String:

String is a final class.

1.String is non-primitive data type.

2.String size cannot be fixed.

3.String internally stored data as character only.

3.String is the sequence of character.(Array of character)

Char [] c={‘a’,’b’,’c’,’d’,’e’}

String s= new String[c]

5.String is immutable object.

To create String there are three classes:

1.String

2.StringBuffer

3.StringBuilder

IMP:equals and hashcode method is not ovverriden in strinbuffer and stringbuilder.

So, you can compare with toString ().

Ex:public class StringBufferComparison {

public static void main(String[] args) {

StringBuffer sb1 = new StringBuffer("Hello");

StringBuffer sb2 = new StringBuffer("Hello");

// Convert to String and compare

if (sb1.toString().equals(sb2.toString())) {

System.out.println("Both StringBuffers have the same content.");

} else {

System.out.println("StringBuffers are different.");

}

}

}

(+) operator only check the SCP.otherwise .concat and any other method always create new String Object.

"Hello" + "Hello" : Only this is reused

"Hello" + str :Create new Object

str+"Hello" :Create new Object

str+str1:Create new Object.

StringBuffer:Synchronized,Thread safe all methods

.equals() method not used directly then it gives false in stringbuffer.

USE:

Ex: str.toString().equals(str2.toString());

And : used .contentEquals().

String Constant Pool:

String constant pool is special memory location present in heap area which used to store string literals.

String constant pool is not apllicaple for grabage collector because jvm internallly refer the string.

Ex. String s = new string(“kaushik”)

This object is store in heap area and constant poll (created two object)

When same object created then only store in heap area(create one object).

String s = “kaushik”;

In this time directly memory accupy into the string constant pool only one object created in scp.

Why String are immutable in java.

If String were mutable, an attacker could modify the password in memory without creating a new object.

Since String is immutable, the original password stays unchanged, and a new object is created instead.

Final keyword:

If we declare using class final keyword then cannot extend class.

If we create final method that cannot overide.

We declare variable using final keyowrd then value has be fixed(Cannot change),and cannot resign the new value.

Final keyword are used to class, method and variable.

Question:

Why string class is final?

Ans. String class is final because it cannot inherit other class properties and because of this no one can modify the features and make string class is final.

Question:

Difference between == and equals operator?

== this operator is used to compare the address of object.

Ex. String s = new String(“kaushik”);

String s2 = new String(“kaushik”);

Sout(s=s2); Output: false

String s3 = “Amit”;

String s4 = “Amit”;

Sout(s3=s4) Output:true

.Equals:

.equals methods is used to compare the content of objects.

Ex. String s = new String(“kaushik”);

String s2 = new String(“kaushik”);

Sout(s.equals(s2)); Output:true

String Constructor:

No args constructor:

public class Stringcons {  
  
 Stringcons()  
 {  
  
 }  
 public static void main(String[] args) {  
 String s = new String();  
   
 }  
}

it create a empty object

String literal constructor:

It is create a two object.

public class Stringcons {  
  
   
 public static void main(String[] args) {  
 String s = new String("deepak");  
  
 }  
}

StringBuffer and StringBuilder constructor

It create mutable object.

public class Stringcons {  
  
  
 public static void main(String[] args) {  
 StringBuffer sb = new StringBuffer("kaushik");  
 String s2 = new String(sb);  
 System.*out*.println(s2);  
  
 }  
}

same as stringbuilder.

Bytes array:

Byte array can be passed as string constructor.

public class Stringcons {  
  
  
 public static void main(String[] args) {  
 byte[] b={101,102,103};  
 String s2 = new String(b);  
 System.*out*.println(s2);  
  
 }  
}

output:efg

Char array[]:

public class Stringcons {  
  
  
 public static void main(String[] args) {  
 char[] b={'a','b','c'};  
 String s2 = new String(b);  
 System.*out*.println(s2);  
  
 }  
}

output:abc

Question:

Why char array is preferred over string for storing passwords?

Ans: String objects are immutable then value cannot change Garbage collector is not applicaple of string pool.

When we created string object We store password in string object then object will not delete because string constant pool is not applicaple for garbage collector,   
So the object will remain there, there is a chance that someone can hack your password.

If we print char array then it prints the reference.

If we store password in char array then it will not heck.

String Methods:

1 .length

It is provide the length of string.

If string value is null then throw the null pointer exception.

2 .isempty

It check the string is empty or not.

Empty method return bollean values.

public class Stringcons {  
  
 public static void main(String[] args) {  
 String name="kaushik";  
 System.*out*.println(name.isEmpty());  
 if(name.isEmpty()==true)  
 {  
 System.*out*.println(name);  
 }else {  
 System.*out*.println("not name");  
 }  
  
 }  
}

3 .trim()

Trim method is used to remove spaces from front and end.

public class Stringcons {  
  
 public static void main(String[] args) {  
 String name=" kaus hik im fine";  
 System.*out*.println(name.trim());  
 }  
}

public class Stringcons {  
  
 public static void main(String[] args) {  
 String name=" kaus hik im fine";  
 System.*out*.println(name.trim().isEmpty());  
 System.*out*.println(name.length());  
 }  
}

Output:

false

21

4 .equals ()

It is compare the content of string.

It is provide the boolean result.

5 .equalsIgnorecase ()

It is provide the boolean result.

In this method is use to uper case and lower case value change.

Example:

public class Stringcons {  
  
 public static void main(String[] args) {  
 String s1 = " kaushik";  
 String s2 = "KAUSHIK";  
  
 System.*out*.println(s1.equals(s2));  
 System.*out*.println(s1.equalsIgnoreCase(s2));  
 System.*out*.println(s1.trim());  
 System.*out*.println(s2.isEmpty());  
 System.*out*.println(s2.length());  
 }  
}

Output:

false

false

kaushik

false

7

Example:

import java.util.Scanner;  
  
public class Check {  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
 String password = sc.next();  
 String conpassword = sc.next();  
 if(password.equals(conpassword))  
 {  
 System.*out*.println("go to home page");  
 }else{  
 System.*out*.println("wrong password");  
 }  
 }  
}

6 .compareTo()

Compareto method return a value in integer.

It is use to compare to string,that convert to string ASCII value

and compare if two string are equal or not.

7 .compareToIgnorecase()

In this method provide the positive and negative value.

It is use to compare to string.

Example:

public class Comparing {  
 public static void main(String[] args) {  
 /\*String s1 = "a";  
 String s2 = "A";  
 System.out.println(s1.compareTo(s2));\*/  
 //Ascii value of a=97 and A=65 output:a-A =>97-65 =>32  
 /\* String s1 = "A";  
 String s2 = "a";  
 System.out.println(s1.compareTo(s2)); //Output:-32\*/  
  
 /\* String s1 = "abc";  
 String s2 = "C";  
 System.out.println(s1.compareTo(s2));  
 //if value is mathced and compare the string Output:30\*/  
 /\*String s1 = "a";  
 String s2 = "A";  
 System.out.println(s1.compareToIgnoreCase(s2));\*/  
 //Output= 0  
 String s1 = "Ab";  
 String s2 = "ab";  
 System.*out*.println(s1.compareToIgnoreCase(s2));  
 }  
}

Normal concat string:

public class Concat {  
 public static void main(String[] args) {  
 String s1 = "kaushik";  
 String s2 = "Prasad";  
 System.*out*.println(s1+s2);  
 System.*out*.println(s1+10);  
 System.*out*.println(s1+20+30);  
 System.*out*.println(s1+10+s2);  
 System.*out*.println(10+s1+s2);  
 System.*out*.println(s1+20/2);  
 System.*out*.println(s1+20\*2);  
 System.*out*.println(20\*2+s1);  
 System.*out*.println(20/2+s1);  
 System.*out*.println(10-5+s1);  
 //System.out.println(s1+10-5); Provide the errorr  
 }  
}

Output:

kaushikPrasad

kaushik10

kaushik2030

kaushik10Prasad

10kaushikPrasad

kaushik10

kaushik40

40kaushik

10kaushik

8 .concat()

In this method is use two string

public class Concat {  
 public static void main(String[] args) {  
 String s1 = "kaushik";  
 String s2 = "Prasad";  
 System.*out*.println(s1.concat(s2));  
   
 }  
}

9 .String.join(“ adding value”,String,String)

This method is used to add value,symbol between two string.

Ex:

public class Concat {  
 public static void main(String[] args) {  
 String s1 = "kaushik";  
 String s2 = "Prasad";  
 String s3 = "Hello";  
 String s4 = "My name is kaushik";  
 System.*out*.println(String.*join*("@",s1,s2));  
 System.*out*.println(String.*join*("10",s1,s2));  
 System.*out*.println(String.*join*(",",s3,s4));  
  
 }  
}

Output:

kaushik@Prasad

kaushik10Prasad

Hello,My name is kaushik

10 .subSequence(strat index,last index)

In this method is used to provide the result from starting to last position.

11 .subString()

In this method is used to provide the result from starting to last index position,but we provide only one index then result is provide index to last index.

Example:

public class Subs {  
 public static void main(String[] args) {  
 String s1 = "My,name is kaushik";  
 System.*out*.println(s1.subSequence(0,10));  
 System.*out*.println(s1.substring(3));  
 System.*out*.println(s1.substring(0,10));  
 }  
}

Output:

My name is

name is kaushik

My name is

12 .replace()

Replace method is used to replace the value.

13.replacefirst()

This method is used to remove first time only.

14.replaceAll()

This method is used to replace all values.

Example:

public class Replaces {  
 public static void main(String[] args) {  
 String name = "my name is kaushik Mahto i am from ranchi jharkhand i have completed my Master is SBU";  
 System.*out*.println(name.replace("is","me"));  
 System.*out*.println(name.replace("Mahto","Prasad"));  
 System.*out*.println(name.replaceFirst("is","in"));  
 System.*out*.println(name.replaceAll("is","in"));  
  
 }  
}

Output:

my name me kaushik Mahto i am from ranchi jharkhand i have completed my Master me SBU

my name is kaushik Prasad i am from ranchi jharkhand i have completed my Master is SBU

my name in kaushik Mahto i am from ranchi jharkhand i have completed my Master is SBU

my name in kaushik Mahto i am from ranchi jharkhand i have completed my Master in SBU

Searching string method:

15 .indexOf

This method is provide the index position value of input string,char,array.

Example:

import java.sql.SQLOutput;  
  
public class Searching {  
 public static void main(String[] args) {  
 String name = "kaushik";  
 System.*out*.println(name.indexOf("s"));  
 System.*out*.println(name.indexOf("z"));  
 System.*out*.println(name.indexOf(" "));  
 }  
}

Output:

3

-1

-1

16 .lastIndexOf()

This method is provide the index position value.

In this method started to find last position to start indexing.

import java.sql.SQLOutput;  
  
public class Searching {  
 public static void main(String[] args) {  
 String name = "kaushaik";  
 System.*out*.println(name.indexOf("a"));  
 System.*out*.println(name.lastIndexOf("a"));  
 }  
}

Output:

1

5

17 .charAt()

This method provide the char position value of the given input.

import java.sql.SQLOutput;  
  
public class Searching {  
 public static void main(String[] args) {  
 String name = "kaushaik";  
 System.*out*.println(name.charAt(3));  
 System.*out*.println(name.charAt(9));  
 }  
}

Output:

s

Exception in thread "main"

18 .contains()

This metod check the input string ,string is available or not,

If available then return true else return false.

public class Searching {  
 public static void main(String[] args) {  
 String name = "kaushaik";  
 System.*out*.println(name.contains("a"));  
 System.*out*.println(name.contains("y"));  
 }  
}

Output:

True

False

19 .startsWith()

This method is to check the given input start index positon is true then return true else return false.

public class Searching {  
 public static void main(String[] args) {  
 String name = "kaushaik";  
 System.*out*.println(name.startsWith("k"));  
 System.*out*.println(name.startsWith("kau"));  
 System.*out*.println(name.startsWith("a"));  
 }  
}

Output:

True

True

False

20 .endsWith()

This method is to check the given input last index positon is true then return true else return false.

Example:

public class Searching {  
 public static void main(String[] args) {  
 String name = "kaushaik";  
 System.*out*.println(name.endsWith("k"));  
 System.*out*.println(name.endsWith("ik"));  
 System.*out*.println(name.endsWith("ak"));  
 System.*out*.println(name.endsWith(""));  
 }  
}

Output:

true

true

false

true

String Conversion:

21 .touppercase()

This is convert the string to uppercase.

22 .tolowercase()

This is convert the string to lowercase.

public class Conversion {  
 public static void main(String[] args) {  
 String name = "Kaushik";  
 int a = 10;  
 System.*out*.println(name.toLowerCase());  
 System.*out*.println(name.toUpperCase());  
 }  
}

Output:

kaushik

KAUSHIK

23 .valueOf ()

This method is used to convert int to string.

public class Conversion {  
 public static void main(String[] args) {  
 int a = 10;  
 int b = 20;  
 System.*out*.println(a+b);  
 String s1 = String.*valueOf*(a);  
 String s2 = String.*valueOf*(b);  
 System.*out*.println(s1+s2);  
 }  
}

Output:

30

1020

24 .tocharArray()

This method is used to convert string to character array.

public class Conversion {  
 public static void main(String[] args) {  
 String name = "kaushik";  
 char [ ] c = name.toCharArray();  
 System.*out*.println(c);  
 }  
}

Output:

kaushik

String :

String is create a immutable Object.

If data does not change frequently then use String.

Ex.name,email,password

EX. If we concate one new string then old value cannot change and modify ,it create on more object and add new string.

StringBuffer:

StringBuffer is mutable Object.

It is synchronized method,synchornized method performance are down and all method are thread safe due to which performance low.

If data we change frequently then use StringBuffer.

Ex. Notepad,calculator

Ex. . If we concate one new string then old object has modify ,its not to create new object.

StringBuffer default capacity 16.

StringBuffer Constructor:

public class Bufferdemo {  
 public static void main(String[] args) {  
 StringBuffer sb = new StringBuffer(); //Constructor  
 System.*out*.println(sb.capacity());  
 StringBuffer sb2 = new StringBuffer("kaushik"); //Constructor  
 System.*out*.println(sb2.capacity());  
 StringBuffer sb3 = new StringBuffer(1000); //Constructor  
 System.*out*.println(sb3.capacity());  
  
 }  
}

Output:

16

23

1000

StringBuffer Method:

1 .capacity()

This method is provide the capacity to store in object.

Ex. Default capacity =16.

(Old capacity \* 2)+2.

public class Bufferdemo {  
 public static void main(String[] args) {  
 StringBuffer sb = new StringBuffer();  
 System.*out*.println(sb.capacity());  
 sb.append("prasad");  
 System.*out*.println(sb.capacity());  
 sb.append("Kaushik prasad");  
 System.*out*.println(sb.capacity()); //(old cpacity\*2)+2 =34  
  
 }  
}

Output:

16

16

34

2 .length()

It is provide the length of string.

3 .append()

This method is use to add the string in last.

4 .charAt(number)

This method is provide the index position of string.

5 .delete(start index,last index)

This method is used to delete the string from starting to last index.

6 .deletecharAt(number)

This method is use to delete the char from string.

7 .equals()

This method is used check the content of object.

StringBuffer class does not override equals methods of object class and string class override the equala methods of object class.

Ex. public class Bufferdemo {  
 public static void main(String[] args) {  
 StringBuffer sb = new StringBuffer("kaushik");  
 StringBuffer sb1 = new StringBuffer("kaushik");  
 System.*out*.println(sb.equals(sb1));  
 StringBuffer sb3 = sb.append("prasad");  
 System.*out*.println(sb3.equals(sb));  
 }  
}

Output:

False

True

8 .indexOf(character)

This method is provide the index position of the string.

It started from starting index.

9 .lastindexof(character)

This method is provide the index position of the string.

It provide from last index.

10 .insert(number,”provide string”)

This method is used to add the string from given index.

11 .replace(start index, last index,”string”)

This method is used to replace the string from given starting index to last index and put given string.

12 .reverse()

This method is used to reverse the string.

It is only applicable for StringBuffer not for string.

13 .subsequence(start index, last index)

This method is provide the string from given starting to given last index-1.

14 .substring(index)

This method is provide the string from given starting index to last index.

If we provide starting index and last index-1 then also same as subsequence method

15 .ensureCapacity()

This method is provide the capacity from we given capacity.

16 .setcharAt(index,’ character’)

This method is used to set the char from given index position to given char.

17 .setlength(index)

This method is used to set the maximum length of string.

public class Str {  
  
  
 public static void main(String[] args) {  
 StringBuilder sb = new StringBuilder("My name is kaushik");  
 System.*out*.println("Original: " + sb);  
  
 // Set length to 5  
 sb.setLength(5);  
 System.*out*.println("After setLength(5): " + sb);  
  
 // Set length to 10  
 sb.setLength(10);  
 System.*out*.println("After setLength(10): " + sb);  
 }  
  
 }

Output:

Original: My name is kaushik

After setLength(5): My na

After setLength(10): My na

18 .trimtoSize()

This method is used to remove the extra memomry.

Ex.

public class Bufferdemo {  
 public static void main(String[] args) {  
 StringBuffer sb4 = new StringBuffer("kaushik");  
 System.*out*.println(sb4.capacity());  
 sb4.ensureCapacity(100);  
 sb4.append("kaushik");  
 System.*out*.println(sb4.capacity());  
 sb4.trimToSize();  
 System.*out*.println(sb4.capacity());  
   
 }  
}

Output:

23

100

14

StrinBuilder Method:

StringBuilder:

It is also same as StringBuffer

In this method create non-synchronized method to provide better performance.

Difference between String, StringBuffer and StringBuilder.

|  |  |  |  |
| --- | --- | --- | --- |
|  | String | StringBuffer | StringBuilder |
| Storage | Heap Area, String constant pool | Heap Area | Heap Area |
| Object | immutable | mutable | mutable |
| Memory | High memory occupy | Consume Less efficient memory | Consume Less memory |
| Thread-Safe | thread safe | It is synchronized method and thread safe | It is non-synchronized method and not thread safe |
| Performance | High | low. | High |
| Use | If data is not changing frequently | If data change frequently | If data change frequently |
|  |  |  |  |