Program to demonstrate Built-in functions of String Class **Code:**

```java
import java.util.Scanner;
class StringFunctions
{
    public static void main(String args[])
    {
        String str1 = "    Hello";
        String str2 = "World";
        String str3 = "From Java";
        String str4 = "In Java";
        String str5 = str3.concat(" ").concat(str4);
        System.out.println("Returns 0 if str1 == \"Hello\": " +
        ↪  str1.compareTo("Hello")); // compareTo()
        System.out.println("str1 == \"Hello\": " + str1.equals("Hello"));        //
        ↪  equals()
        System.out.println( "str1 == \"Hello\" After trimming: " +
        ↪  str1.trim().equals("Hello")); // trim()
        System.out.println("str2.compareToIgnoreCase(\"world\"): " +
        ↪  str2.compareToIgnoreCase("world"));// compareToIgnoreCase()
        System.out.println("Compare str2.toLowerCase() and \"World\" ignoring the
        ↪  case: " + str2.toLowerCase().equalsIgnoreCase("World"));//
        ↪  toLowerCase() & equalsIgnoreCase()
        System.out.println("str3.toUpperCase(): " + str3.toUpperCase()); //
        ↪  toUpperCase()
        System.out.println("Replace First occurence of \"Java\" with \"Command
        ↪  Prompt\": "+ str5.replaceFirst("Java", "Command Prompt")); //
        ↪  replaceFirst()
        System.out.println("Replace All occurences of \"Java\" with \"Command
        ↪  Prompt\": " + str5.replaceAll("Java", "Command Prompt")); //
        ↪  replaceAll()
        System.out.println("Does str5.replaceAll(\"Java\", \"Command Prompt\")
        ↪  contains \"Java\": "+ str5.replaceAll("Java", "Command
        ↪  Prompt").contains("Java")); // contains()
        System.out.println("str5 ends with \"Java\": " + str5.endsWith("Java")); //
        ↪  endsWith()
        StringBuilder s = new StringBuilder(str3); // For using contentEquals which
        ↪  takes a CharSequence parameter
        System.out.println("Content of s equals content of str3: " +
        ↪  str3.contentEquals(s)); // contentEquals()
        System.out.println("\"\" is empty: " + "".isEmpty()); // isEmpty()
        System.out.println("Replace first occurence of I with O in str4: " +
        ↪  str4.replace('I', 'O')); // replace()
        System.out.println("Length of str5: " + str5.length()); // length()
```

```
26          System.out.println("Character at index 3 in str5: " + str5.charAt(3)); //
             ↪  charAt()
27          System.out.println("Substring of str5 from the index of where \"Java\" is
             ↪  found: " + str5.substring(str5.indexOf("Java"))); // substring()
28          char arr[] = str1.toCharArray(); // toCharArray()
29          for(int i = 0; i < arr.length; i++)
30          {
31              if(arr[i] == ' ')
32              {
33                  arr[i] = '_';
34              }
35          }
36          System.out.print("Replacing spaces with underscore in arr: ");
37          System.out.print(arr);
38      }
39  }
```

Output:

```
Returns 0 if str1 == "Hello": -40
str1 == "Hello": false
str1 == "Hello" After trimming: true
str2.compareToIgnoreCase("world"): 0
Compare str2.toLowerCase() and "World" ignoring the case: true
str3.toUpperCase: FROM JAVA
Replace First occurence of "Java" with "Command Prompt": From Command Prompt In Java
Replace All occurences of "Java" with "Command Prompt": From Command Prompt In Command Prompt
Does str5.replaceAll("Java", "Command Prompt") contains "Java": false
str5 ends with "Java": true
Content of s equals content of str3: true
"" is empty: true
Replace first occurence of I with O in str4: On Java
Length of str5: 17
Character at index 3 in str5: m
Substring of str5 from the index of where "Java" is found: Java In Java
Replacing spaces with underscore in arr: ____Hello
```

Matrix Class:

```
1   // matrix/Matrix.java
2   package matrix;
3   import java.util.Scanner;
4   public class Matrix
5   {
6       int arr[][];
7       int rows, columns;
8       public Matrix(int rows, int columns)
9       {
10          arr = new int[rows][columns];
```

```
        this.rows = rows;
        this.columns = columns;
    }
    public Matrix()
    {
        arr = new int[2][2];
        rows = 2;
        columns = 2;
    }
    public int elementAt(int row, int column)
    {
        return arr[row][column];
    }
    public void setElement(int row, int column, int data)
    {
        arr[row][column] = data;
    }
    public void setMatrix()
    {
        Scanner sc = new Scanner(System.in);
        for(int i = 0; i < rows; i++)
        {
            for(int j = 0; j < columns; j++)
            {
                System.out.print("mat[" + i +"]" + "[" + j + "]: ");
                this.setElement(i, j, sc.nextInt());
            }
        }
    }
    public String toString()
    {
        StringBuilder str = new StringBuilder();
        for(int i=0; i < rows; i++)
        {
            for(int j = 0; j < columns; j++)
            {
                str.append(this.elementAt(i, j));
                str.append(' ');
            }
            str.append('\n');
        }
        return str.toString();
    }

    public Matrix transpose()
    {
        Matrix matTranspose = new Matrix(rows, columns);
```

```
        for(int i = 0; i < rows; i++)
        {
            for(int j = 0; j < columns; j++)
            {
                matTranspose.setElement(i, j, this.elementAt(j, i));
            }
        }
        return matTranspose;
    }

    public boolean equals(Matrix mat)
    {
        if( rows != mat.columns || columns != mat.columns)
        {
            System.out.print("Cannot Compare these matrices");
        }
        for(int i = 0; i < rows; i++)
        {
            for(int j = 0; j < columns; j++)
            {
                if(this.elementAt(i, j) != mat.elementAt(i, j))
                {
                    return false;
                }
            }
        }
        return true;
    }

    public int getColumns()
    {
        return columns;
    }

    public int getRows()
    {
        return rows;
    }
}
```

To check if the entered matrix is symmetric or not **Code:**

```
// Symmetric.java
import java.util.Scanner;
import matrix.Matrix;
class Symmetric
{
    static boolean isSymmetric(Matrix mat)
```

```
    {
        return mat.equals(mat.transpose());
    }
    public static void main(String args[])
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the order of the Matrix: ");
        int order = sc.nextInt();
        Matrix mat2 = new Matrix(order, order);
        for(int i = 0; i < order; i++)
        {
            for(int j = 0; j < order; j++)
            {
                System.out.print("mat[" + i +"]" + "[" + j + "]: ");
                mat2.setElement(i, j, sc.nextInt());
            }
        }
        System.out.println(mat2);
        System.out.println("The Matrix is " + ((isSymmetric(mat2) ? "Symmetric" :
         ↪  "Not Symmetric")));
    }
}
    // SampleClass.java
    import package1.*;
    import package2.ClassA;
    import package2.packageA.*;
    class SampleClass
    {
        public static void main(String args[])
        {
            package1.Class1 c1 = new package1.Class1();
            Class2 c2 = new Class2();
            Class3 c3 = new Class3();
            ClassA cA = new ClassA();
            package2.packageA.Class1 c31 = new package2.packageA.Class1();
            System.out.println("hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi
             ↪  hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi
             ↪  hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi");
            {
                {
                   System.out.println("hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi
                     ↪  hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi hi
                     ↪  hi hi hi hi hi hi hi hi hi hi);
                }
            }
        }
    }
```

Output:

```
Enter the order of the Matrix: 3
mat[0][0]: 1
mat[0][1]: 0
mat[0][2]: 0
mat[1][0]: 0
mat[1][1]: 1
mat[1][2]: 0
mat[2][0]: 0
mat[2][1]: 0
mat[2][2]: 1
1 0 0
0 1 0
0 0 1

The Matrix is Symmetric
```

To Perform Matrix Multiplication **Code:**

```java
// Multiplication.java
import java.util.Scanner;
import matrix.Matrix;
class Multiplication
{
    static Matrix multiply(Matrix mat1, Matrix mat2)
    {
        if(mat1.getColumns() != mat2.getRows())
        {
            System.out.println("Cannot Multiply these matrices");
        }
        Matrix matMult = new Matrix(mat1.getRows(), mat2.getColumns());
        for(int i=0; i < mat1.getRows(); i++)
        {
            for(int j=0; j < mat1.getColumns(); j++)
            {
                for(int k = 0; k < mat2.getRows(); k++)
                {
                    matMult.setElement(i, j, matMult.elementAt(i, j) +
 ↪  mat1.elementAt(i, k) * mat2.elementAt(k, j));
                }
            }
        }
        return matMult;
    }
    public static void main(String args[])
    {
        Scanner sc = new Scanner(System.in);
```

```
        System.out.print("Enter the no of rows of Matrix1: ");
        int rows = sc.nextInt();
        System.out.print("Enter the no of columns of Matrix1: ");
        int columns = sc.nextInt();
        Matrix mat1 = new Matrix(rows, columns);
        mat1.setMatrix();
        System.out.print("Enter the no of rows of Matrix2: ");
        rows = sc.nextInt();
        System.out.print("Enter the no of columns of Matrix2: ");
        columns = sc.nextInt();
        Matrix mat2 = new Matrix(rows, columns);
        mat2.setMatrix();
        System.out.println("mat1 x mat2 = ");
        System.out.print(multiply(mat1, mat2));
    }
}
```

Output:

```
Enter the no of rows of Matrix1: 2
Enter the no of columns of Matrix1: 2
mat[0][0]: 1
mat[0][1]: 1
mat[1][0]: 1
mat[1][1]: 1
Enter the no of rows of Matrix2: 2
Enter the no of columns of Matrix2: 2
mat[0][0]: 1
mat[0][1]: 2
mat[1][0]: 1
mat[1][1]: 2
mat1 x mat2 =
2 4
2 4
```

Reverse the string and decide whether it is palindrome or not and Capitalize the String **Code:**

```
import java.util.Scanner;
class Pallindrome
{
    public static void main(String args[])
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a String: ");
        String in = sc.next();
        char str[] = in.toCharArray();
        char rev[] = new char[str.length];
        for(int i = 0; i < str.length; i++)
        {
```

```
13              rev[i] = str[str.length - 1 - i];
14          }
15          System.out.println("The String is " + (in.equals(new String(rev)) ?
            ↪  "Pallindrome" : "Not Pallindrome"));
16          System.out.println("Capitalized String: " + in.toUpperCase());
17      }
18  }
```

Output:

```
Enter a String: naman
The String is Pallindrome
Capitalized String: NAMAN
```

Code:

```java
class TestStringBuffer
{
  public static void main(String args[])
  {
    StringBuffer stb = new StringBuffer("Hi There");
    StringBuffer stb2 = new StringBuffer("Hi Java Hi There");
    System.out.println("Capacity Before trimming: " + stb.capacity());
    System.out.println("Length Before trimming: " + stb.length());
    stb.trimToSize();
    System.out.println("Capacity After trimming, before ensureCapaity():
     ↪  " + stb.capacity());
    System.out.println("Length After trimming, before ensureCapacity(): "
     ↪  + stb.length());
    stb.ensureCapacity(30);
    System.out.println("Capacity after ensureCapaity(), before
     ↪  setLength(): " + stb.capacity());
    System.out.println("Length after ensureCapacity(), before
     ↪  setLength(): " + stb.length());
    int length = stb.length();
    stb.setLength(32);
    System.out.println("Capacity after setLength(): " + stb.capacity());
    System.out.println("Length after setLength(): " + stb.length());
    stb.setLength(length);
    System.out.println("Deleting First Hi from stb: " +
     ↪  stb.delete(stb.indexOf("Hi"), stb.indexOf("Hi") +
     ↪  "Hi".length()));
    int indexOfThere = stb.indexOf("There");
    System.out.println("Before: " + stb + "\nDeleting char \'T\' from
     ↪  stb: " + stb.deleteCharAt(indexOfThere));
    stb.setCharAt(indexOfThere, 'T');
    System.out.println("Adding \'T\' back to stb: " + stb);
    System.out.println("Replacing There with Java in stb: " +
     ↪  stb.replace(indexOfThere, indexOfThere + "There".length(),
     ↪  "Java"));
    System.out.println("Inserting Hi at the start of stb: " +
     ↪  stb.insert(0, "Hi"));
    System.out.println("Appending Hi There to stb" + stb.append("Hi
     ↪  There"));
    System.out.println("Comparing stb with stb2: " + stb.compareTo(stb2));
    System.out.println("Substring of the stb with portion after last Hi
     ↪  removed: " + stb.substring(stb.lastIndexOf("Hi", stb.length() -
     ↪  1)));
    System.out.println("Reverese of stb2: " + stb2.reverse());
    System.out.println("Finally:\nstb: "+stb +"\nstb2: " + stb2);

```

```
  }
}
```

Output:

```
Capacity Before trimming: 24
Length Before trimming: 8
Capacity After trimming, before ensureCapaity(): 8
Length After trimming, before ensureCapacity(): 8
Capacity after ensureCapaity(), before setLength(): 30
Length after ensureCapacity(), before setLength(): 8
Capacity after setLength(): 62
Length after setLength(): 32
Deleting First Hi from stb:  There
Before:  here
Deleting char 'T' from stb:  here
Adding 'T' back to stb:  Tere
Replacing There with Java in stb:  Java
Inserting Hi at the start of stb: Hi Java
Appending Hi There to stbHi JavaHi There
Comparing stb with stb2: 40
Substring of the stb with portion after last Hi removed: Hi There
Reverese of stb2: erehT iH avaJ iH
Finally:
stb: Hi JavaHi There
stb2: erehT iH avaJ iH
```

Code:

```java
import java.util.*;
class TestVector
{
    public static void main(String args[])
    {
        Vector<Integer> vec = new Vector<Integer>();
        System.out.println("\tCapacity of vec: " + vec.capacity());
        System.out.println("\tSize of vec: " + vec.size());
        vec.trimToSize();
        System.out.println("After trimToSize(): ");
        System.out.println("\tCapacity of vec: " + vec.capacity());
        System.out.println("\tSize of vec: " + vec.size());
        for(int i = 1; i <= 10; i++)
        {
            vec.add(i);
        }
        System.out.println("After Adding Elements: ");
        System.out.println("\tCapacity of vec: " + vec.capacity());
        System.out.println("\tSize of vec: " + vec.size());
        System.out.println(vec);
```

```
21          vec.ensureCapacity(30);
22          System.out.println("After ensureCapacity(30): ");
23          System.out.println("\tCapacity of vec: " + vec.capacity());
24          System.out.println("\tSize of vec: " + vec.size());
25          vec.setSize(15);
26          System.out.println("After setSize(15): ");
27          System.out.println("\tCapacity of vec: " + vec.capacity());
28          System.out.println("\tSize of vec: " + vec.size());
29          for(int i = 0; i <= 9; i++)
30          {
31              vec.set(i, 10 + (i % 2));
32          }
33          System.out.println("Setting all elemnts in the Vector to a
             ↪  different value: ");
34          for(int i = 0; i <= 14; i++)
35          {
36              System.out.print(vec.elementAt(i) + " ");
37          }
38          System.out.printf("\n");
39          System.out.println("First Element of the Vector: " +
             ↪  vec.firstElement());
40          System.out.println("Index of first occurence of 11: "+
             ↪  vec.indexOf(11));
41          System.out.println("Index of first occurence of 11 after index 4:
             ↪  "+vec.indexOf(11, 4));
42          System.out.println("Last Element of the Vector: " +
             ↪  vec.lastElement());
43          System.out.println("Index of last occurence of 11: "+
             ↪  vec.lastIndexOf(11));
44          System.out.println("Index of last occurence of 11 before index 8:
             ↪  "+ vec.lastIndexOf(11, 8));
45          System.out.println("Initially: " + vec);
46          vec.removeElement(null);
47          System.out.println("Vector after removing a null element\n" +
             ↪  vec);
48          Vector<Integer> vec1 = new Vector<Integer>();
49          vec1.add(null);
50          vec.removeAll(vec1);
51          System.out.println("Vector after removing all null elements\n" +
             ↪  vec);
52          vec.insertElementAt(11, 5);
53          System.out.println("Vector after inserting 11 at index 5
             ↪  elements\n" + vec);
54          vec.clear();
55          System.out.println("Vector after vec.clear()\n" + vec);
56          System.out.println("The Vector is Empty: " + vec.isEmpty());
57         System.out.println("The Vector contains 11: " + vec.contains(11));
```

```
58      }
59  }
```

Output:

```
  Capacity of vec: 10
  Size of vec: 0
After trimToSize():
  Capacity of vec: 0
  Size of vec: 0
After Adding Elements:
  Capacity of vec: 16
  Size of vec: 10
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
After ensureCapacity(30):
  Capacity of vec: 32
  Size of vec: 10
After setSize(15):
  Capacity of vec: 32
  Size of vec: 15
Setting all elemnts in the Vector to a different value:
10 11 10 11 10 11 10 11 10 11 null null null null null
First Element of the Vector: 10
Index of first occurence of 11: 1
Index of first occurence of 11 after index 4: 5
Last Element of the Vector: null
Index of last occurence of 11: 9
Index of last occurence of 11 before index 8: 7
Initially: [10, 11, 10, 11, 10, 11, 10, 11, 10, 11, null, null, null, null, null]
Vector after removing a null element
[10, 11, 10, 11, 10, 11, 10, 11, 10, 11, null, null, null, null]
Vector after removing all null elements
[10, 11, 10, 11, 10, 11, 10, 11, 10, 11]
Vector after inserting 11 at index 5 elements
[10, 11, 10, 11, 10, 11, 11, 10, 11, 10, 11]
Vector after vec.clear()
[]
The Vector is Empty: true
The Vector contains 11: false
```

Name: Aum Kulkarni
Divison: D6AD
Roll No 36
Experiment 11: Program on abstract class and abstract methods

Code:

```java
import java.util.Scanner;
import static java.lang.Math.*;
abstract class Shape
{
  public abstract double area();
  public String toString()
  {
    Double result = Double.valueOf(area());
    String res = result.toString();
    return res;
  }
}
class Rectangle extends Shape
{
  double length, breadth;
  public Rectangle(double length, double breadth)
  {
    this.length = length;
    this.breadth = breadth;
  }
  public double area()
  {
    return length * breadth;
  }
  public String toString()
  {
    String str = "The area of a Rectangle with length " + length + " and
     ↪  width " + breadth + ": " + area();
    return str;
  }
}
class Triangle extends Shape
{
  double side1, side2, side3;
  public Triangle(double side1, double side2, double side3)
  {
    this.side1 = side1;
    this.side2 = side2;
    this.side3 = side3;
  }
  public double area()
  {
    double s = (side1 + side2 + side3) / 2;
    return sqrt(s * (s - side1) * (s - side2) * (s - side3));
  }
```

```
45    public String toString()
46    {
47      String str = "The area of a Triangle with sides " + side1 + " " +
          ↪  side2 + " " + side3 + ": " + area();
48      return str;
49    }
50  }
51  class Circle extends Shape
52  {
53    double radius;
54    public Circle(double radius)
55    {
56      this.radius = radius;
57    }
58    public double area()
59    {
60      return PI * radius * radius;
61    }
62    public String toString()
63    {
64      String str = "The area of a Circle with radius " + radius + ": " +
          ↪  area();
65      return str;
66    }
67  }
68  class AbstracMethods
69  {
70    public static void main(String args[])
71    {
72      Shape[] shapes = new Shape[3];
73      shapes[0] = new Triangle(10, 15, 20);
74      shapes[1] = new Rectangle(10, 20);
75      shapes[2] = new Circle(10);
76      for(int i = 0; i < 3; i++)
77      {
78        System.out.println(shapes[i]);
79      }
80    }
81  }
```

Output:

```
The area of a Triangle with sides 10.0 15.0 20.0: 72.61843774138907
The area of a Rectangle with length 10.0 and width 20.0: 200.0
The area of a Circle with radius 10.0: 314.1592653589793
```

If we make Rectangle class final and make a class Square that inherits from it the Java compiler will show an error

```java
final class Rectangle extends Shape
{
  double length, breadth;
  public Rectangle()
  {
    length  = 0.0;
    breadth = 0.0;
  }
  public Rectangle(double length, double breadth)
  {
    this.length  = length;
    this.breadth = breadth;
  }
  public double area()
  {
    return length * breadth;
  }
  public String toString()
  {
    String str = "The area of a Rectangle with length " + length + " and
     ↪  width " + breadth + ": " + area();
    return str;
  }
}
class Square extends Rectangle
{
}
```

```
AbstractMethods.java:36: error: cannot inherit from final Rectangle
class Square extends Rectangle
                     ^
1 error
```