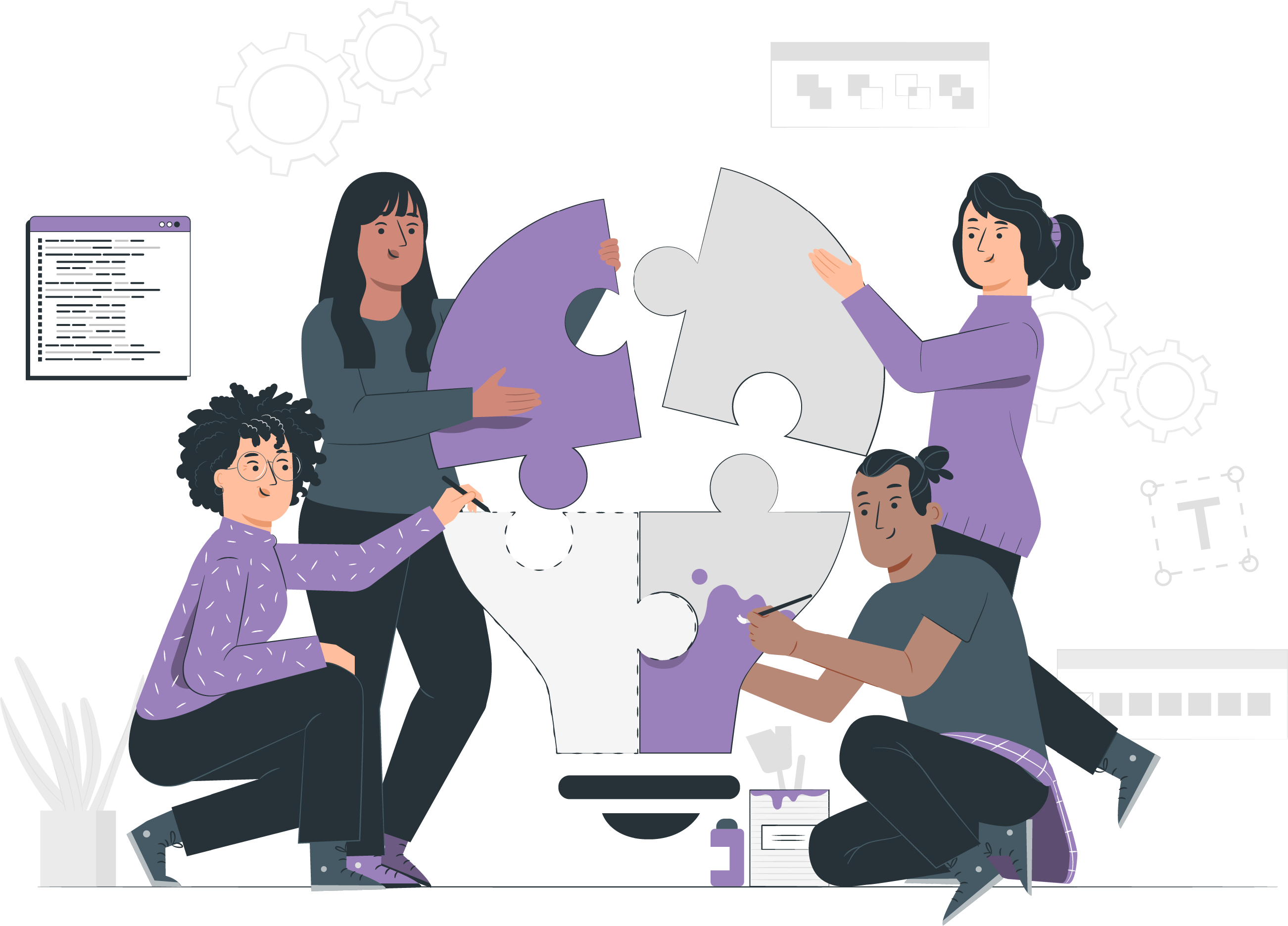
**1.What is a String in Java?**



=>String it refers to an Object in java present in package called java.lang.String. String refers to collection of characters.

***Example***   
String s= "sachin"; System.out.println(s);//sachin

String s =new String("sachin");

System.out.println(s);//sachin

In java String object is by default immutable, meaning once the object is created we cannot change the value of the object, if we try to change then those changes will be reflected on the new object not on the existing object.

**2. Types of String in Java are?**

=> In java Strings are classified into 2 type?

* **Mutable String**
* **Immutable String**

**Mutable String**   
Once if we create a String, on that String if we try to perform any operation and if those changes get reflected in the same object then such strings are called “Mutable String”.

**Example:** StringBuffer, StringBuilder

public class Mutable {  
 public static void main(String[] args) {  
 StringBuilder name1=new StringBuilder("Dipayan");  
 name1.append(" Rana");  
 System.*out*.println(name1);//Dipayan Rana  
 }  
}

**Immutable String**   
Once if we create a String, on that String if we try to perform any operation then those changes won’t be reflected in the same object, rather a new object will be created. Such type of String is called as “Immutable String”.

**Example:** String

public class Mutable {  
 public static void main(String[] args) {  
 String name1=new StringBuilder("Dipayan");  
 name1.concat(" Rana");  
 System.*out*.println(name1);//Dipayan  
 }  
}

**3. In java how many ways can you create string objects in Java?**

=> There are two ways to create String object:

1. **By string literal**
2. **By new keyword**

**1) String Literal**

Java String literal is created by using double quotes. For Example:

String s="welcome";

Each time you create a string literal, the JVM checks the "string constant pool" first. If the string already exists in the pool, a reference to the pooled instance is returned. If the string doesn't exist in the pool, a new string instance is created and placed in the pool. For **example:**

String s1="Welcome";

String s2="Welcome";//It doesn't create a new instance



In the above example, only one object will be created. Firstly, JVM will not find any string object with the value "Welcome" in string constant pool that is why it will create a new object. After that it will find the string with the value "Welcome" in the pool, it will not create a new object but will return the reference to the same instance.

**Note:** String objects are stored in a special memory area known as the "string constant pool".

**2) By new keyword**

String s=**new** String("Welcome");//creates two objects and one reference variable

In such case, [JVM](https://www.javatpoint.com/jvm-java-virtual-machine) will create a new string object in normal (non-pool) heap memory, and the literal "Welcome" will be placed in the string constant pool. The variable s will refer to the object in a heap (non-pool).

**Java String Example**

**public** **class** StringExample{

**public** **static** **void** main(String args[]){

String s1="java";//creating string by Java string literal

**char** ch[]={'s','t','r','i','n','g','s'};

String s2=**new** String(ch);//converting char array to string

String s3=**new** String("example");//creating Java string by new keyword

System.out.println(s1);

System.out.println(s2);

System.out.println(s3);

}

}

**Output:**

java

strings

example

**4.What is a string constant pool?**

=> A string constant pool represents a special heap memory part which is used to store string constants or string literals.

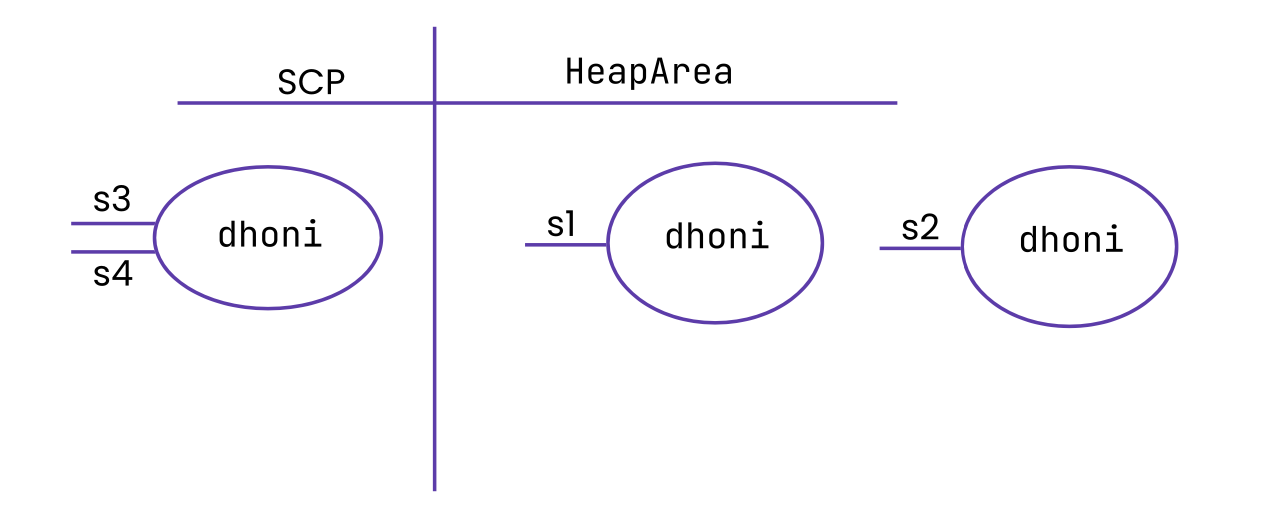
JVM checks the string constant pool every time whenever a string literal is created. JVM will create a new object if string literal is not present in the string pool but no new object will be created in the pool if string literal is already present in the pool, a reference of the already existing object will be returned.

***Note***

* Object creation in SCP is always optional,1st JVM will check if any object already created with required content or not.
* If it is already available then it will reuse the existing object instead of creating the new Object.
* If it is not available only then a new object will be created, so we say in SCP there is no chance of existing 2 objects with the same content.
* In SCP duplicates are not permitted.
* Garbage Collector cannot access SCP Area, Even though Object does not have any reference still object is not eligible for GCW
* All SCP objects will be destroyed only at the time of JVM ShutDown .

**Example**   
String s1=new String("dhoni"); String s2=new String("dhoni"); String s3="dhoni";   
String s4="dhoni";

**Output**   
Two objects are created in the heap with data as "dhoni" with reference as S1,S2. One object is created in SCP with the reference as S3,S4.



**Importance of SCP**

* In our program if any String object is required to use repeatedly then it is not recommended to create multiple objects with same content it reduces performance of the system and affects memory utilisation.
* We can create only one copy and we can reuse the same object for every requirement. This approach improves performance and memory utilisation. We can achieve this by using "scp".
* In SCP several references pointing to the same object the main disadvantage in this approach is by using one reference if we are performing any change the remaining references will be impacted. To overcome this problem sun people implemented immutability concept for String objects.
* According to this once we create a String object we can't perform any changes in the existing object if we are trying to perform any changes with those changes a new String object will be created hence immutability is the main disadvantage of scp.

**5.What do you mean by mutable and immutable objects?**

* The mutable objects can be changed to any value or state without adding a new object. Whereas, the immutable objects can not be changed to its value or state once it is created. In the case of immutable objects, whenever we change the state of the object, a new object will be created.
* Mutable objects provide a method to change the content of the object. Comparatively, the immutable objects do not provide any method to change the values.
* The mutable objects support the setters and getters both. Comparatively, the immutable objects support only setters, not getters.
* The Mutable objects are may or may not be thread-safe, but the immutable objects are thread-safe by default.
* The mutable class examples are StringBuffer, Java.util.Date, StringBuilder, etc. Whereas the immutable objects are legacy classes, wrapper classes, String class, etc.

***Example***

public class Demo\_mutable\_immutable {  
 public static void main(String[] args) {  
 String name1="Dipayan ";  
 name1.concat("Rana");  
 System.*out*.println(name1);*//Output:Dipayan* StringBuilder name2=new StringBuilder("Dipayan");  
 name2.append(" Rana");  
 System.*out*.println(name2);*//Output:Dipayan Rana* }  
}

**6.Where exactly is the string constant pool located in the memory?**

=> A string constant pool is a separate place in the heap memory where the values of all the strings literal which are defined in the program are stored.

