                        }
                        else
                        {
                        System.out.println(str1+ " is greater than" +str2);
                        }
        }
  }
```

4. Write a Java program to counts occurrences of a certain character in a given string.

Program –

```
        import java.util.Scanner;
        public class CountOccurance
        {
           public static void main(String[] args)
           {
                String input;
                Scanner sc = new Scanner(System.in);
                System.out.println("Enter String");
                input = sc.next();
                char search;
                System.out.println("Enter Character which u want to search");
                search = sc.next().charAt(0);
                int count = 0;
                for(int i=0;i<input.length();i++)
                {
                      if(input.charAt(i)==search)
                      count++;
                }
                System.out.println("Character search "+ search +" appears "+count+"
                times");
           }
        }
```

5. Write a Java program to concatenate a given string with itself of a given number of times.

Program –

```
           import java.util.Scanner;
           public class ConcatString
           {
                public static void main(String[] args)
                {
                String str, s1=" ";
                Scanner sc = new Scanner(System.in);
                System.out.println("String to conactenate");
                str = sc.nextLine();
                int n;
                System.out.println("Enter the number of times to concatenate the
                given string");
                n=sc.nextInt();
                for(int i=0;i<n;i++)
                {
                     s1 += str;
                }
                System.out.println(s1);
           }
     }
```

6. Check the given string is palindrome or not.

Program –

```
import java.util.Scanner;
public class PalindromeString
{
      public static void main(String[] args)
      {
            String x, y = "";
             Scanner a = new Scanner(System.in);
             System.out.print("Enter string you want to check:");
             x = a.nextLine();
             int len = x.length();
             for(int k = len - 1; k >= 0; k--)
             {
             y = y + x.charAt(k);
             }
             if(x.equalsIgnoreCase(y))
             {
                   System.out.println("The string is palindrome.");
             }
             else
             {
             System. out.println("The string is not a palindrome.");
             }
      }
}
```

7. Write a Java program to sort in ascending and descending order by length of the given array of strings.

Sample Output:
Original unsorted colors: [Green, White, Black, Pink, Orange, Blue, Champagne,
Indigo, Ivory]
Sorted color (descending order): [Champagne, Orange, Indigo, Green, White, Black,
Ivory, Pink, Blue]
Sorted color (ascending order): [Pink, Blue, Green, White, Black, Ivory, Orange

Program –

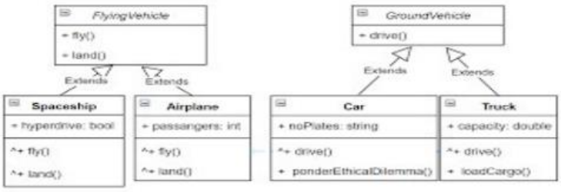
```
import java.util.Arrays;
public class Sorting
{
     public static void main(String[] args)
     {
          String[] Colours = {"Green", "White", "Black", "Pink", "Orange", "Blue", 
"Champagne", "Indigo", "Ivory"};
          System.out.print("Original String -");
          Arrays.sort(Colours,(a,b) -> a.length());
          System.out.println(Arrays.toString(Colours));
          Arrays.sort(Colours, (a, b) -> b.length() - a.length());
          System.out.println(Arrays.toString(Colours));
     }
}
```

8. Java Program to prove that strings are immutable in java.

Program -

```
public class MutableStr
{
      public static void main Check(Object x, Object y)
      {
         if (x == y)
         {
              System.out.println("Both pointing to the same reference");
         }
         else
          {
               System.out.println("Both are pointing to different reference");
          }
         }
         public static void main(String[] args) {
             String st1 = "Java";
             String st2 = "Java";
             System.out.println("Before Modification in st1");
             Check(st1, st2);
             st1 += "ava";
             System.out.println("After Modification");
             Check(st1, st2);
         }
     }
```

9. Java program to implement below classes using inheritance



Program –

FlyingVehical.java

```
    public class FlyingVehical
    {
     public void land()
```

```
 {
       System.out.println("Land method in flyingvehical class");
 }

 public void fly()
{
       // TODO Auto-generated method stub
       System.out.println("Fly method in flyingvehical class");
 }
}
```

GroundVehical.java

```
public class GroundVehical
{
 public void drive()
 {
       System.out.println("Drive method in Ground Vehical");
 }
}


public class Spaceship extends FlyingVehical
{
 boolean  hypendrive;

 public Spaceship(boolean hypendrive)
 {
       super();
       this.hypendrive = hypendrive;
 }

 @Override
 public void fly()
 {
       super.fly();
       System.out.println("fly method of spaceship class");
 }

 public void land()
 {
       super.land();
       System.out.println("land method of spaceship class");
 }
}



public class Airplane extends FlyingVehical
{
 int passengers;
 public Airplane(int passengers)
 {
       super();
       this.passengers = passengers;
 }

 @Override
   public void fly()
  {
  System.out.println("Fly method of airpalne class");
  }

  public void land()
  {

  System.out.println("land method of airplane class");
  }
}



public class Car extends GroundVehical
{
 String noPlates;
  public Car()
  {

  }
  public Car(String noPlates)
  {
        super();
        this.noPlates = noPlates;
        System.out.println("noplate "+ noPlates);
        }

        @Override
        public void drive()
        {
        // TODO Auto-generated method stub
        super.drive();
        System.out.println("drive method of car class");
        }

        public void pounderEthicalDlemma()
        {
        System.out.println("pounderEthicalDlemma method of car class");
  }
}


public class Truck extends GroundVehical
{
 double capacity;
 public Truck()
 {

 }

 public Truck(double capacity)
 {
  super();
  this.capacity = capacity;
  System.out.println("capacity is "+capacity);
 }

 @Override
 public void drive()
 {
  System.out.println("drive method of truck class");
 }

 public void loadCargo()
 {
  System.out.println("loadcargo method of truck class");
 }
}


public class Test
{
 public static void main(String[] args)
 {
      Spaceship space=new Spaceship(true);
      space.fly();
      space.land();

      Airplane air = new Airplane(32);
      int passengers = 100;
      Airplane a1=new Airplane(passengers);
      a1.fly();
      a1.land();

      Car c=new Car("ka passing");
      c.drive();
      c.pounderEthicalDlemma();

      Truck t=new Truck(1234.5);
      t.drive();
      t.loadCargo();
 }
}
```