Module no : 6 Strings Date : 16/10/2023

Agenda :-

1. String class and objects
2. String methods
3. String using new
4. String pool
5. Extracting characters
6. String Builder
7. Char arrays vs strings

String class :-

Definition :- The string class represents character strings. All string literals in in java programs, such as "abc", are implemented as instances of this class. Strings in java are immutable

Sample program :-

```
class Main {
    public static void main (String args[]) {
        String str = "abc";
    }
}
```

Note :-

* Memory allocation is similar to arrays.

String Classes:

Program:

```
package com.company;

public class Main {
    public static void main(String args[]){
        String s = "abc123"; //String literal
        System.out.println(s);
        System.out.println(s.length());
        System.out.println(s.charAt(5));
        System.out.println(s.charAt(2));
    }
}
```

Output:

```
"C:\Program Files\Java\jdk-21\bin\java.exe"
abc123
6
3
c

Process finished with exit code 0
```

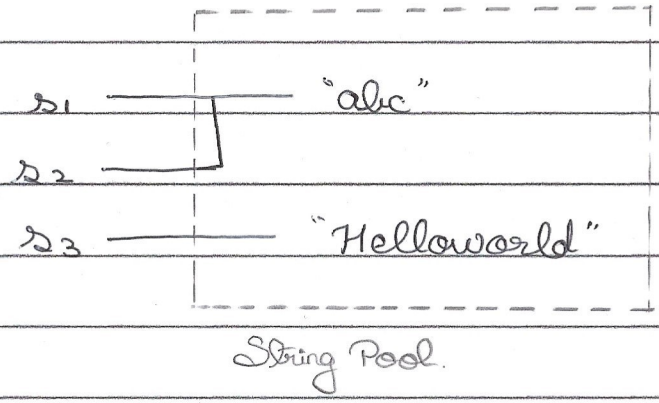
String Pool :-

Definition :- The string pool is the area in the heap memory where string literals are stored.

Example :-

We have 2 strings :

```
String s1 = "abc";
String s2 = "abc";
String s3 = "Helloworld";
```



reduces space in java.

String Pool:

Program:

```
package com.company;

public class Main {
    public static void main(String args[]){
        String s = "abc123";
        String s_1 = "abc123";
        // s and s_1 both shares the same memory, only the variable name differs
        boolean isSame = s == s_1;
        System.out.println(isSame);
    }
}
```

Output:

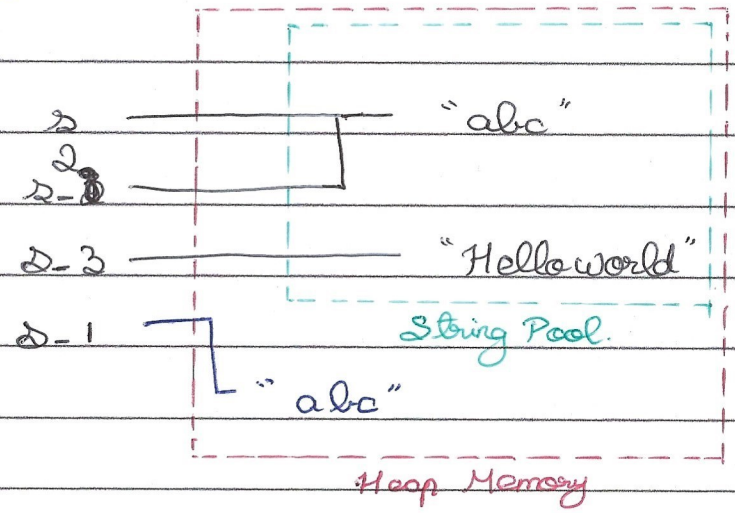
```
"C:\Program Files\Java\jdk-21\bin\java.exe"
true

Process finished with exit code 0
```

Strings using new :-

Definition : We can also create new strings using the new keyword. Those strings do not reside in the string pool, instead they are created and stored in the heap memory.

Example :-

```
String s = "abc";
String s-1 = new String("abc");
String s-2 = "abc";
String s-3 = "Hello World";
```



String Using 'new':

Program:

```
package com.company;

public class Main {
    public static void main(String args[]){
        String s = "abc123";
        String s_1 = new String("abc123");

        boolean isSame = s == s_1;
        System.out.println(isSame);

        boolean isEqual = s.equals(s_1);
        System.out.println("Using 's.equals(s_1)' function:" + isEqual);
    }
}
```

Output:

```
"C:\Program Files\Java\jdk-21\bin\java.exe"
false
Using 's.equals(s_1)' function:true

Process finished with exit code 0
```

String Methods :-

String class method :- String class offers several methods out of the box. We will be looking at some of them in this Module.

① CharAt :-

CharAt method allows us to access the character at the specified index. [String str = "Hello world" ; str.charAt(5);]

② length :-

The length() Method returns the length of the string.

[String str = "Hello World" ; str.length() ;]

③ indexOf :-

Index of methods returns the index of the first occurence of the specified char or string in the given string. if it doesn't present, then it returns -1. [String str = "Hello World"; str.indexOf("o");]

④ equals :-

Equals method is used to compare whether the two strings contain the same sequence of characters. [String str = "Hello world"; String str-1 = new String ("Hello World") ; System.out.println(str == str2); System.out.println (str.equals(str-1));]

⑤ Contains :-

Returns true if and only if this string contains the specified sequence of char values. [String str = "Hello World"; str.contains("ll");]

⑥ toLowerCase, toUpperCase :-

Returns new strings after changing the case. Original string remains intact. [String str = "Hello World"; str.toLowerCase(); str.toUpperCase();]

⑦ replace :-

Replaces the target string (first) with the given replacement string (second) and returns a new string. [String str = "I love code"; str.replace("code", "Java");]

⑧ substring :-

Returns a string that is a substring of this string. The substring begins at the specified beginIndex and extends to the character at index endIndex - 1. Thus the length of substring is endIndex - beginIndex. If endIndex is not given, it is taken as string length. [String str = "I love Programming"; str.substring(7); str.substring(2,6);]

Output:

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent
str.charAt(8) ==> r
str.length() ==> 11
str.indexOf('o') ==> 4
str.indexOf('abc') ==> -1
str.equals(str_1) ==> true
false
str.contains("ll") ==> true
str.contains("abc") ==> false
str.toUpperCase() ==> HELLO WORLD
str.toLowerCase() ==> hello world
str_2.replace("Programming","Java") ==> I Love Java
str_2.substring(7) ==> Programming
str_2.substring(2,6) ==> Love

Process finished with exit code 0
```

String Class methods:

Program:

```
package com.company;

public class Main {
    public static void main(String args[]){
        // charAt
        String str = "Hello World";
        System.out.println("str.charAt(8) ==> "+str.charAt(8));
        // length
        System.out.println("str.length() ==> "+str.length());
        // indexOf
        System.out.println("str.indexOf('o') ==> "+str.indexOf('o'));
        System.out.println("str.indexOf('abc') ==> "+str.indexOf("abc"));
        // equals
        String str_1 = new String("Hello World");
        System.out.println("str.equals(str_1) ==> "+str.equals(str_1));
        System.out.println("(str==str_1) ==> "+str==str_1);
        // contains
        System.out.println("str.contains(\"ll\") ==> "+str.contains("ll"));
        System.out.println("str.contains(\"abc\") ==> "+str.contains("abc"));
        // toUpperCase, toLowerCase
        System.out.println("str.toUpperCase() ==> "+str.toUpperCase());
        System.out.println("str.toLowerCase() ==> "+str.toLowerCase());
        // replace
        String str_2 = "I Love Programming";
        System.out.println("str_2.replace(\"Programming\",\"Java\") ==> "+ str_2.replace("Programming", "Java"));
        // substring
        System.out.println("str_2.substring(7) ==> "+str_2.substring(7));
        System.out.println("str_2.substring(2,6) ==> "+str_2.substring(2,6));
    }
}
```

Reverse of a String :-

Task :- Given a string as input, print its reverse.

Sample input :- hello

Sample output :- olleh

Reversing a String - Problem:

Program:

```
package com.company;

import java.util.*;
public class Main {
    public static void main(String args[]){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the String which you need to reverse: ");
        String str = sc.nextLine();
        String reversed = "";
        for (int i = str.length()-1; i >= 0; i--){
            reversed+=str.charAt(i);
        }
        System.out.println("The reversed String ==> "+reversed);
    }
}
```

Output:

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaa
Enter the String which you need to reverse: hello
The reversed String ==> olleh

Process finished with exit code 0
```

String Builder :-

Definition :-

Java StringBuilder class is used to create mutable (modifiable) string.

Sample program :-

```
class Main {
    public void static main (String args[]) {
        StringBuilder sb = new StringBuilder ("Hello");
        sb.append (" World");
        System.out.println (sb);    // Hello World.
        sb.insert (5, '_');
        System.out.println (sb);    // Hello_World
        sb.replace (5, sb.length (), "Java");
        System.out.println (sb);    // Hello Java.
    }
}
```

String Builder:

Program:

```
package com.company;

public class Main {
    public static void main(String args[]){
        StringBuilder sb = new StringBuilder("Hello");
        sb.append(" World");
        System.out.println(sb);
        sb.insert(5,"_");
        System.out.println(sb);
        sb.replace(5, sb.length(), ", I Love Programming");
        System.out.println(sb);
        sb.delete(0,7);
        System.out.println(sb);
        String str = sb.toString();
        System.out.println(str);
    }
}
```

Output:

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-ja
Hello World
Hello_ World
Hello, I Love Programming
I Love Programming
I Love Programming

Process finished with exit code 0
```

Is it Palindrome?

Task:- Given a string check whether it is a palindrome or not.

Sample input : 2 toneet

Sample output : Yes.

Is It a Palindrome.? - Problem:

Program:

```
package com.company;

import java.util.*;
public class Main {
    public static void main(String args[]){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the String which you need to Check Plaindrome: ");
        String str = sc.nextLine();
        String reversed = "";
        for (int i = str.length()-1; i >= 0; i--){
            reversed+=str.charAt(i);
        }
        System.out.println("The reversed String ==> "+reversed);
        if (str.equals(reversed)){
            System.out.println("It is a Palindrome String");
        }
        else{
            System.out.println("It is not a Palindrome String");
        }
    }
}
```

Output:

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaag
Enter the String which you need to reverse: tenet
The reversed String ==> tenet
It is a Palindrome String

Process finished with exit code 0
```
