public int compareTo(String str) :
public int compareToIgnoreCase(String str) :

- It is used to compare two String **character by character** by using UNICODE value.
- Comparing two Strings Character by character using UNICODE value is known as **Lexicographical Comparison.**
- The return type of this method is int, actually it returns the difference of UNICODE value.
- While comparing the character we have follwing cases :

  a) If both the characters are same then IT will **zero**
  b) If first character UNICODE value is greater than second character then It will return **+ve**
  c If first character UNICODE value is less than second character then It will return **-ve**

```
package com.ravi.String_handling;

public class LexicographicalComparison
{
   public static void main(String[] args)
   {
      String s1 ="Sachin";    //PQRS
      String s2 = "Sachin";
      String s3 = "Ratan";

      System.out.println(s1.compareTo(s2)); //0
      System.out.println(s1.compareTo(s3)); //1   S > R
      System.out.println(s3.compareTo(s1)); //-1   R < S

      System.out.println(".................");

      String s4 = "Apple";
      String s5 = "apple";
      System.out.println(s4.compareTo(s5));  // A   a  [65  97]
      System.out.println(s5.compareTo(s4));  // a   A  [97  65]

      System.out.println("...............");
      String s6 = "Ravi";
      String s7 = "Rajeev";
      System.out.println(s7.compareTo(s6)); // -12

      System.out.println("...............");
      String s8 = "Apple";
      String s9 = "apple";
      System.out.println(s8.compareToIgnoreCase(s9)); //0
   }
}
```

public String substring(int startIndex)
public String substring(int startIndex, int endIndex)

0	1	2	3	4	5	6	7	8	Index Position
H	Y	D	E	R	A	B	A	D	

- It is used to retrieve the part of the String from the original string.
- The first index is inclusive and 2nd index is **exclusive**.
- If first index and last index both are same then It will not print anything
- If first index is greater than second index then it will generate an exception java.lang.StringIndexOutOfBoundsException
- If we pass any index as a negative then it will generate an exception java.lang.StringIndexOutOfBoundsException

```
package com.ravi.String_handling;

public class SubstringDemo {

    public static void main(String[] args)
    {
        String str = "Hyderabad";
        System.out.println(str.substring(2));//derabad
        System.out.println(str.substring(3,7));//erab
        System.out.println(str.substring(4,4)); //Will not print anything
        System.out.println(str.substring(7,4)); //StringIndexOutOfBoundsException
        System.out.println(str.substring(-2,5));
    }
}
```

public String trim() :

- It is used to remove the white spaces from the begning and end of the String.
- It will not remove white space from the middle of the String.

```
package com.ravi.String_handling;

public class TrimDemo
{
    public static void main(String[] args)
    {
        String s1="   Tata    ";
        System.out.println(s1+"Nagar");

         s1 = "          Hello       Data        ";
        System.out.println(s1.trim() +"Base");
    }
}
```

public boolean isEmpty() :

Will verify whether the String is empty or not, Here empty means length is 0 or not, If length is 0 then it will return true otherwise false.

```
package com.ravi.String_handling;

public class IsEmptyDemo
{
    public static void main(String[] args)
    {
        String s1 = "";
        System.out.println(s1.isEmpty());

        String s2 = " ";
        System.out.println(s2.isEmpty());

        String s3 = "NIT";
        System.out.println(s3.isEmpty());
    }
}
```

public boolean isBlank() :

- It will return true if String is empty OR contains only white space character.
- It is available from java 11V.

```
package com.ravi.String_handling;

public class IsBlank
{
    public static void main(String[] args)
    {
        String s1 = "";
        System.out.println(s1.isBlank()); //true

        String s2 = " ";
        System.out.println(s2.isBlank()); //true

        String s3 = "\n \t ";
        System.out.println(s3.isBlank()); //true

        String s4 = "Java";
        System.out.println(s4.isBlank());   //false

    }

}
```

public int indexOf(String str)

It will serach the index position of the first occurrance of the specified String as a parameter.

If the specified string is not available in the existing string then it will return -1.

```
package com.ravi.String_handling;

public class IndexOfDemo
{
    public static void main(String[] args)
    {
        String  str = "India is my country and It is in Asia";
         int index = str.indexOf("is");
        System.out.println("First Occurrance of 'is'  is :"+index);
    }
}
```

public int lastIndexOf(String x)

It will serach the index position of the last occurrance of the String.

It takes String as a parameter and return type of this method is int.

```
package com.ravi.String_handling;

public class LastIndexOf
{
    public static void main(String[] args)
    {
        String s1 = "it is a nice city";
         int lastIndex = s1.lastIndexOf("it");
        System.out.println("Last occurrance of it, is :"+lastIndex+ "th position");
    }
}
```

public String [] split(String regex) :

- It is used to break OR Split the given string based on regex parameter. Here regex is nothing but regular expresion.
- Based on the given regex it will break the String so the return type is String array.

```
package com.ravi.String_handling;

public class SplitDemo {

    public static void main(String[] args)
    {
        String s1 = "Java is a high level language";
        String[] words = s1.split(" ");

        for(String word : words)
        {
           System.out.println(word);
        }

        System.out.println(".....................");

        String s2 = "Java is a high level language";
        words =  s2.split("a");

        for(String word : words)
        {
           System.out.println(word);
        }

    }
}
```

public char[] toCharArray() :

- Will convert the String into character array.

```
package com.ravi.String_handling;

import java.util.Arrays;

public class ToCharArrayDemo
{
    public static void main(String[] args)
    {
        String s1 = "Java technology";

        char[] chars = s1.toCharArray();
        System.out.println(Arrays.toString(chars));
    }
}
```

public byte[] getBytes() :

- Used to convert the String into byte array.
- It is useful when we want to write binary data in the file.
- The return type of this method is byte[] so it is in binary format.

```
package com.ravi.String_handling;

import java.util.Arrays;

public class BetBytesDemo {

    public static void main(String[] args)
    {
        String str = "abcdef";

        //String to byte (Binary Data)
        byte[] bytes = str.getBytes();
        System.out.println(Arrays.toString(bytes));

    }

}
```