LAB # 6

JAVA STRING CLASS

OBJECTIVE

To Study Java String class and Java String Buffer.

THEORY

Strings, which are widely used in Java programming, are a sequence of characters. In the Java programming language, strings are objects.

The Java platform provides the String class to create and manipulate strings.

Creating Strings:

The most direct way to create a string is to write:

```
String greeting = "Hello world!";
```

Whenever it encounters a string literal in your code, the compiler creates a String object with its value in this case, "Hello world!'.

As with any other object, you can create String objects by using the new keyword and a constructor. The String class has eleven constructors that allow you to provide the initial value of the string using different sources, such as an array of characters:

```
public class StringDemo{

   public static void main(String args[]){
      char[] helloArray = { 'h', 'e', 'l', 'l', 'o', '.'};
      String helloString = new String(helloArray);
      System.out.println( helloString );
   }
}
```

This would produce following result:

```
hello.
```

Note: The String class is immutable, so that once it is created a String object cannot be changed. If there is a necessity to make alot of modifications to Strings of characters then you should use String Buffer & String Builder Classes.

String Methods:

Here are the list methods supported by String class:

Methods with Description
char charAt(int index) Returns the character at the specified index.
int compareTo(Object o) Compares this String to another Object.
int compareTo(String anotherString) Compares two strings lexicographically.
int compareToIgnoreCase(String str) Compares two strings lexicographically, ignoring case differences.
String concat(String str) Concatenates the specified string to the end of this string.
boolean equals(Object anObject) Compares this string to the specified object.
boolean equalsIgnoreCase(String anotherString) Compares this String to another String, ignoring case considerations.
byte getBytes() Encodes this String into a sequence of bytes using the platform's default charset, storing the result into a new byte array.
int indexOf(int ch) Returns the index within this string of the first occurrence of the specified character.
int indexOf(int ch, int fromIndex) Returns the index within this string of the first occurrence of the specified character, starting the search at the specified index.
int indexOf(String str) Returns the index within this string of the first occurrence of the specified substring.
int lastIndexOf(int ch) Returns the index within this string of the last occurrence of the specified character.
int lastIndexOf(String str) Returns the index within this string of the rightmost occurrence of the specified substring.
int lastIndexOf(String str, int fromIndex) Returns the index within this string of the last occurrence of the specified substring, searching backward starting at the specified index.
int length() Returns the length of this string.
String replace(char oldChar, char newChar) Returns a new string resulting from replacing all occurrences of oldChar in this string with newChar.

char[] toCharArray() Converts this string to a new character array.
String toLowerCase() Converts all of the characters in this String to lower case using the rules of the default locale.
String toLowerCase(Locale locale) Converts all of the characters in this String to lower case using the rules of the given Locale.
String toString() This object (which is already a string!) is itself returned.
String toUpperCase() Converts all of the characters in this String to upper case using the rules of the default locale.
String toUpperCase(Locale locale) Converts all of the characters in this String to upper case using the rules of the given Locale.
String trim() Returns a copy of the string, with leading and trailing whitespace omitted.
static String valueOf(primitive data type x) Returns the string representation of the passed data type argument.

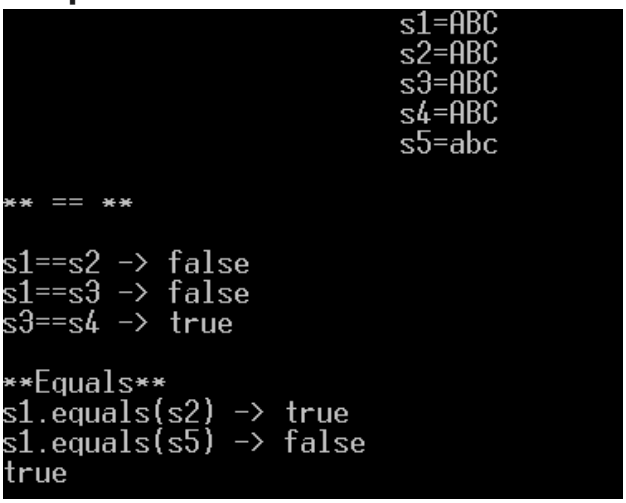
Program#1

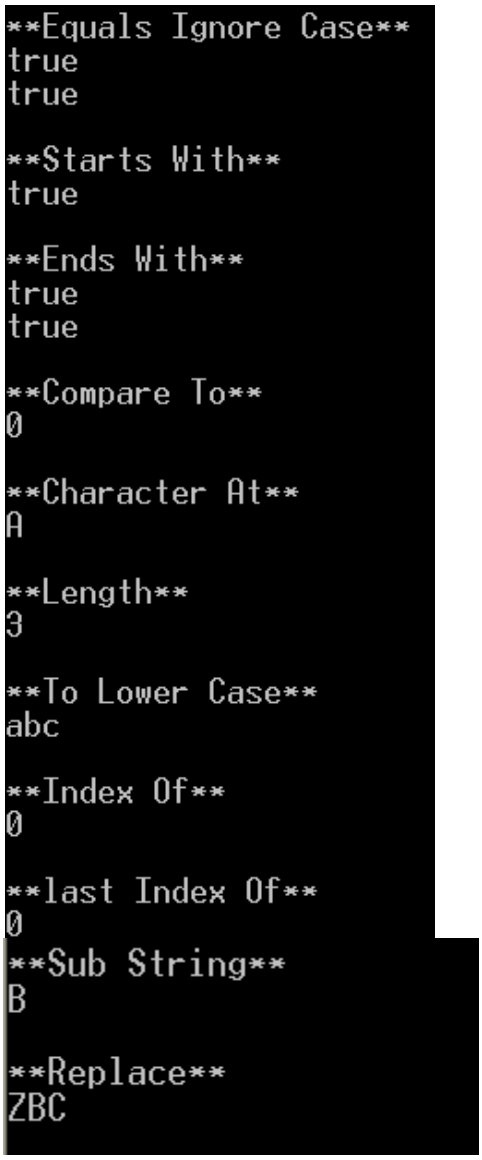
This program uses different string classes.

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

String s1 = new String("ABC");
String s2 = new String("ABC");
String s3  ="ABC";
String s4  ="ABC";
String s5 = new String("abc");

System.out.println("\t\t\ts1="+s1);
System.out.println("\t\t\ts2="+s2);
System.out.println("\t\t\ts3="+s3);
System.out.println("\t\t\ts4="+s4);
System.out.println("\t\t\ts5="+s5);

System.out.println("\n** == **");
System.out.println("\ns1==s2 -> "+(s1==s2));
System.out.println("s1==s3 -> "+(s1==s3));
System.out.println("s3==s4 -> "+(s3==s4));

//Equals
System.out.println("\n**Equals**");
System.out.println("s1.equals(s2) -> "+s1.equals(s2));
System.out.println("s1.equals(s5) -> "+s1.equals(s5));
System.out.println("XYZ".equals("XYZ"));
```

```
//Equals Ignore Case
System.out.println("\n**Equals Ignore Case**");
System.out.println(s1.equalsIgnoreCase(s5));
System.out.println("XYZ".equalsIgnoreCase("xyz"));

//Starts With
System.out.println("\n**Starts With**");
System.out.println(s1.startsWith("A"));

//Ends With
System.out.println("\n**Ends With**");
System.out.println(s1.endsWith("C"));
System.out.println("SSUET Karachi".endsWith("i"));

//Compare To
System.out.println("\n**Compare To**");
System.out.println(s1.compareTo(s2));

//Character At
System.out.println("\n**Character At**");
System.out.println(s1.charAt(0));

//Length
System.out.println("\n**Length**");
System.out.println(s1.length());

//To Lower Case
System.out.println("\n**To Lower Case**");
System.out.println(s1.toLowerCase());

//Index Of
System.out.println("\n**Index Of**");
System.out.println(s1.indexOf('A'));

//last Index Of
System.out.println("\n**last Index Of**");
System.out.println(s1.lastIndexOf('A'));

//Sub String
System.out.println("\n**Sub String**");
System.out.println(s1.substring(1,2));

//Replace
System.out.println("\n**Replace**");
System.out.println(s1.replace('A','Z'));
}
}
```

Output:


s1=ABC
s2=ABC
s3=ABC
s4=ABC
s5=abc
** == **
s1==s2 -> false
s1==s3 -> false
s3==s4 -> true
Equals
s1.equals(s2) -> true
s1.equals(s5) -> false
true


Equals Ignore Case
true
true
Starts With
true
Ends With
true
true
Compare To
0
Character At
A
Length
3
To Lower Case
abc
Index Of
0
last Index Of
0
Sub String
B
Replace
ZBC

String Buffer

StringBuffer is a peer class of **String** that provides much of the functionality of strings. As you know, **String** represents fixed-length, immutable character sequences. In contrast, **StringBuffer** represents growable and writeable character sequences. **StringBuffer** may have characters and substrings inserted in the middle or appended to the end. **StringBuffer** will automatically grow to make room for such additions and often has more characters pre allocated than are actually needed, to allow room for growth.

StringBuffer Constructors

StringBuffer defines these three constructors:

```
StringBuffer( )
StringBuffer(int size)
StringBuffer(String str)
```

The default constructor (the one with no parameters) reserves room for 16 characters without reallocation. The second version accepts an integer argument that explicitly sets the size of the buffer. The third version accepts a **String** argument that sets the initial contents of the **StringBuffer** object and reserves room for 16 more characters without reallocation. **StringBuffer** allocates room for 16 additional characters when no specific buffer length is requested, because reallocation is a costly process in terms of time. Also, frequent reallocations can fragment memory. By allocating room for a few extra characters, **StringBuffer** reduces the number of reallocations that take place.

Program#2

Creating String Buffer Objects. length, capacity, ensureCapacity, setLength, append, insert, charAt, setChatAt and toString Functions.

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

StringBuffer s1 = new StringBuffer("ABC");
StringBuffer s2 = new StringBuffer();
StringBuffer s3 = new StringBuffer(50); // Capacity 50

System.out.println("s1= "+s1);                              //Print data
System.out.println("s1.length()   = "+s1.length());        //Print length = 3
System.out.println("s1.capacity() = "+s1.capacity());      //print capacity. Default 16 + 3 = 19
System.out.println("s2.length()   = "+s2.length());        //print length = 0
System.out.println("s1.capacity() = "+s2.capacity());      //print default capacity 16
System.out.println("s3.length()   = "+s3.capacity());      //Print 50

//Append
System.out.println("**\nAppend**");
System.out.println("Capacity of S2 = "+s2.capacity()); //Capacity remain 16
System.out.println("Length of S2 = "+s2.length());
s2.append("ABCDEF");
```

```
System.out.println(s2);
System.out.println("Capacity of S2 = "+s2.capacity()); //Capacity remain 16
System.out.println("Length of S2 = "+s2.length());
s2.append("GHIJKLMNOP");
System.out.println("Capacity of S2 = "+s2.capacity()); //Capacity remain 16
System.out.println("Length of S2 = "+s2.length());
s2.append("Q");
System.out.println("Capacity of S2 = "+s2.capacity()); //Capacity 34 after length 16+
System.out.println("Length of S2 = "+s2.length());

//Character At
System.out.println("\n**Character At**");
System.out.println(s1.charAt(0));

//Set Character At
System.out.println("\n**Set Character At**");
System.out.println(s1); //Original Value
s1.setCharAt(0,'X');
System.out.println(s1); //After Change

//To String
System.out.println("\n**To String**");
System.out.println(s1); //Original Value in String Buffer
String s = s1.toString();
System.out.println(s); // Convert from String Buffer to String
}
}
```

Output:

```
s1= ABC
s1.length()   = 3
s1.capacity() = 19
s2.length()   = 0
s1.capacity() = 16
s3.length()   = 50
**
Append**
Capacity of S2 = 16
Length of S2 = 0
ABCDEF
Capacity of S2 = 16
Length of S2 = 6
Capacity of S2 = 16
Length of S2 = 16
Capacity of S2 = 34
Length of S2 = 17

**Character At**
A

**Set Character At**
ABC
XBC

**To String**
XBC
XBC
```

LAB TASK

1. Write a program that extracts username and the domain information from an E-mail address. For example, if the email address is "user@mydomain.com", your program will print

   User name = user
   Domain = mydomain
   Extension = com

2. A PALINDROME is a word which has SAME SPELLING whether it is read from Left to Right or from Right to Left. Example: MOM, DAD, DEED, PEEP and NOON. Other words which are not PALINDROME are HELLO, DOOR and FEET. Write a program that can take a String as user input in Capital Letters and then Print YES as Output if the Input is a PALINDROME otherwise NO.