

8th - 9th October 2024 - Class Objectives

- ▶ Summative Test #3 (Reminder : Due on 17th October, 2024 Thursday)
- ▶ Holiday Homework – Due on 8th October, 2024 – Before 9 am – Canvas Assignments
- ▶ Strings in Java
- ▶ String Methods in Java:
 - (a) charAt()
 - (b) equals()
 - (c) equalsIgnoreCase()
 - (d) compareTo()
 - (e) startsWith()
 - (f) endsWith()
 - (g) indexOf()
 - (h) lastIndexOf()
 - (i) toUpperCase()
 - (j) toLowerCase()
 - (k) length()
 - (l) toCharArray()
 - (m) subString()
 - (n) concat()
 - (o) contains()
 - (p) empty()
 - (q) valueOf()
 - (r) join()
 - (s) replace()
 - (t) replaceFirst()
 - (u) replaceAll()
- ▶ Java Programs + FRQs / Quiz – MCQs using Strings in Java and its methods
- ▶ Worksheets in class

Strings in Java

► What are Strings?

- A string in literal terms is a **series of characters**.
- String **isn't a primitive data type** in Java. Strings are **objects**.
- The **Java platform** provides the **String class** to **create and manipulate strings**.
- In java, string is an **immutable object** which means it is **constant** and can **cannot be changed once it has been created**.

■ Syntax:

<String_Type> <string_variable> = “<sequence_of_string>”;

■ Example:

```
String str = “Good Day!!”;
```

Strings in Java

► String Initialization in Java:-

► There are **two ways** to create a String in Java:

- a) String literal
- b) Using new keyword
- c) Using char array

■ Example: String Literal:

```
String str1 = "Welcome";
```

■ Example: new keyword (create object of the class):

```
String str1 = new String("Welcome");
```

■ Example: new keyword (create object of the class using char array as parameter):

```
private char ch [4] = {'a', 'b', 'c', '$', '3'};
```

```
private String str1 = new String(ch);
```

Strings in Java

► Example:

```
public class Example{  
    public static void main(String args[]){  
        //creating a string by java string literal  
        String str = "Welcome to String object";  
        char arrch[] = {'h','e','l','l','o'}; //creating a char array  
        //converting char array arrch[] to string str2  
        String str2 = new String(arrch);  
        //creating another java string str3 by using new keyword  
        String str3 = new String("Java String Example");
```

Strings in Java

```
//Displaying all the three strings  
System.out.println(str);  
System.out.println(str2);  
System.out.println(str3);  
}  
}
```

Strings in Java

► Java String `charAt(int index)` Method:

- The Java String `charAt(int index)` method **returns the character at the specified index** in a string.
- The `s.charAt(0)` would **return the first character of the string represented by instance s.**

Strings in Java

] ChatAtDemo.java ☒

```
1 package PackageTwo;  
2  
3 public class ChatAtDemo {  
4  
5     public static void main(String[] args) {  
6  
7         String str = "Today is a weekday";  
8  
9         //This will return the first char of the string  
10        char ch1 = str.charAt(0);  
11  
12        //This will return the 7th char of the string  
13        char ch2 = str.charAt(6);  
14  
15        System.out.println("Character at 0 index is: " + ch1);  
16  
17        System.out.println("Character at 6th index is: " + ch2);  
18  
19    }  
20  
21 }  
22 }
```

Problems @ Javadoc Declaration Console ☒

<terminated> ChatAtDemo [Java Application] C:\Program F

Character at 0 index is: T
Character at 6th index is: i

Strings in Java

- **Java String equals() and equalsIgnoreCase() Method:**
- The String equals() and equalsIgnoreCase() methods are used for **comparing two strings**.
- The String **equals()** performs **case sensitive comparison**.
- The String **equalsIgnoreCase()** performs **case in-sensitive comparison**.

Strings in Java

```
1 package PackageTwo;  
2  
3 public class StringDemo {  
4  
5     public static void main(String[] args) {  
6  
7         String str1 = "Today is a weekday";  
8         String str2 = "Today is a weekday";  
9         String str3 = "TODAY IS A WEEKDAY";  
0  
1         System.out.println(str1.equals(str2)); //returns true  
2  
3         System.out.println(str1.equals(str3)); //returns false  
4  
5         System.out.println(str1.equalsIgnoreCase(str2)); //returns true  
6     }  
7  
8 }  
9
```

```
Problems @ Javadoc Declaration Console ×  
<terminated> StringDemo [Java Application] C:\Program  
true  
false  
true
```

Strings in Java

► Java String compareTo() Method:-

- The Java String compareTo() method is used for **comparing two strings**.
- If **both the strings** are **equal** then this **method returns 0** else it **returns positive or negative value**.

Strings in Java

```
package PackageTwo;

public class StringDemo {

    public static void main(String[] args) {

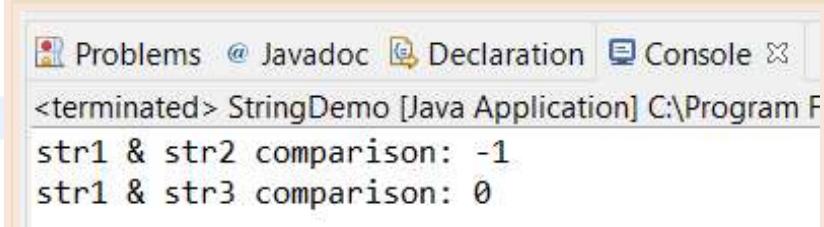
        String str1 = "String is getting compared";
        String str2 = "The compareTo method";
        String str3 = "String is getting compared";

        int var1 = str1.compareTo(str2);

        System.out.println("str1 & str2 comparison: "+var1);

        int var2 = str1.compareTo(str3);

        System.out.println("str1 & str3 comparison: "+var2);
    }
}
```



Problems @ Javadoc Declaration Console

<terminated> StringDemo [Java Application] C:\Program F

str1 & str2 comparison: -1

str1 & str3 comparison: 0

Strings in Java

► int compareTo(String other):

Returns a **value < 0** if this is **less than other**

Returns **zero** if this is **equal to other**

Returns a **value > 0** if this is **greater than other**

```
public static void main(String[] args) {  
  
    String ss1="hello";  
    String ss2="hello";  
    String ss3="meklo";  
    String ss4="hemlo";  
    String ss5="flag";  
  
    System.out.println(ss1.compareTo(ss2));//Returns 0 because both are equal  
  
    System.out.println(ss1.compareTo(ss3));//Returns -5 because "h" is 5 times lower than "m"  
  
    System.out.println(ss1.compareTo(ss4));//Returns -1 because "l" is 1 times lower than "m"  
  
    System.out.println(ss1.compareTo(ss5));//Returns 2 because "h" is 2 times greater than "f"
```

Strings in Java

► **Example:** Consider the following code segment.

```
String s1 = "avocado";
```

```
String s2 = "banana";
```

```
System.out.println(s1.compareTo(s2) + " " + s2.compareTo(s1));
```

Which of these could be the result of executing the code segment?

- (A) 0 0
- (B) -1 -1
- (C) -1 1
- (D) 1 -1
- (E) 1 1

Strings in Java

► Correct ans is C -1 and 1

Strings in Java

► Java String startsWith() Method:-

- The startsWith() method of String class is used for **checking prefix** of a **String**.
- It **returns a boolean value true or false** based on **whether** the given string begins with the **specified letter or word**.

► Java String endsWith() Method :-

- Java String endsWith(String suffix) method checks whether the **String ends** with a **specified suffix**.
- This method **returns a boolean value true or false**.
- If the **specified suffix** is **found** at the **end** of the **string** then it **returns true** else it **returns false**.

Strings in Java

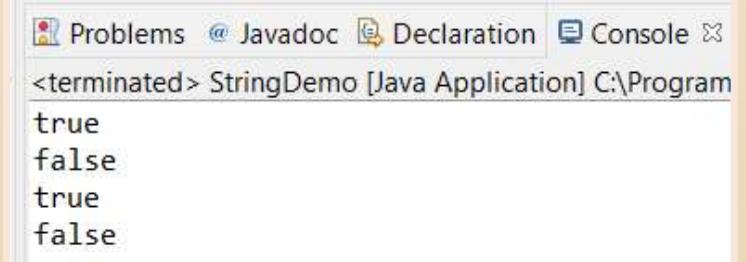
```
package PackageTwo;

public class StringDemo {
    public static void main(String[] args) {
        String s = "This is a sample String";
        //checking whether the given string starts with "This"
        System.out.println(s.startsWith("This"));

        //checking whether the given string starts with "Hi"
        System.out.println(s.startsWith("Hi"));

        //checking whether the given string ends with "String"
        System.out.println(s.endsWith("String"));

        //checking whether the given string ends with "in"
        System.out.println(s.endsWith("in"));
    }
}
```



The screenshot shows the Eclipse IDE interface with the 'Console' tab selected. The output window displays the following text:

```
Problems @ Javadoc Declaration Console <terminated> StringDemo [Java Application] C:\Program
true
false
true
false
```

Strings in Java

- **Java String indexOf(int ch) Method :-**
- Java String indexOf() method is used to find the **index of a specified character or a substring** in a given String.
- It returns the **index of the first occurrence of character ch** in a given String.

Strings in Java

```
package PackageTwo;

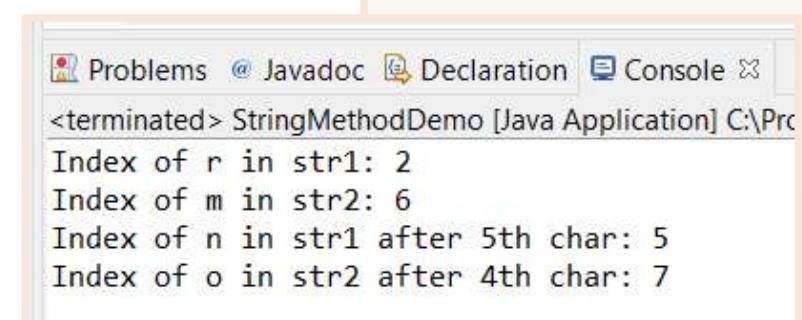
public class StringMethodDemo {
    public static void main(String[] args) {
        String str1 = new String("Morning sunrise");
        String str2 = new String("Night moon");

        System.out.println("Index of r in str1: "+str1.indexOf('r'));

        System.out.println("Index of m in str2: "+str2.indexOf('m'));

        System.out.println("Index of n in str1 after 5th char: " + str1 .indexOf('n', 5));

        System.out.println("Index of o in str2 after 4th char: "+ str2 .indexOf('o', 4));
    }
}
```



The screenshot shows the Java application window with the following details:

- Toolbar: Problems, @ Javadoc, Declaration, Console.
- Console tab: Selected.
- Output pane:
 - <terminated> StringMethodDemo [Java Application] C:\Pro
 - Index of r in str1: 2
 - Index of m in str2: 6
 - Index of n in str1 after 5th char: 5
 - Index of o in str2 after 4th char: 7

Strings in Java

► Java String lastIndexOf() Method:-

- The lastIndexOf() method which is used to find out the **index of last occurrence** of a character or a substring in a given String.

```
package PackageTwo;

public class StringMethodDemo {

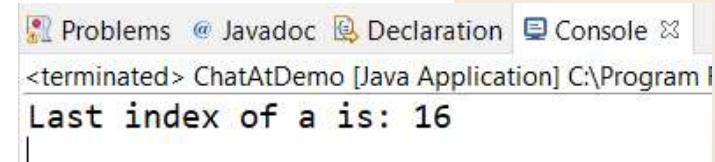
    public static void main(String[] args) {

        String str1 = new String("Sunrise");
        String str2 = new String("Sunset");

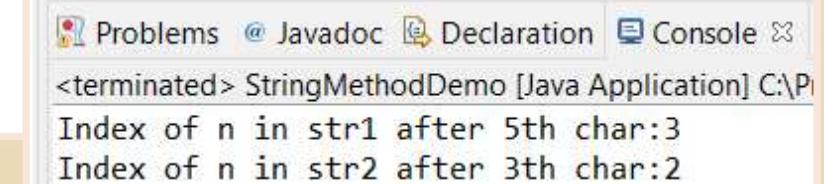
        System.out.println("Index of n in str1 after 5th char:" + str1.lastIndexOf('r'));

        System.out.println("Index of n in str2 after 3th char:" + str2.lastIndexOf('n'));
    }
}
```

```
String str = "Today is a weekday";
int i = str.lastIndexOf('a');
System.out.println("Last index of a is: " + i);
```



Problems @ Javadoc Declaration Console <terminated> ChatAtDemo [Java Application] C:\Program Files\Java\chatat\src\com\javatutorial\StringMethodDemo.java Last index of a is: 16



Problems @ Javadoc Declaration Console <terminated> StringMethodDemo [Java Application] C:\Program Files\Java\chatat\src\com\javatutorial\StringMethodDemo.java Index of n in str1 after 5th char:3 Index of n in str2 after 3th char:2

Strings in Java

Java String toLowerCase() and toUpperCase():-

- The method toLowerCase() converts the characters of a String into **lower case characters**.
- The method toUpperCase() converts the characters of a String into **upper case characters**.

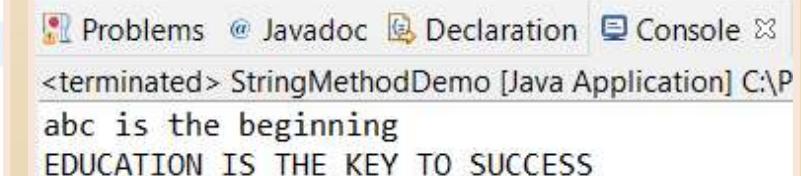
```
package PackageTwo;

public class StringMethodDemo {

    public static void main(String[] args) {

        String str = new String("ABC IS THE BEGINNING");
        System.out.println(str.toLowerCase());

        String word = "Education is the key to success";
        System.out.println(word.toUpperCase());
    }
}
```



Problems @ Javadoc Declaration Console <terminated> StringMethodDemo [Java Application] C:\P
abc is the beginning
EDUCATION IS THE KEY TO SUCCESS

Strings in Java

Java String length() Method :-

- Java String length() method is used to find out the **length** of a String.
- This method returns an integer number which represents the number of characters (length) in a given string **including white spaces**.

```
package PackageTwo;

public class StringMethodDemo {

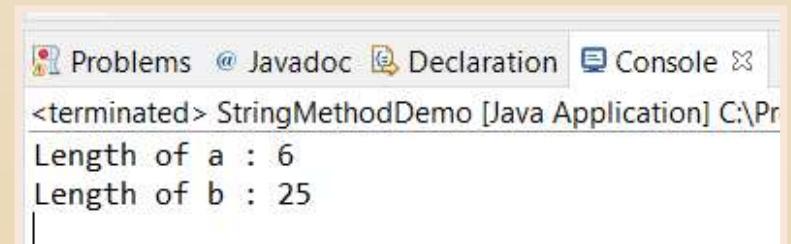
    public static void main(String[] args) {

        String a = new String("String");

        String b = new String("Immutable Strings in Java");

        System.out.println("Length of a : " + a.length());

        System.out.println("Length of b : " + b.length());
    }
}
```



The screenshot shows the Eclipse IDE interface with the 'Console' tab selected. The console window displays the output of the Java application. It shows two lines of text: 'Length of a : 6' and 'Length of b : 25'. The console tab also includes tabs for 'Problems', '@ Javadoc', 'Declaration', and 'Console'.

```
Problems @ Javadoc Declaration Console ×
<terminated> StringMethodDemo [Java Application] C:\Pr
Length of a : 6
Length of b : 25
```

Strings in Java

Java – String toCharArray() Method:-

- The method **toCharArray()** returns an **Array of chars** after converting a String into sequence of characters.

```
package PackageTwo;

public class StringMethodDemo {

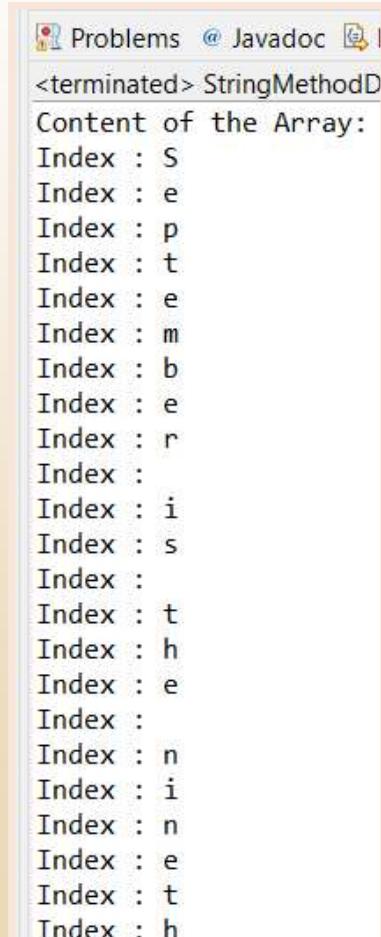
    public static void main(String[] args) {

        String one = "September is the nineth month of the year";
        |  
        char[] arr = one.toCharArray();

        System.out.println("Content of the Array: ");

        for(char c : arr) {

            System.out.println("Index : " + c + " ");
        }
    }
}
```



```
Problems @ Javadoc
<terminated> StringMethodD
Content of the Array:
Index : S
Index : e
Index : p
Index : t
Index : e
Index : m
Index : b
Index : e
Index : r
Index :
Index : i
Index : s
Index :
Index : t
Index : h
Index : e
Index :
Index : n
Index : i
Index : n
Index : e
Index : t
Index : h
```

Strings in Java

➤ The `substring()` Method:

- The method `substring()` **returns a new string** that is a **substring** of given **string**.
- There are **two ways** to use this method:

a) `String substring(int beginIndex)`:

- Returns the substring **starting from the specified index** (`beginIndex`) and **extends to the character present at the end of the string**.

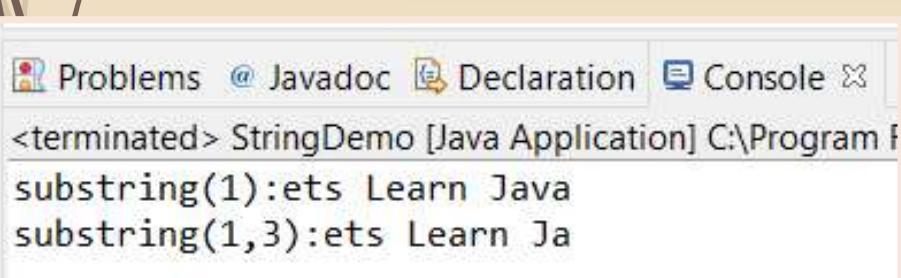
b) `String substring(int beginIndex, int endIndex)` :

- Returns a **new string** that is a **substring of this string**.
- The substring **begins** at the **specified index** (`beginIndex`) and **extends to the character at index `endIndex - 1`**.

Strings in Java

```
package PackageTwo;

public class StringDemo {
    public static void main(String[] args) {
        String mystring = new String("Lets Learn Java");
        System.out.println("substring(1):" + mystring.substring(1));
        System.out.println("substring(1,3):" + mystring.substring(1,13));
    }
}
```



A screenshot of an IDE interface showing Java code and its execution results. The code defines a class `StringDemo` with a `main` method that creates a `String` object `mystring` with the value "Lets Learn Java". It then prints two substrings: `substring(1)` and `substring(1,3)`.

The IDE interface includes tabs for Problems, Javadoc, Declaration, and Console. The Console tab shows the output:

```
<terminated> StringDemo [Java Application] C:\Program F
substring(1):ets Learn Java
substring(1,3):ets Learn Ja
```

Strings in Java

Java String concat() Method:

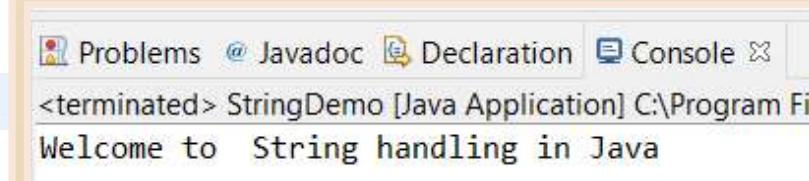
- Java string concat() method **concatenates multiple strings**.
- This method **appends the specified string at the end of the given string and returns the combined string**.

```
package PackageTwo;

public class StringDemo {

    public static void main(String[] args) {

        String x = "Welcome";
        x = x.concat(" to ");
        x = x.concat(" String handling in Java ");
        System.out.println(x);
    }
}
```



The screenshot shows the Eclipse IDE interface with the 'Console' tab selected. The console window displays the output of the Java application 'StringDemo'. The output text is 'Welcome to String handling in Java', which is the result of concatenating the four strings defined in the code.

```
Problems @ Javadoc Declaration Console
<terminated> StringDemo [Java Application] C:\Program Fi
Welcome to String handling in Java
```

Strings in Java

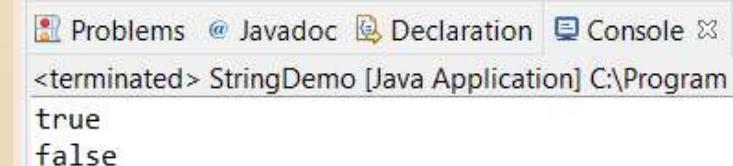
- Java String contains() method:
- Java String contains() method checks whether a **particular sequence of characters is part of a given string or not.**
- This method **returns true** if a specified sequence of characters is present in a given string, otherwise it **returns false**.

```
package PackageTwo;

public class StringDemo {

    public static void main(String[] args) {

        String mystr = "Do you like watching Game of Thrones";
        System.out.println(mystr.contains("like")); // returns true
        System.out.println(mystr.contains("game")); // returns false
    }
}
```



The screenshot shows the Java IDE interface with the following details:

- Top bar: Problems, @ Javadoc, Declaration, Console, X
- Message bar: <terminated> StringDemo [Java Application] C:\Program
- Console output:

```
true
false
```

Strings in Java

Java String isEmpty() method:

- Java String isEmpty() method checks whether a **String** is **empty** or **not**.
- This method returns **true** if the given **string** is **empty**, **else** it returns **false**.

```
package PackageTwo;

public class StringDemo {

    public static void main(String[] args) {

        //empty string
        String str3="";
        //non-empty string
        String str4="hello";
        //prints true
        System.out.println(str3.isEmpty());
        //prints false
        System.out.println(str4.isEmpty());
    }
}
```

Problems
<terminated
true
false

Strings in Java

Java String valueOf() method:-

The Java String valueOf() method is used to find the **value at a variable**.

```
package PackageTwo;

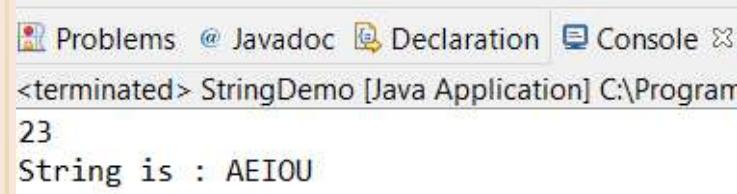
public class StringDemo {

    public static void main(String[] args) {

        int number = 23;
        String str = String.valueOf(number);
        System.out.println(str);

        //converting an array to a string
        char vowel[] = {'A', 'E', 'I', 'O', 'U'};
        String w = String.valueOf(vowel);
        System.out.println("String is : " + w);

    }
}
```



A screenshot of an IDE interface showing the output window. The title bar includes 'Problems', '@ Javadoc', 'Declaration', and 'Console'. The message area shows the output of the program: '<terminated> StringDemo [Java Application] C:\Program' followed by the two lines of output: '23' and 'String is : AEIOU'.

Strings in Java

- Java String join() method:
- Java String join() method **concatenates the given Strings and returns the concatenated String.**

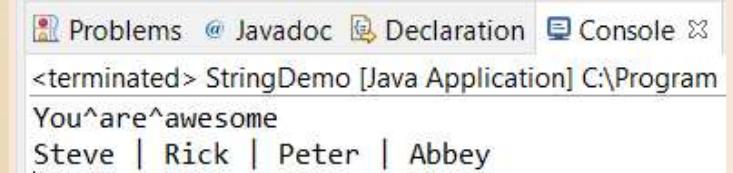
```
package PackageTwo;

public class StringDemo {

    public static void main(String[] args) {

        //The first argument to this method is the delimiter
        String strrr=String.join("^","You","are","awesome");
        System.out.println(strrr);

        //Converting an array of String to the list
        String[] list = new String[] {"Steve", "Rick", "Peter", "Abbey"};
        String names = String.join(" | ", list);
        System.out.println(names);
    }
}
```



The screenshot shows an IDE interface with a code editor and a terminal window. The code editor contains the Java code provided above. The terminal window shows the output of the program, which consists of two lines of text. The first line is the result of the first String.join() call, showing the strings "You", "are", and "awesome" joined by a caret character (^). The second line is the result of the second String.join() call, showing the array elements "Steve", "Rick", "Peter", and "Abbey" joined by a vertical bar (|) character.

```
Problems @ Javadoc Declaration Console ×
<terminated> StringDemo [Java Application] C:\Program
You^are^awesome
Steve | Rick | Peter | Abbey
```

Strings in Java

Java String replace() method:-

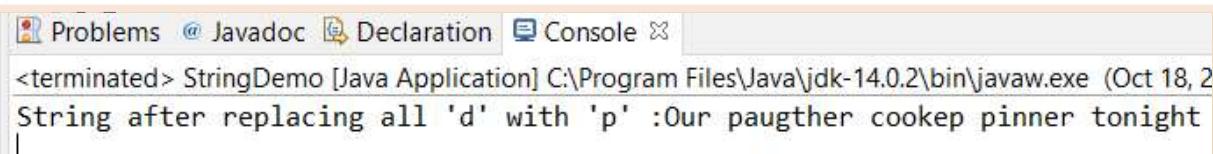
- String replace(char oldChar, char newChar):

It replaces all the occurrences of a **oldChar** character with **newChar** character.

- For example, "pog pance".replace('p', 'd') would return dog dance.

```
package PackageTwo;

public class StringDemo {
    public static void main(String[] args) {
        String str = new String("Our daughter cooked dinner tonight");
        System.out.print("String after replacing all 'd' with 'p' :");
        System.out.println(str.replace('d', 'p'));
    }
}
```



The screenshot shows an IDE interface with a code editor and a terminal window. The code editor contains the Java code provided above. The terminal window shows the output of running the program: it prints the string "String after replacing all 'd' with 'p' :" followed by the modified string "Our paugther cookep pinner tonight".

```
Problems @ Javadoc Declaration Console
<terminated> StringDemo [Java Application] C:\Program Files\Java\jdk-14.0.2\bin\javaw.exe (Oct 18, 2023)
String after replacing all 'd' with 'p' :Our paugther cookep pinner tonight
```

Strings in Java

Java String replaceFirst() Method:-

- The replaceFirst() replaces the **first occurrence** of a string with a **new specified string**.

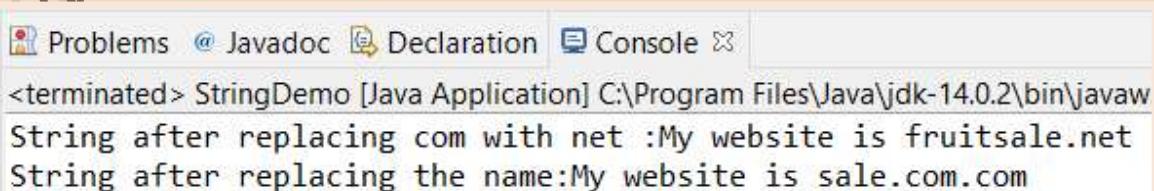
```
package PackageTwo;

public class StringDemo {

    public static void main(String[] args) {

        String a = new String("My website is fruitsale.com");
        System.out.print("String after replacing com with net :");
        System.out.println(a.replaceFirst("com", "net"));
        System.out.print("String after replacing the name:");
        System.out.println(a.replaceFirst("fruitsale", "sale.com"));

    }
}
```



A screenshot of an IDE interface showing Java code and its execution results. The code defines a class StringDemo with a main method that prints two modified strings. The first modification replaces 'com' with 'net', resulting in 'My website is fruitsale.net'. The second modification replaces 'fruitsale' with 'sale.com', resulting in 'String after replacing the name:My website is sale.com.com'. The IDE tabs at the top include 'Problems', '@ Javadoc', 'Declaration', and 'Console'.

```
Problems @ Javadoc Declaration Console <terminated> StringDemo [Java Application] C:\Program Files\Java\jdk-14.0.2\bin\javaw
<terminated> StringDemo [Java Application] C:\Program Files\Java\jdk-14.0.2\bin\javaw
String after replacing com with net :My website is fruitsale.net
String after replacing the name:My website is sale.com.com
```

Strings in Java

Java String replaceAll() Method:-

- The replaceAll() method **replaces all the occurrences of old string with the new string.**

```
package PackageTwo;

public class StringDemo {

    public static void main(String[] args) {

        String t = new String("My .com site is WayToHeaven.com");
        System.out.print("String after replacing all com with net: " );
        System.out.println(t.replaceAll("com", "net"));

    }
}
```

Problems @ Javadoc Declaration Console

<terminated> StringDemo [Java Application] C:\Program Files\Java\jdk-14.0.2\bin\javaw.exe (Oct 1
String after replacing all com with net: My .net site is WayToHeaven.net

Java Programs

- Q1) Write the following Java Programs using for loop, while loop and do..while loop:
- a) The sum of all even numbers between a and b (all inclusive), where a and b are inputs.
- Q2) Write a Java program to enter the numbers till the user wants and at the end it should display the count of positive, negative and zeros entered.



End

