* **String**

1. **How to create immutable class in java? IMP**

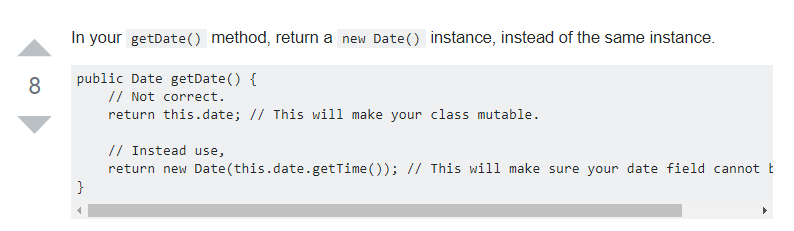
To create immutable class in java, you must do following steps.

1. Declare the class as final so that it can’t be extended, and its behaviour should not be overridden.
2. Make all fields private so that direct access is not allowed and final so that no one will change its value once initialized.
3. Initialize all the fields via a constructor performing **deep copy**.
4. Provide **only getter** methods for variables. Do not provide setters.
5. When getter method is called do not return actual reference of the object instead return copy of it.

Example: immutable. ImmutbleExample.java is there in String project.

1. **How to make immutable object, if the class containing date object which can be modified even the containing class is immutable? IMP**

In getDate() method, return a new Date() instance, instead of the same instance.



1. **What is the meaning of immutable in terms of String? IMP**

Once string object has been created, its value can't be changed.

The simple meaning of immutable is unmodifiable or unchangeable.

class Testimmutablestring{

public static void main(String args[]){

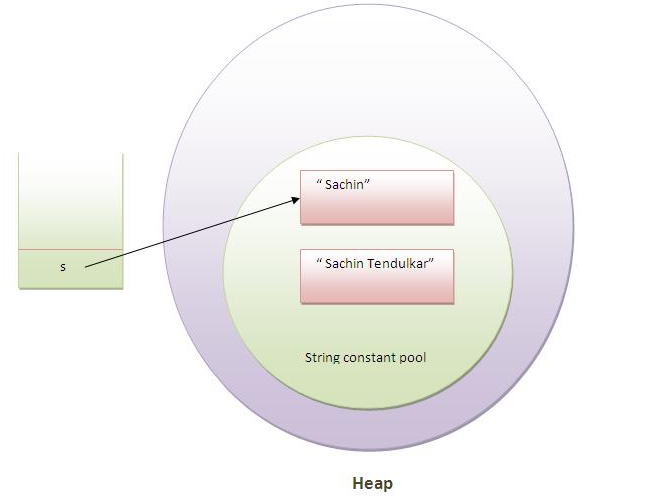
String s="Sachin";

s.concat(" Tendulkar");//concat() method appends the string at the end

System.out.println(s);//will print Sachin because strings are immutable objects

}

}



Example of immutable classes String and all wrapper classes.

1. **Why string objects are immutable in java? Why String is made final in Java? IMP**
2. **String pool**: String is declared as final so that it can be immutable

Because java uses the concept of string literal and String pool concept.

Java has special implantation called String pool.

Say we are creating the string CHANDRA 10 times in our project but only one object will be created in the String pool.

1. **Hash code**: As string is immutable its hash code will be same always and it can cache the hashcode. It is very helpful when we use hash map because String is famous key in hash code.
2. **Securit**y: String is made final so that its behaviour should not be overridden by anyone. This way we get security.

For example: For opening network connection we can pass IP and port in the form of String, For opening database connection we can pass database URL in the form of String and to read any file we can pass file path in the form of string . In all these cases if string is not immutable then there will be a serious security issue.

1. **Thread-safe**: We no need to synchronize the string operations externally.
2. **How many ways we can create the string object? IMP**

There are two ways to create the string object, by string literal and by new keyword

String s="javatpoint";

String s=new String ("javatpoint");

1. **How many objects will be created in the following code? IMP If possible get more samples on how many objects are created?**

String s1="Welcome";

String s2="Welcome";

String s3="Welcome";

Only one object. These are created in String constant pool.

1. **How many objects will be created in the following code? IMP**

String s = new String("Welcome");

Two objects, one in string constant pool and other in non-pool (heap).

1. **In String new or “” which one will you prefer and why? IMP**

I prefer “”.

When we create String using “” (double quote) only one object will be created in String constant pool.

But if we use new key word to create string 2 objects will be created one in String constant pool and another one in heap.

1. **In String how equals works? IMP**

Equals method in String class checks if the two strings have same character sequence or not.

The method is implemented as below.

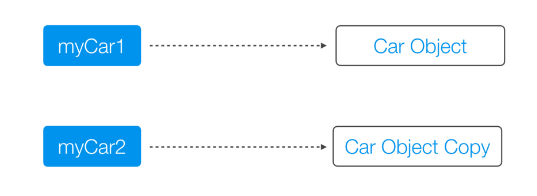
* 1. If the String Objects are equals as per == operator, true is returned. else
  2. Method checks if input object is instance of String .if input object is not instance of String then false will be returned.else
  3. The length of the input Sting and this string will be compared if both are not equals then false will be returned else
  4. Each character of both the strings are compared if all character sequence are same then true will be returned else false will be returned.

1. **public** **boolean** equals(Object anObject) {
2. **if** (**this** == anObject) {
3. **return** **true**;
4. }
5. **if** (anObject **instanceof** String) {
6. String anotherString = (String)anObject;
7. **int** n = value.length;
8. **if** (n == anotherString.value.length) {
9. **char** v1[] = value;
10. **char** v2[] = anotherString.value;
11. **int** i = 0;
12. **while** (n-- != 0) {
13. **if** (v1[i] != v2[i])
14. **return** **false**;
15. i++;
16. }
17. **return** **true**;
18. }
19. }
20. **return** **false**;
21. }
22. **What is shallow and deep copy?**

**Shallow copy**: Creates copy of t he reference variable.



**Deep Copy**: An object copy creates a copy of the object itself.



1. **Why java uses the concept of string literal?**

To make Java more memory efficient (because no new objects are created if it exists already in string constant pool).

1. **What is the basic difference between string and stringbuffer object? IMP**

String is an immutable object. StringBuffer is a mutable object.

1. **What is the difference between StringBuffer and StringBuilder ? IMP**

StringBuffer is synchronized whereas StringBuilder is not synchronized.

1. **How to check if a String and String Buffer objects are same or not? IMP**

The answer is not equals because StringBuffer class does not override equals.

Using String.contentEquals(StringBuffer);

In this case left operand in String and Right Operand is String buffer. May be if left operand is StringBuffer then we need to call StringBuffer.toString.contentEquals(StringBuffer)

1. **How to check if 2 objects are equals or not by value? IMP**

We can check if 2 objects are equals or not we need to override equals method and provide correct implementation to check if both objects are equals or not.

Suppose if we have circle object and we want to find if 2 circles are equal then logic will be if radius of 2 circle are same then 2 circles are equal. Below is implementation of equal method.

Public Boolean equals(Object o)

{

If (o instanceOf Circle)

{

Circle c = (Circle) o;

getRadius ()= c.getRadius();

}

return false;

}

1. **What is the purpose of toString() method in java ?**

The toString() method returns the string representation of any object. If you print any object, java compiler internally invokes the toString() method on the object. So overriding the toString() method, returns the desired output, it can be the state of an object etc. depends on your implementation.

1. String pool details? Search more questions otherwise whatever you know is fine
2. **Why atomic Integer and what is its usage? IMP**

Generally atomic operations are necessity in multithreaded environment to avoid data inconsistency.

So we can use atomic integer when we do not want shared integer be accessed by multiple threads at same time.

1. **Why atomic integer is required when volatile keyword is already there? IMP**

If you declare primitive integer as volatile then individual read and write operation on that variable is atomic.

Suppose if you do i++ then it is equivalent to i = i+1 in this case it requires 2 operation one read and one write. Here there is chance that value of i can be changed by another thread before one thread read and write.

This is the reason almost all primitives have atomic integers.