

6/12/22

Case (4) +



GADDALA ASHISH
in LinkedIn

```

String s1=new String("dhoni");
          ↑ I

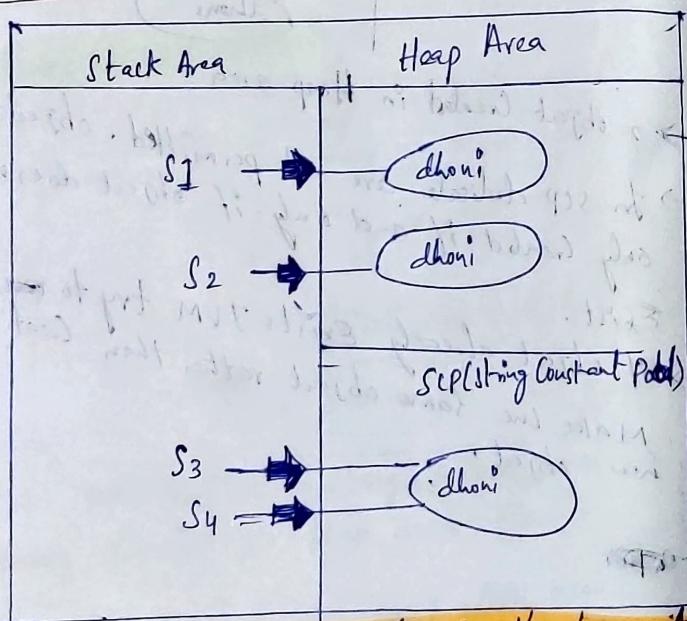
String s2=new String("dhoni");
          ↑ II

System.out.println(s1==s2); //false

String s3=" " dhoni";
          ↑ III

String s4=" dhoni";
          ↑ IV

System.out.println(s3==s4); //true
  
```



Output: Two objects are created in the heap with data as "dhoni" with reference of s1 and s2
→ One object is created in SCP with the reference of s3, s4.

(a) In case of I and II

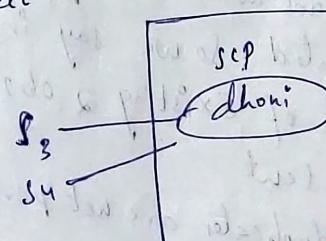
- 2 objects created for s1 and s2 string.
- In SCP, duplicates are not permitted.
- Objects are only created if and only if object does not exist.

→ If object already exists JVM try to make use of same object rather than creating new object.

(b) In case of III and IV

- Object already created in SCP so same object will be reused in Case III and IV.

→ s3 and s4 are created using direct literals.
→ Same object is reshared by s4.



Conclusion:

- ① Duplicates not permitted in SCP.
- ② Duplicates are permitted in Heap Area.
- ③ If objects in heap area not have reference variable then the object get cleared by garbage collector.
- ④ In case of object in SCP which is not having reference variable will not be cleared by garbage collector.

⑤ The memory of object in SCP which is not pointed by reference Variable will be cleared ~~only~~ at the time of JVM shutdown.

⑥ When we stop Execution of program that time memory of ~~not~~ garbage object get cleared.

Note: ① Object creation in SCP is always optional.

② JVM will check is any object already created with required Content or not.

③ if it is not available only then new object will be created. So we say in SCP there is no chance of existing 2 objects with the same content.

④ In SCP duplicates are not permitted.

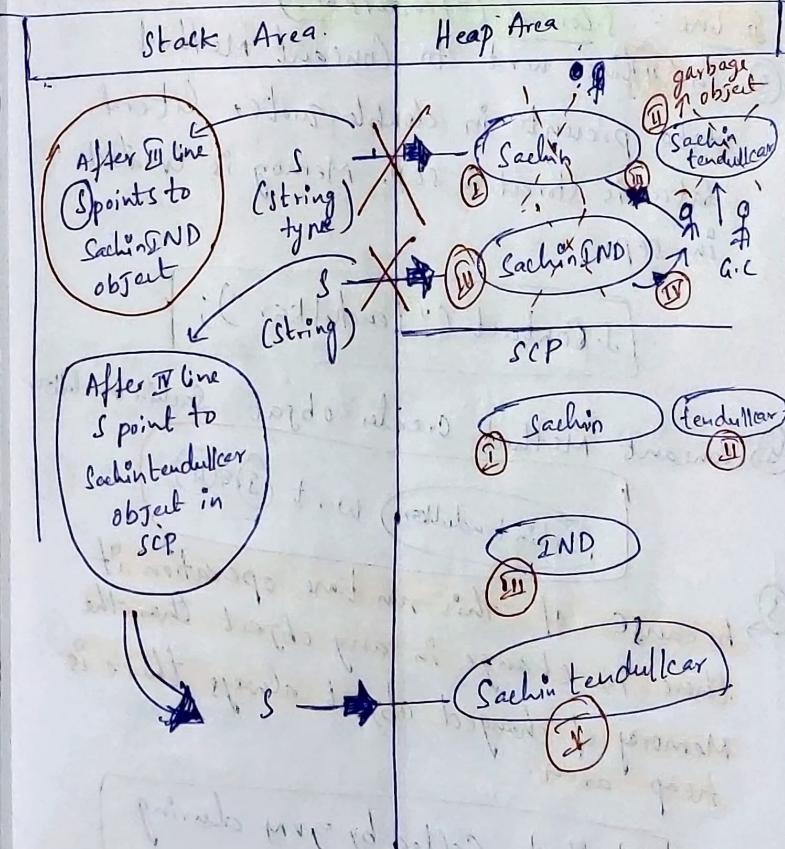
⑤ Garbage Collector Cannot access SCP Area,

Even though object does not have any reference still object is not eligible for garbage collector.

⑥ All SCP objects will be destroyed only at the time of JVM shutdown.

Case ⑤

String s = new String ("Sachin"); → I
 s.concat ("tendulkar"); → II
 s = s.concat ("IND"); → III
 s = "Sachintendulkar"; → IV
 System.out.println(s); → V



GADDALA ASHISH
in LinkedIn

S-line

① for ② line 2 objects are created in heap area and SCP.

② S is string type, if it is immutable, if you want to change a new memory is given.

③ ↳ objects are immutable in string type.

④ ② line : S.Concat ("tendulkar")

⑤ In ④ lines went to Concat Method

data present in double quotes literal data is stored in SCP. Memory is created in SCP.

S.Concat ("tendulkar");

⑥ Concat Method creates object Sachin tendulkar

w.r.t ① & ②

⑦ because of "this run time operation if there is change in any object then the memory of changed object always there is heap area."

→ ↳ Methods called by JVM during run time.

→ this object is garbage object.

→ garbage collector collects this object which is without reference.

⑧ ③ line : S = S.Concat ("IND");

⑨ Memory is given in SCP for literal data

IND in SCP

⑩ here Concat Method is called by JVM during runtime of creates object in Heap area

Sachin IND

⑪ it is collected by reference S.

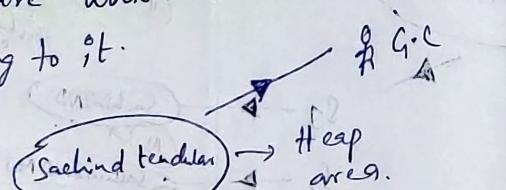
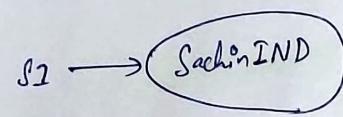
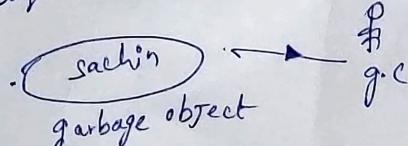
⑫ created object given to S variable.

⑬ S is pointing to object Sachin IND.

S → Sachin IND

⑭ Sachin object in heap area becomes garbage object. garbage collector collects this object.

Sachin → Garbage object
Garbage Collector

- ① Line ① - `String s1 = new String ("Sachin");`
- ② Copy of objects created in heap and SCP.
- ③ s1 is points to object in heap area.
- ④ Line 8 - `s1 = s1.concat ("tendulkar");`
- ⑤ Memory for direct literal is given on ~~heap~~ SCP
- ↳ "tendulkar" object created on SCP.
- ⑥ "Sachin tendulkar" object created in heap area which don't have reference Variable pointing to it.
- 
- ⑦ No reference for object. garbage collector collects this object
- ⑧ Line 13 - `s1 += "IND";`
- ⑨ s1 = s1 + "IND"; (`String + "IND"`)
double acting here
- Addition operator do 2 things
- ① **Addition** (both operands are number type)
 - ② **Concatenation** ↳ if one operand is string then Concatenation is done
- ⑩ Memory for direct literal is given on SCP
- IND → SCP
- ⑪ s1 += "IND"; → is runtime operation.
- ↳ Memory of ~~this~~ is given on heap area
- ⑫ SachinIND object is created in heap area.
- ⑬ s1 Points to SachinIND
- 
- ⑭ Sachin object in heap area is collected by garbage collector.
- 

GADDALA ASHISH
Linkedin

(u) Line - IV String S2 = S1.Concat("MI");

① Memory for direct literal created on SCP

MI → SCP

② Concatenation is done and stored in heap area

S2 → SachinINDMI → heap area.

③ S2 is points to SachinINDMI object.

(3) Line IV and S1 = S.O.P(S1);
S.O.P(S2); //

S1 → SachinIND

S2 → SachinINDMI

→ Prints output on Console.

Eg :-

String S1 = new String ("you Cannot change me!");

String S2 = new String ("you Cannot change me");

System.out.println(S1 == S2); // false

String S3 = "you Cannot change me!";

S.O.P(S1 == S3); // false

String S4 = "you Cannot change me!"

S.O.P(S3 == S4); // true

String S5 = "you Cannot" + "Change Me!";

S.O.P(S3 == S5) // true

String S6 = "you Cannot";

String S7 = S6 + "Change Me!"; (New object is created in heap)

S.O.P(S3 == S7); // false

final String S8 = "you Cannot";

String S9 = S8 + "Change Me!"; you Cannot Change Me!

S.O.P(S3 == S9); // true

S.O.P(S6 == S8); // true

→ final type Variable + Compiler know what is value Where it is required it will use



GADDALA ASHISH
LinkedIn

Stack Area

Heap Area

S1

you cannot change me

S2

you cannot change me

S3

you cannot change me

scf

S4

you cannot change
me!

S5

S9

you cannot

change me!

S6

S8



GADDALA ASHISH

LinkedIn

- Q) Check whether data available in SCP by heap object reference
- String s = new String ("Sachin"); 07/12/22
- how to check data available in SCP using string s1?
- using Method we can check. String is predefined class.
- using heap object reference we can check by Scanning.

Intern Method Example ①

Q) Checks whether data available in SCP :-

String s1 = new String ("Sachin");

~~s1.equals(s2);~~
String s2 = s1.intern();

System.out.println(s1 == s2);

↳ checking data available in SCP. gives reference collected by s2

String s3 = "Sachin";

System.out.println(s2 == s3);

↳ s1 and s2 are not pointing to same object.

↳ because different objects created

↳ //true
↳ intern method doing search in SCP

↳ s2 and s3 referring to same object in SCP.



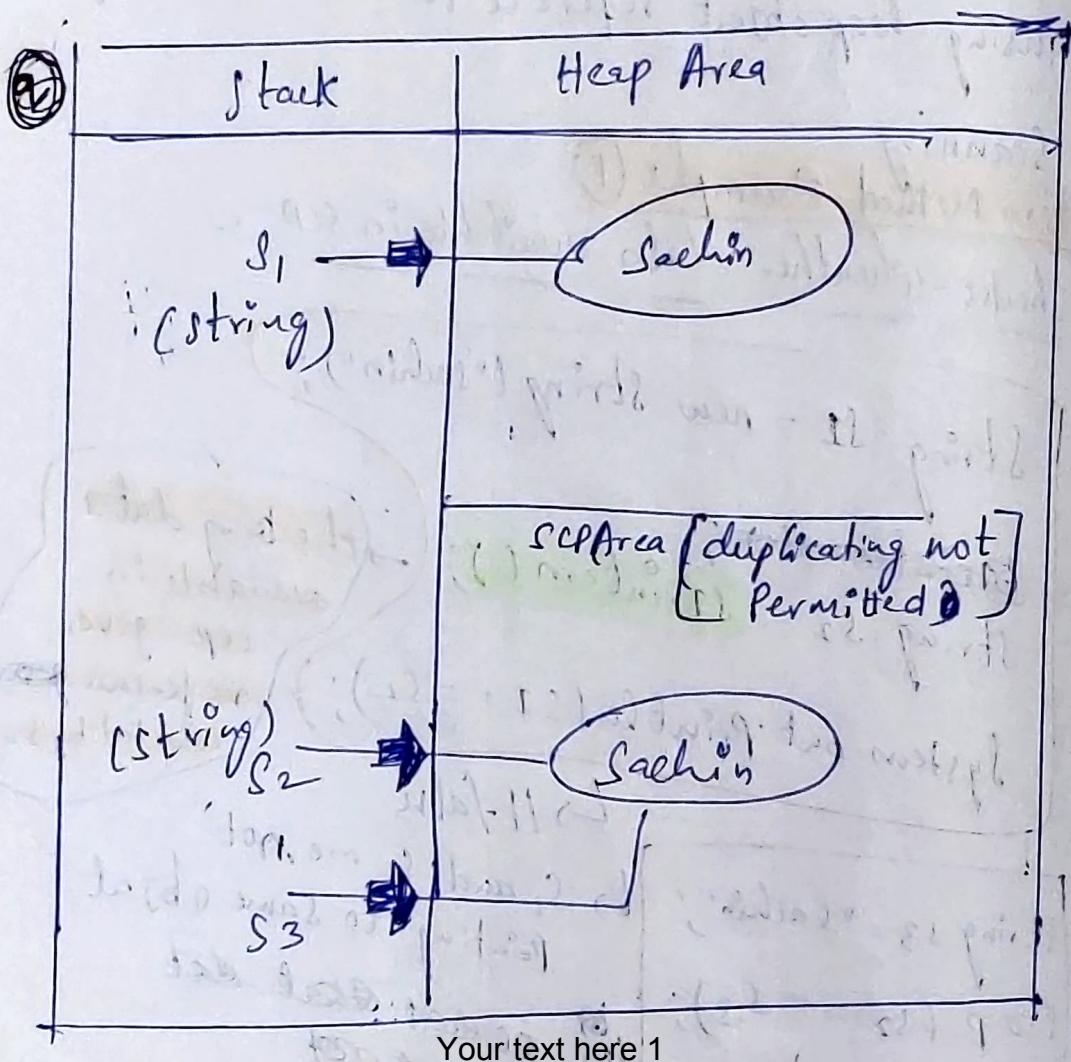
GADDALA ASHISH

in LinkedIn

→ intern method search for ~~objects~~, object data not in heap but in SCP.

→ s_2 and s_3 refers to same object in SCP.

→ it shows data is present in SCP.



GADDALA ASHISH
in LinkedIn

Intern Method

Example ②

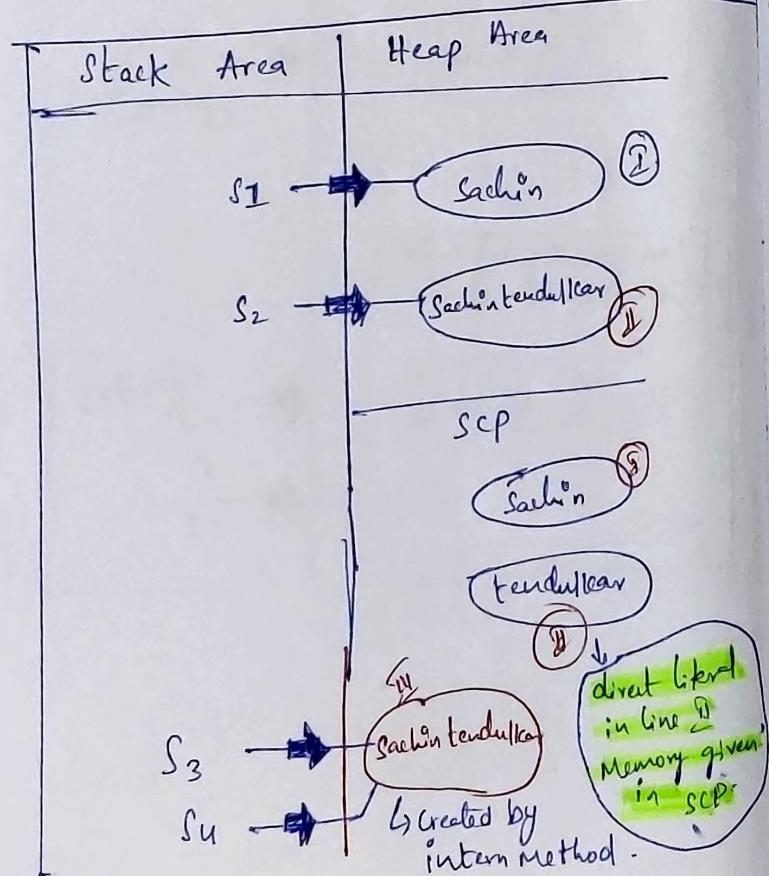
String s1 = new String ("Sachin"); → ①

String s2 = s1.concat (" tendulkar"); → ②

String s3 = s2.intern(); → ③

String s4 = "Sachintendulkar"; → ④

System.out.println (s3 == s4); // true → ⑤



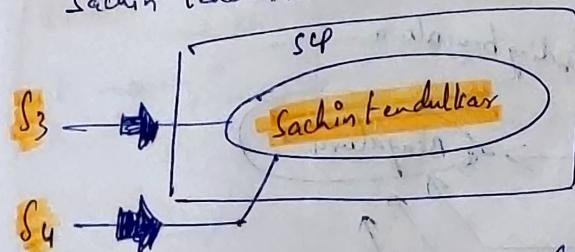
Line-⑤ :-
→ Intern Method goes to SCP and tries to search data Sachintendulkar in SCP.
→ data not available in SCP.

- if data found in SCP Intern Method gives reference
- if data not found in SCP, it creates object

↳ Intern Method

→ here data not found in SCP then Intern method creates object Sachintendulkar in SCP.

→ ~~to ④~~
→ S3 and S4 refers to same object "Sachintendulkar" in SCP



Line ⑤ :- if true then we can say data found in SCP.



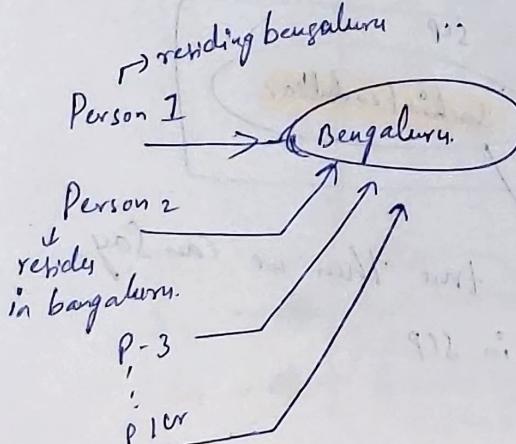
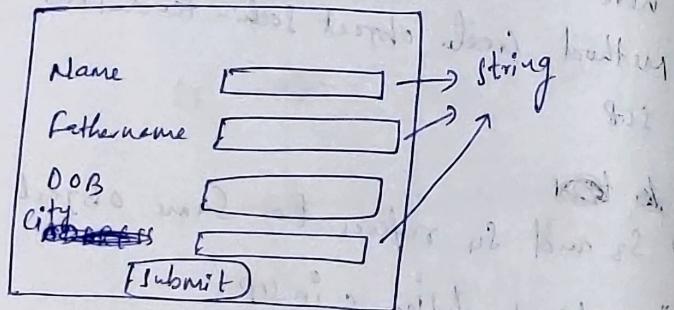
GADDALA ASHISH
in LinkedIn

Discussion:

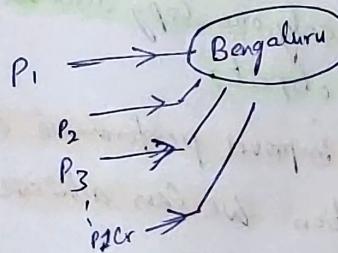
- ① Why concept of SCP is available for strings?
- ② Why concept string objects are classified into 2 types-
 - (a) mutable
 - (b) immutable

Need of SCP :- (Practical Example)

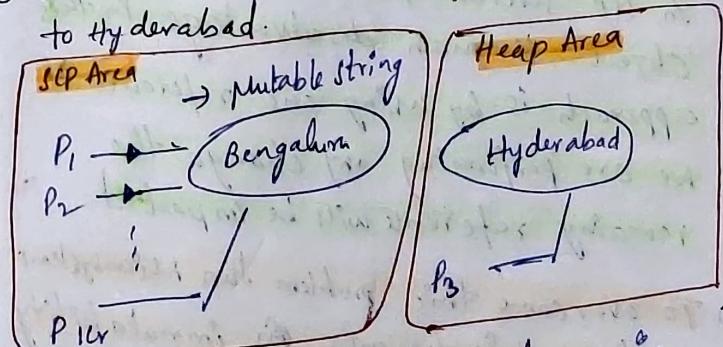
⇒ Building Aadhar Card Application



- ~~multiple~~
- ① one object should be created and multiple references pointed to same object.
- ↳ Consumption Memory is less in heap area.
- ④ one copy pointed by multiple references
- ↳ PSCP Area is given to that object
- eg. ① Persons residing in bangalore.



- ② Person P3 want to change from bengaluru to hyderabad.



- if change required then change is done on new object created for it in heap area.



GADDALA ASHISH
LinkedIn

① importance of SCP :-

- ① In our program if an string object is required to use repeatedly then it is not recommended to create multiple object with same content.
 - ↳ If reduces performance of the system and effects memory utilization.
- ② We can create only one copy and we reuse the same object.
 - ↳ This approach improves performance and memory utilization we can achieve this by using "SCP".
- ③ In SCP several references pointing to same object the main disadvantage in this approach is by using one reference if we are performing any change the remaining reference will be impacted.
 - ↳ To overcome this problem Jan Microsystems developers implements the immutability concept for string objects.

Q According to this once we create a string object we can't perform any change in the existing object. If we are trying to perform any change a new string object will be created hence immutability is the main advantage of SCP.



GADDALA ASHISH
LinkedIn

② String At API Level :

API = Application Programming Interface

↳ Some one wrote the code and he will give .class file, End users will use and take the benefit.

→ String.class → Java Community Provided this class.

→ Entire Java we are learning as API only

→ ~~public static void main(String[] args){~~
System.out.println("Hello");
}

→ We are using String & System class which is written by developers, taking help from someone code.

→ println is Method.

→ We are taking help from someone code.

→ We are learning as API level only

③ Command to see API

java -p java.lang.String

→ it shows all Methods of strings.

Eg: concat()

intern()

equals()

↳ showing profile nature of java command for class called java.lang.String

↳ it shows all Methods of strings
Eg: concat()
intern()

→ it is called API Dissection

↳ what API contains?

(i) Constructors : if a method name is same as that of class name we call it as **Constructor**.

Eg: Class Demo

{
Public Demo(){
}

y



GADDALA ASHISH
in LinkedIn

String Class Constructor :-

① `String s = new String()` \Rightarrow Creates an Empty String object

② `String s = new String(string literal)`

\hookrightarrow Creates an object with string literal on heap.

③ `String s = new String(stringBuffer sb)`

\hookrightarrow Creates an equivalent string object for string Buffer.

④ `String s = new String(char[] ch)`

\hookrightarrow make character Array as String.

\hookrightarrow Creates an equivalent String object for character Array

⑤ `String s = new String(byte[] b)`

\hookrightarrow Creates an equivalent String object for byte array.

Eg①.

`char[] ch = {'J', 'A', 'V', 'A'}`;

`String s1 = new String(ch);`

`System.out.println(s1); // Java`

Eg②

`byte[] b = {65, 66, 67, 68};` \rightarrow Unicode value are taken for this number

`String s = new String(b);`

`s.o.p();` \rightarrow ABCD

Important Methods of Strings :-

① `public int charAt(int index)`:

② `public String concat(String str)`;

③ `public boolean equals(Object o)`.

④ `public String substring(int begin)`

⑤ `public String substring(int begin, int end)`.

⑥ `public String equalsIgnoreCase(String s)`

⑦ `public int length()`

⑧ `" string replace (char old, char new)"`

⑨ `" String toLowerCase()`

⑩ `" String toUpperCase()`

⑪ `" String trim()`

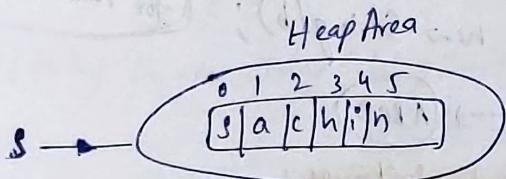
⑫ `public int indexOf(char ch)`

⑬ `public int lastIndexOf(char ch)`

(14) Using String Methods

String s = new String ("Sachin");

→ In Java string is a object. internally data will stored in the form of array, it is Character Array. Collection of character is string.



(i) charAt() → Method

String s = new String ("Sachin");

System.out.println(s[2]) // Compile time error.

System.out.println(s.charAt(3)); ↳ Java gave string as object not as array.

System.out.println(s.charAt(-1));
s.o.p(s.charAt(500)); // exception which

→ In order to access data by index which is stored in string but given as object by using methods.

String Index out of Bounds Exception

Eg String s = new String ('Sachin');
s.o.p (s.charAt(-1));
s.o.p (s.charAt(500));

Length of String :

String s = "Sachin";

System.out.println(s.length()); // 6

↳ Method

int [] arr = {10, 20, 30};

System.out.println(arr.length);

↳ Variable(s)
Property

Note

① Method is for object.

② Property is Variable of a class.

Integer Array class
(proxy class)

Class String ↳ Method name
{ int length();

Public ↓
↓

Class ↳

int length();

Variable
or
Property
name

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y

Y