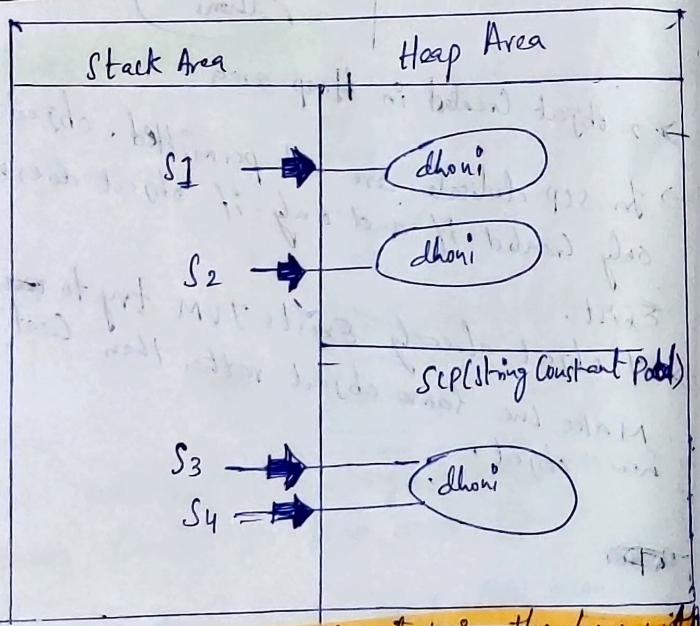
16/12/22

Case (4):-

```
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 as s1 and s2

→ One object is created in SCP with the reference as s3, s4.

(a) In case of I and II

→ 2 object Created for s_1 and s_2 String.

→ In SCP duplicates are not permitted.

→ objects only Created if and only if object does not Exist.

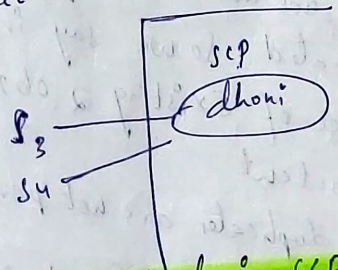
→ if object already Exists JVM try to Make use of same object rather than creating new object.

(b) In case of III and IV

→ object already Created in SCP so same object will be reused in Case III and IV

→ s_3 and s_4 are Created using direct literals

→ Same object is reshared by s_4.



Conclusions:-

(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 garabage 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 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.

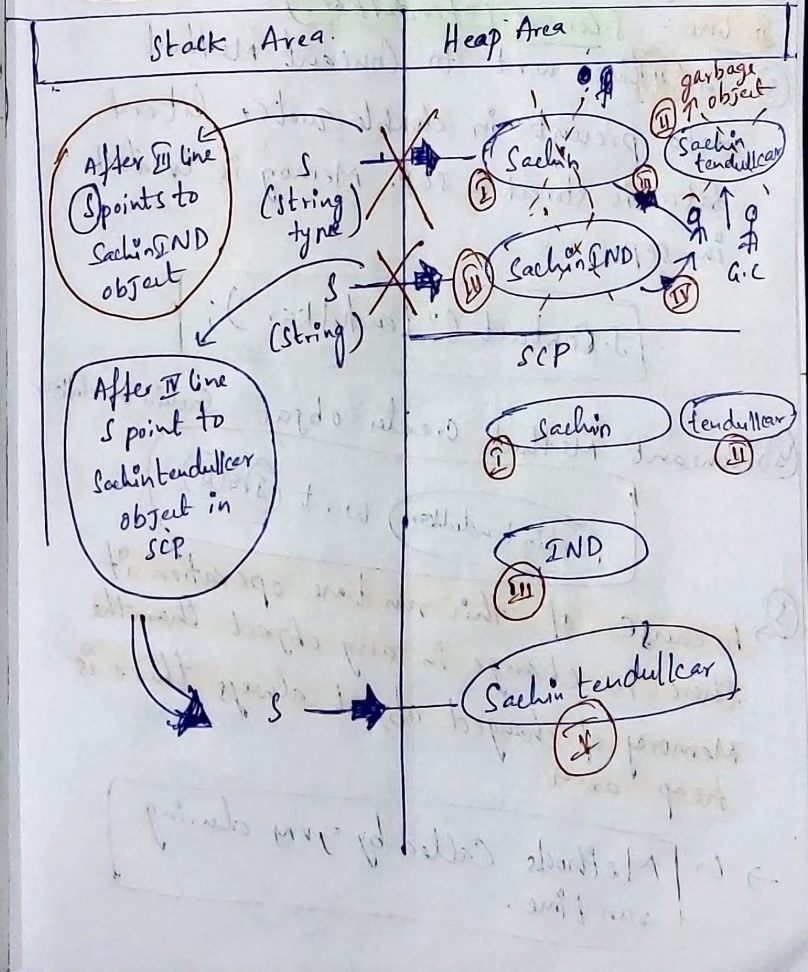
④ in SCP duplicates are not permitted.

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 shut-down.

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
```



S - line

① for (I) line 2 no objects are created in heap area and SCP. (copy of)

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

↳ objects are immutable in string type.

II line : S.Concat ("tendulkar")

③ In (II) line wrt to Concant Method data present in double quote literal data is stored in SCP. Memory is created in SCP.

S.Contact ("tendulkar");

④ Concant Method creates objet Sachin tendulkar

Sachin tendulkar. w.r.t (I) & (II)

⑤ 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 garabage object.

→ garbage collector collects this object which is without refere.

III line : S = S.Concat ("IND");

① Memory is given in SCP for literal data

IND in SCP

② here Concat Method is called by JVM during run time of Creates object in Heap area

SachinIND

③ it is colledcted by refereen S.

④ Created object given to S variable.

⑤ S is pointing to object SachinIND

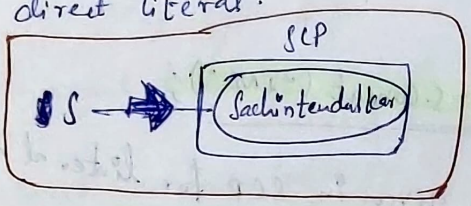
S → SachinIND

⑥ Sachin object in heap area becomes garbage object. garbage collector collects this object.

Sachin → Garbage object. Garbage collector

(IV) line: s = "Sachintendulkar";

① Memory for this is given in SCP. if is direct literal.



② Now s points to Sachintendulkar object in SCP. then SachinIND object becomes garbage object and it is Collected by garbage Collector.

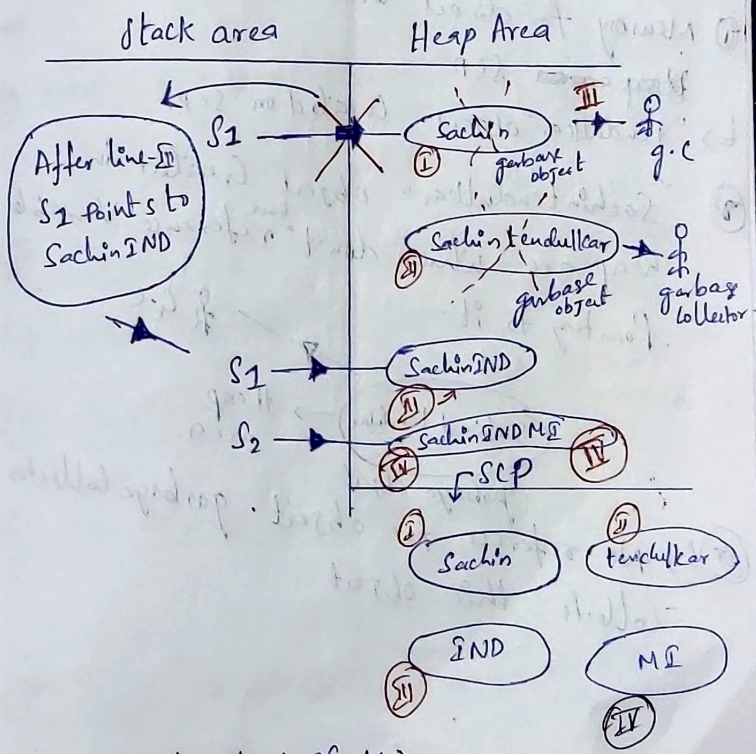
(V) line : System.out.println(s);

① Saching tendulkar is printed on Console

Output :- ① Direct literals are always placed in SCP. Because of runtime operation if object is required to Create Compulsorily that object should ~~do~~ be placed on the heap, but not on scp

Case ⑥ :-

```
String s1 = new String("sachin");          → (I)
s1.Concat("tendulkar");                    → (II)
s1 += "IND";                               → (III)
String s2 = s1.Concat("MI");               → (IV)
System.out.println(s1); // sachin IND      → (V)
System.out.println(s2); // sachin INDMI    → (VI)
```



→ total objects : 8(4+4)

→ Eligible for gc = 2

1) line-1 → String S1 = new String ("Sachin");

① 2 Copy of objects Created in heap and scp.

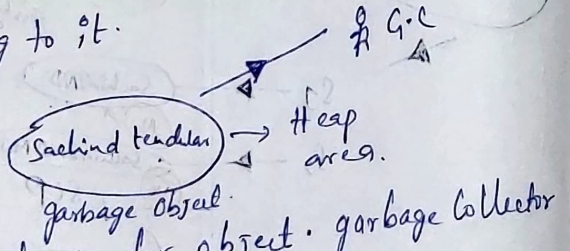
② S1 is points to object in heap area.

2) line-2 → S1.Concat ("tendulkar");

① Memory for direct literal is given on ~~heap area~~ SCP

↳ "tendulkar" object Created on SCP.

② "Sachin tendulkar" object Created in heap are which don't have reference Variable Pointing to it.



③ ↳ No reference for object. garbage Collector Collects this object

3) line-3 → S1 += "IND";

S1 = S1 + "IND"; (String + "IND")

↓ double acting hero

→ Addition operator do 2 things

(a) Addition (both operands are number type)

(b) Concatenation

↳ if one operand is String then Concatination is done

① Memory for direct literal is given on scp

IND → SCP

② S1 += "IND"; → is run time operation.

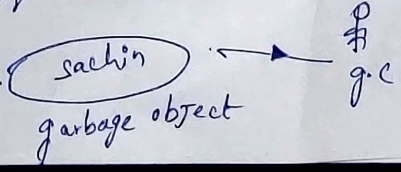
↳ Memory of this given on heap area

③ SachinIND object is Created in heap area.

④ S1 Points to SachinIND

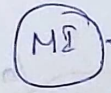
S1 → SachinIND

⑤ Sachin object in heap area is collected by garbage collector.

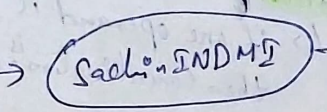


④ Line - IV String S2 = S1.Concat("MI");

① Memory for direct literal created on SCP



② Concatenation is done and stored in heap area



③ S2 is points to SachinINDMI object.

⑤ Line IV and V :- S.o.p(S1);
S.o.p(S2);

S1 → SachinIND
S2 → SachinINDMI

→ Prints output on Console.

Eg :-

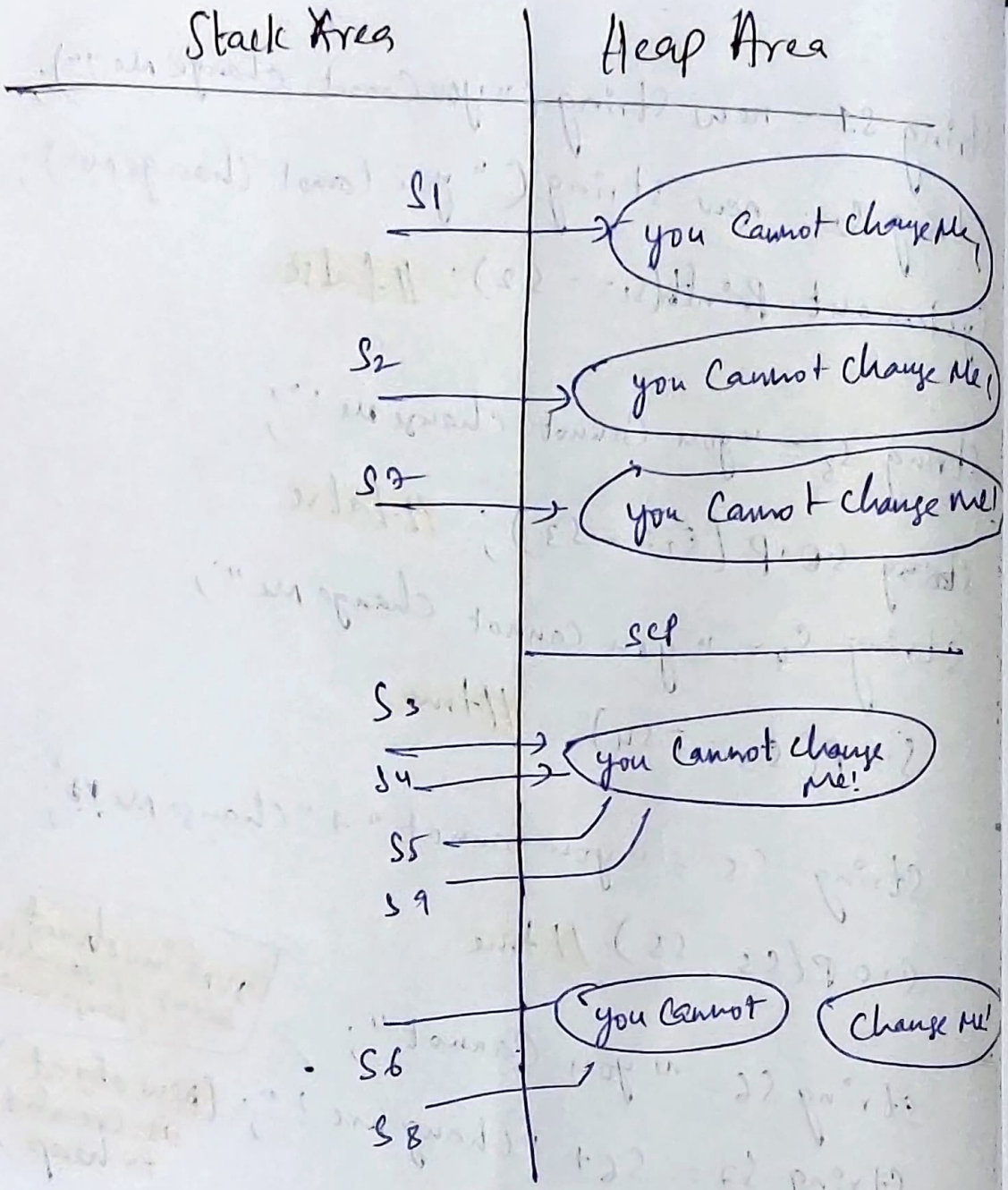
```
String S1 = new String ("you Cannot change Me!");
String S2 = new String ("you Cannot changeMe");
System.out.println(S1==S2); // false

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+ "chang Me!";
S.o.p(S3==S7); // false

final string S8 = "you Cannot";
string S9 = S8 + "chang me!";
S.o.p (S3==S9); // true
S.o.p ( S6==S8); // true
```

JVM involvement because of variable (heap area)

(New object is Created in heap)

final: Compiler involvement

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


Stack Area
Heap Area
S1
you Cannot change Me!
S2
you Cannot change Me!
S7
you Cannot change Me!
scp
S3
S4
you Cannot change Me!
S5
S9
you Cannot
Change Me!
S6
S8

⑧ Check wheather data available in SCP by heap object reference

→ String s = new String ("sachin"); 07/12/23

→ 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

@ Checks wheather data available in SCP :-

```
String s1 = new String ("sachin");
String s2 = s1.intern();
System.out.println(s1 == s2);
```

→ checking data available in scp. gives reference collected by s2

↳ // false

↳ s_1 and s_2 are not pointing to same object.

```
String s3 = "sachin";
s.o.p (s2 == s3);
```

↳ [redacted]

→ // true

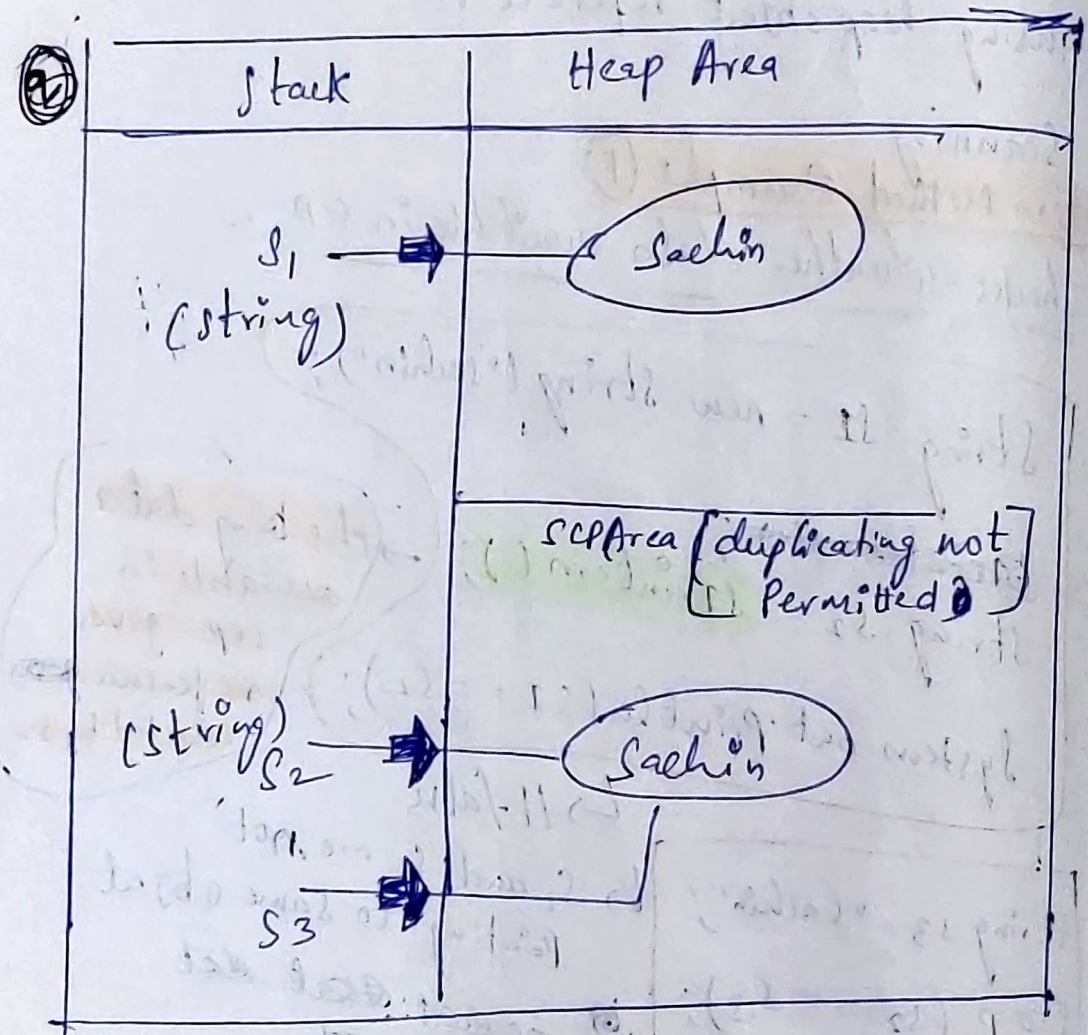
↳ intern method doing search in SCP

↳ s_2 and s_3 refering to same object in SCP.

→ intern Method Search for ~~object~~ S1 object data not in heap but in scp.

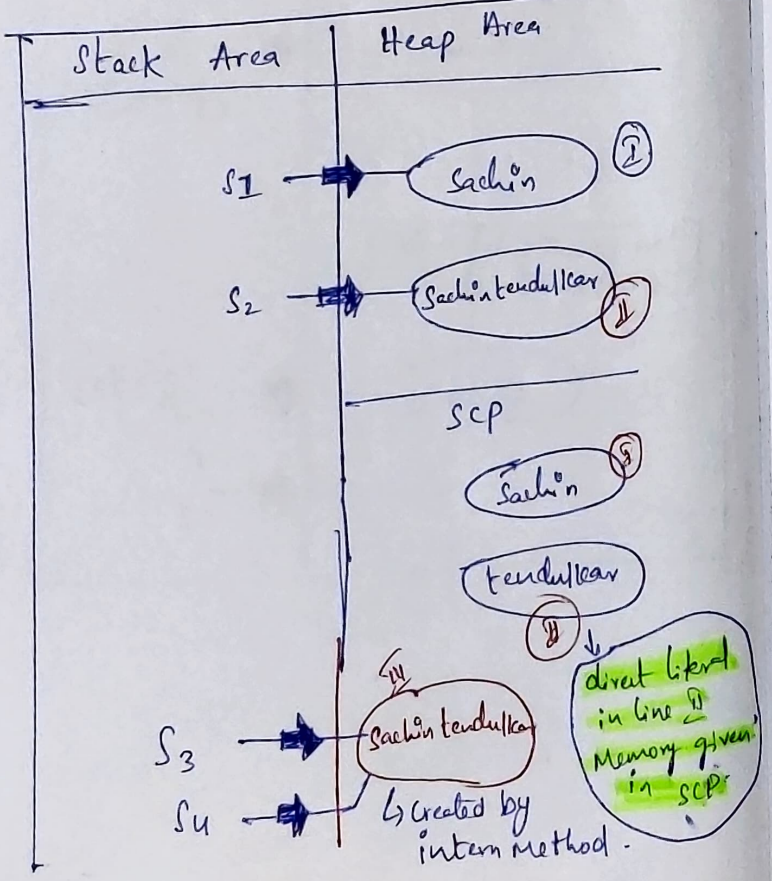
→ S2 and S3 refers to same object in SCP.

→ it shows data is present in SCP.



Intern Method Example (2)

```
String s1 = new String("Sachin");          -- I
String s2 = s1.Concat("tendulkar");        -- II
String s3 = s2.intern();                   -- III
String s4 = "Sachintendulkar";             -- IV
System.out.println(s3==s4); //true         -- V
```



Line - III :-

→ 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

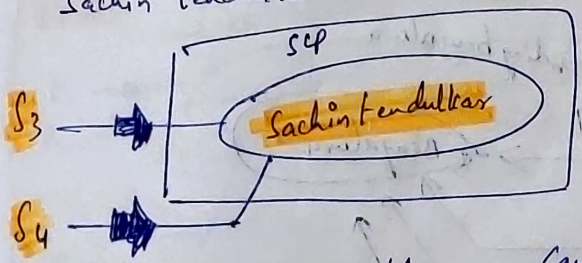
→ if data not found in scp, it Creates object

↳ intern Method

→ here data not found in scp then intern Method create object Sachin tendulkar in SCP.

→ [redacted]

→ s3 and s4 refers to same object "Sachin tendulkar" in scp



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

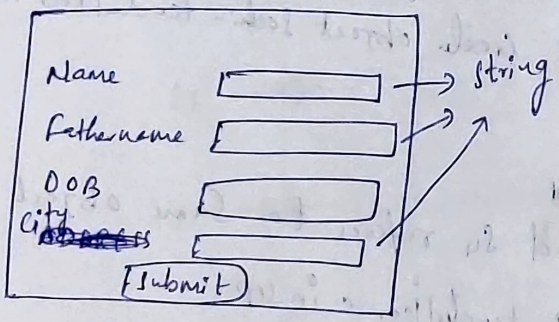
Discussion:

① Why Concept of SCP is available for Strings?

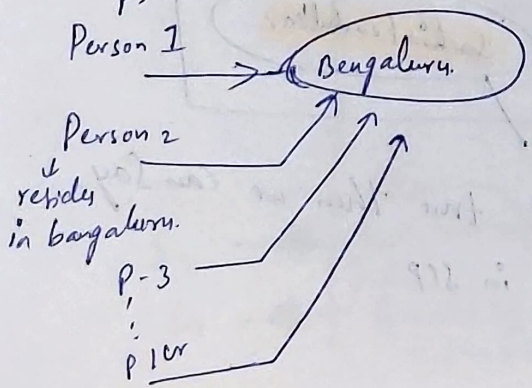
② Why Concept string objects are classified into 2 types.
- (a) Mutable
- (b) immutable

Need of SCP :- (Practical Example)

⇒ Building Adhar Card Application



↳ residing bengaluru ...



→ ~~multiple~~

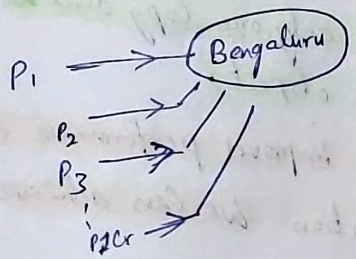
① one object should be Created and multiple reference pointed to same object

↳ Consuption Memory is less in heap area.

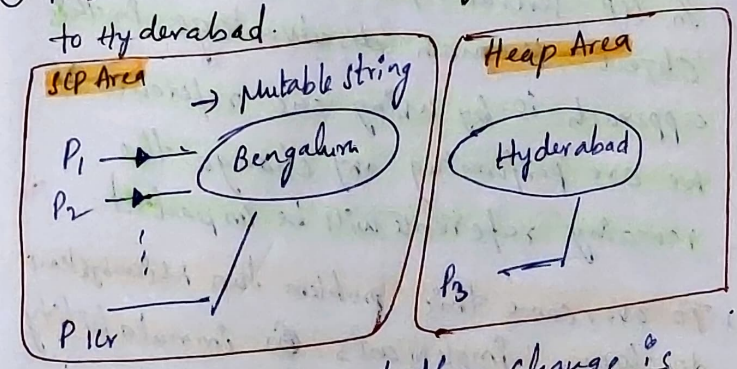
↳ one Copy pointed by multiple references

↳ SCP Area is given to that object

eg: ① Persons residing in bengaluru



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.

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.

↳ it 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 remaing reference will be impacted.

↳ To overcome this problem Sun Microsystems developers implements immutability concept for string objects.

④ 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.

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 benfit.

→ 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

① Command to see API

javap java.lang.String → showing Profile nature of java Command for class called java.lang.String

→ it shows all Methods of strings

Eg: Concat ( )
intern ( )
Equals ( )

↳ it shows all Methods of strings

Eg: Concat ( )
inter ( )

→ it is called API Disection
↳ 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(){

    }
}
```

String Class Constructor :-

① String s = new String() ⇒ Creates an Empty String object

② String s = new String(String Literal)
↳ Creates an object with String literal on heap.

③ String s = new String(StringBuffer sb)
↳ Creates an Equivalent String object for String Buffer.

④ String s = new String(char[] ch)
↳ make character Array as String.
↳ Creates an Equivalent String object for character Array

⑤ String s = new String(byte[] b)
↳ 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};
String s = new String(b);
s.o.p(s) → ABCD
```
→ Unicode value are taken for this number

③ Important Methods of Strings :-

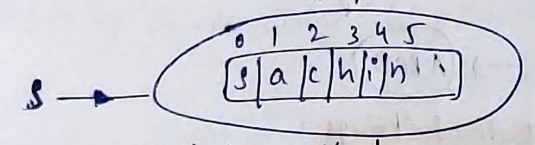
① Public char CharAt(int index);
② Public String Concat(String str);
③ public Boolean Equals(object o)
④ Public String Substring(int begin)
⑤ Public String Substring(int begin, int End);
⑥ Public Boolean EqualsIgnoreCase(String s)
⑦ Public Int length()
⑧ " String replace(Char old, char new)
⑨ " String toLowerCase()
⑩ " String toUpperCase()
⑪ " String trim()
⑫ Public int indexof(char ch)
⑬ Public int Lastindexof(char ch)

(2.4)

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)); → h
System.out.Println(s.charAt(-1));
s.o.p(s.charAt(500)); // Exception
```

→ Java gave string as object not as array.

→ 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(or) Property

Note:
① Method is for object.
② Property is Variable of a class.