DSA

Module 7

Strings

Author
Srinivas Dande

7. Strings

- String is a Sequence of Characters.
- String can be represented in java in the following ways
    1) String class
    2) StringBuffer
    3) StringBuilder
    4) Array of Characters
    5) ArrayList of Characters

7.1. String Class

- String is a built-in class in java.lang package.
- String is final class, so you can't define the subclass for String class.
- String class implements the following interfaces:
    - java.io.Serializable
    - java.lang.Comparable
    - java.lang.CharSequence
- String class has following variable to hold data.

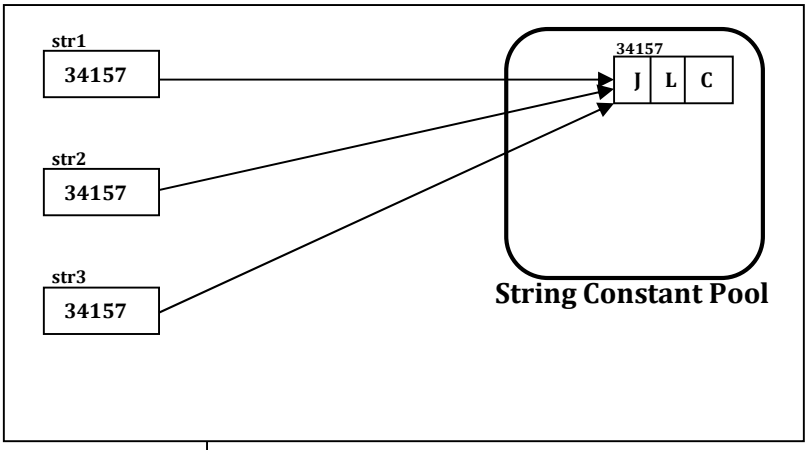
  private final char value[];
- String objects are immutable objects. It means once the object is created then the content or data of the object can't be modified.
- When you try to modify the contents of object then new object will be created as a result.
- You can create the object of String in two ways.
    - Without using new operator.
    - Using new operator.

7.1.1. Creating String object without new operator

1) JVM allocates the memory for the String reference variable.
2) JVM verifies the String literal in the String Constant Pool.
3) If String literal is not available in the String Constant Pool then it creates new String object inside the String Constant pool and newly created String object address will be assigned to String reference variable.
4) If String literal is available then existing String object address will be assigned to String reference variable.

Demo1.java

```
class Lab457{
public static void main(String as[]){
String str1="JLC";
String str2="JLC";
String str3="JLC";
System.out.println(str1==str2);
System.out.println(str1==str3);
System.out.println(str2==str3);
}
}
```

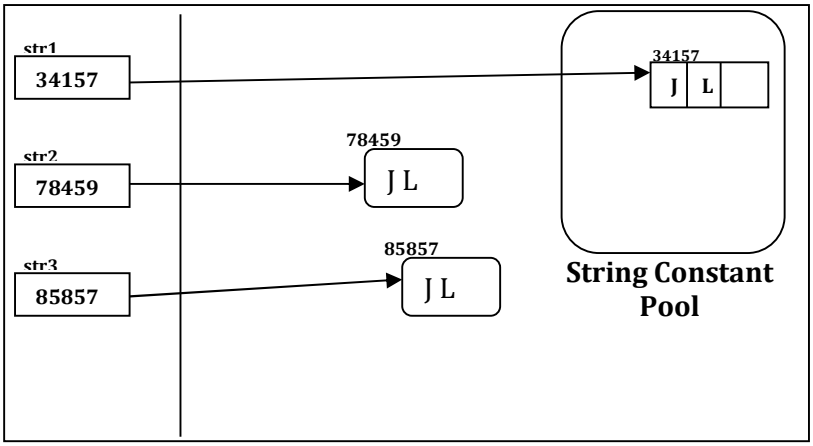


7.1.2 Creating String object with new operator

1) JVM allocates the memory for the String reference variable.
2) JVM verifies the String literal in the String Constant Pool.
3) If String literal is not available in the String Constant Pool then it creates new String object inside the String Constant pool.
4) If String literal is available in the String Constant Pool then ignores that.
5) It creates another new String object outside the constant pool and assigns address of the newly created String object outside the pool to String reference variable.

Demo2.java

```
class Lab458{
public static void main(..... as[]){
String str1="JLC";
String str2=new String("JLC");
String str3=new String("JLC");
System.out.println(str1==str2);
System.out.println(str1==str3);
System.out.println(str2==str3);
}
}
```



7.2. StringBuffer Class

- StringBuffer is a final class in java.lang package.
- It is used to store the sequence of characters.
- This is used when there is a necessity to make a lot of modifications to characters of String.
- StringBuffer is a mutable sequence of characters, It means the contents of the StringBuffer can be modified after creation.
- Every StringBuffer object has a capacity associated with it.
- The capacity of the StringBuffer is the number of characters it can hold.
- The capacity increases automatically as more contents added to it.
- The methods available in the StringBuffer class are synchronized.
- StringBuffer class is thread-safe i.e. multiple threads cannot access it simultaneously.

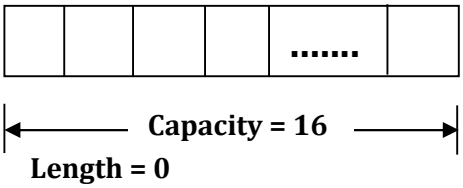
7.3. StringBuilder Class

- StringBuilder is a final class in java.lang package.
- StringBuilder is a newly added class from Java 5.
- StringBuilder class functionality is same as StringBuffer only except the methods available in the StringBuilder class are non synchronized.
- StringBuilder class is not thread-safe i.e. multiple threads can access it simultaneously.

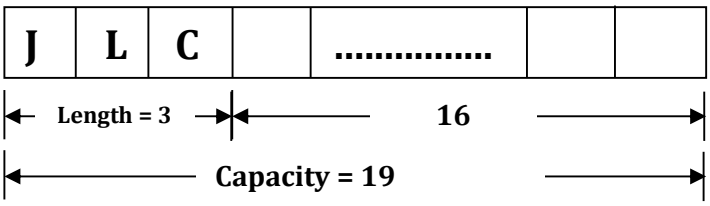
7.4. About StringBuffer and StringBuilder

- equals() is not overriden in StringBuilder and StringBuffer class.
- When you want to compare contents of StringBuilder or StringBuffer object then you need to convert the object of StringBuilder or StringBuffer into String.
- hashCode() is not overriden in the StringBuilder and StringBuffer class. It uses the default implementation of Object class.
- StringBuilder and StringBuffer are not implementing java.lang.Comparable interface, So You can't use compareTo() with StringBuilder and StringBuffer.

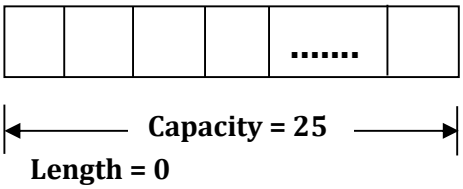
StringBuilder sb=new StringBuilder();



StringBuilder sb=new StringBuilder("JLC");



StringBuilder sb=new StringBuilder(25);



7.5. Examples on Strings

Lab1.java

```
package com.jlcindia.strings;

/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */

public class Lab1 {
	public static void main(String[] args) {

	String str1="JLC";
	String str2="JLC";

	System.out.println(str1==str2);

	String str3=new String("JLC");
	System.out.println(str1==str3);

	}
}
```

Lab2.java

```
package com.jlcindia.strings;

/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */

public class Lab2 {
	public static void main(String[] args) {

	String str1="Hello";
	String str2=" Guys";

	String str3="Hello Guys";
	String str4= str1.concat(str2);
	String str5="Hello".concat(" Guys");

	System.out.println(str3==str4); //F
	System.out.println(str3==str5); //F
	System.out.println(str4==str5); //F

	String str6=str1+str2;
	String str7="Hello" + " Guys"; //IMP- Literal + Literal

	System.out.println(str3==str6);
	System.out.println(str3==str7);

	}
}
```

Lab3.java

```
package com.jlcindia.strings;

/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */

public class Lab3 {
	public static void main(String[] args) {

	String str1="JLCD";
	String str2="JLCD";
	String str3="JLCd";

	System.out.println(str1.equals(str2));

	System.out.println(str3.equals(str2));
	System.out.println(str3.equalsIgnoreCase(str2));

	System.out.println(str1.compareTo(str2)); //Same
	System.out.println(str2.compareTo(str3)); //str2 is smaller
	System.out.println(str3.compareTo(str2)); //str3 is Bigger



	}

}
```

Lab4.java

```
package com.jlcindia.strings;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */
public class Lab4 {
	public static void main(String[] args) {

	String str="srinivasd";

	System.out.println(str);
	System.out.println(str.length());

	char chArr[] = str.toCharArray();

	for(int i=0;i<chArr.length;i++)
			System.out.print(chArr[i]+" ");

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

	chArr[0]='S';
	chArr[8]='D';

	for(int i=0;i<chArr.length;i++)
		System.out.print(chArr[i]+" ");

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

	String str1=new String(chArr);
	System.out.println(str1);
	System.out.println(str);
	}
}
```

Lab5.java

```
package com.jlcindia.strings;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */

public class Lab5 {
	public static void main(String[] args) {

	String str="srinivasd";

	System.out.println(str.length());
	System.out.println(str.charAt(0));
	System.out.println(str.charAt(8));

	System.out.println(str.substring(5));
	System.out.println(str.substring(3,8));
	System.out.println(str.substring(3,9));

	System.out.println(str.indexOf("nivas"));
	System.out.println(str.indexOf("i"));
	System.out.println(str.lastIndexOf("i"));

	System.out.println(str.contains("vas"));
	System.out.println(str.contains("nivas"));
	System.out.println(str.contains("hello"));

	}
}
```

Lab6.java

```
package com.jlcindia.strings;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */

public class Lab6 {
	public static void main(String[] args) {

	StringBuffer sb1=new StringBuffer("JLC");
	StringBuffer sb2=new StringBuffer("JLC");

	System.out.println(sb1==sb2);

	System.out.println(sb1.equals(sb2));

	boolean b1= sb1.toString().contentEquals(sb2.toString());
	boolean b2= sb1.toString().equals(sb2.toString());

	System.out.println("b1 : "+b1);
	System.out.println("b2 : "+b2);

	sb1.append(" Guys");
	System.out.println(sb1);

	}
}
```

Lab7.java

```
package com.jlcindia.strings;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */

public class Lab7 {
	public static void main(String[] args) {

	StringBuilder sb1=new StringBuilder("JLC");
	StringBuilder sb2=new StringBuilder("JLC");

	System.out.println(sb1==sb2);

	System.out.println(sb1.equals(sb2));

	boolean b1= sb1.toString().contentEquals(sb2.toString());
	boolean b2= sb1.toString().equals(sb2.toString());

	System.out.println("b1 : "+b1);
	System.out.println("b2 : "+b2);

	sb1.append(" Guys");
	System.out.println(sb1);

	sb1.reverse();
	System.out.println(sb1);
	}

}
```

7.6. Problems on Strings

Problem1A.java

```
package com.jlcindia.strings;

/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */

//Problem1- Reverse the given String

public class Problem1A {
	public static void main(String[] args) {

	String str = "Hello";
	System.out.println(str);

	StringBuffer sb=new StringBuffer(str);

	sb.reverse();

	String revStr = sb.toString();
	System.out.println(revStr);

	}
}

// Time Complexity - - O(1)
// Aux Space - O(1)
```

Problem1B.java

```
package com.jlcindia.strings;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */
//Problem1- Reverse the given String
public class Problem1B {
	static String reverseString(String str) {

		char chArr[] = str.toCharArray();
		int n = chArr.length;
		int start=0;
		int end=n-1;
		while(start<end) {
			char temp = chArr[start];
			 chArr[start] =  chArr[end];
			 chArr[end]=temp;

			 start++;
			 end--;
		}
		String revStr=new String(chArr);
		return revStr;
	}
	public static void main(String[] args) {
		String str = "Hello";
		System.out.println(str);

		String revStr = reverseString(str);
		System.out.println(revStr);
	}
}

// Time Complexity - -  O(n/2)/O(n)
// Aux Space - O(n)
```

Problem2.java

```
package com.jlcindia.strings;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */
//Problem2- Reverse Words of given Sentence

public class Problem2 {

	static void reverse(char chArr[], int start, int end) {
		while (start < end) {
			char temp = chArr[start];
			chArr[start] = chArr[end];
			chArr[end] = temp;

			start++;
			end--;
		}
	}

	static void reverseWords(char chArr[]) {
		int n = chArr.length;
		int start = 0;
		int end = 0;
		while (end < n) {
			if (chArr[end] == ' ') {
				reverse(chArr, start, end - 1);
				start = end + 1;
			}
			end++;
		}
		reverse(chArr, start, end-1); // Reverse Last Word
		reverse(chArr, 0, n-1); //Reverse the Entire String
	}
```

```
        public static void main(String[] args) {

                String str = "SD  OK";
                System.out.println(str);

                char chArr[] = str.toCharArray();
                reverseWords(chArr);

                String revStr= new String(chArr);
                System.out.println(revStr);

        }
}

//Time Complexity - O(3n) /O(n)
//Aux Space --- O(n)
```

Problem3.java

```
package com.jlcindia.strings;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */
//Problem3- Frequency of each Character of given String with lower
// Alphabets in Sorted Order
public class Problem3 {

		static void charFrequency(String str) {
			int n= str.length();

			int count[] = new int[26];

			for(int i=0;i<n;i++) {
				char ch=str.charAt(i);
				count[ch-97]++;
			}

			for(int i=0;i<count.length;i++) {
				if(count[i]>0) {
				char ch = (char) (97+i);
				System.out.println(ch + "\t"+count[i]);
				}
			}
		}
	public static void main(String[] args) {
		String str = "srinivas";
		charFrequency(str);
	}
}

//Time Complexity - O(n)
//Aux Space --- O(26)/O(1)
```