**Java String Interview Questions and Answers**

**Increment and Decrement Question Link**

**//** Must Read

**http://www.instanceofjava.com/2015/07/increment-decrement-operators-interview.html**

# [The substring() Method in JDK 6 and JDK 7](https://www.programcreek.com/2013/09/the-substring-method-in-jdk-6-and-jdk-7/)

1. What substring() does?

The substring(int beginIndex, int endIndex) method returns a string that starts with beginIndex and ends with endIndex-1.

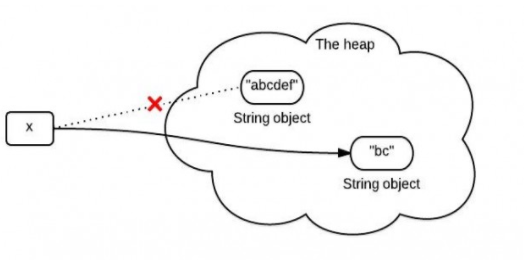
|  |
| --- |
| String x = "abcdef";  x = x.substring(1,3);  System.out.println(x); |

Output:

bc

2. What happens when substring() is called?

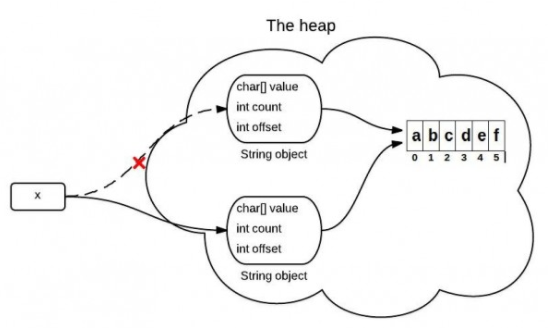
You may know that because x is immutable, when x is assigned with the result of x.substring(1,3), it points to a new string like the following:



However, this diagram is not exactly right. What exactly happens when substring() is called is different between JDK 6 and JDK 7.

3. substring() in JDK 6

String is supported by a char array in the back-end. In JDK 6, the String class contains 3 fields: char value[], int offset, int count. They are used to store real character array, the first index of the array, the number of characters in the String.



When the substring() method is called, it creates a new string, but the string's value still points to the same array in the heap. The difference between the two Strings is their count and offset values.

The following code is simplified and only contains the key point for explain this problem.

|  |
| --- |
| *//JDK 6*  String(**int** offset, **int** count, **char** value[]) {  **this**.value = value;  **this**.offset = offset;  **this**.count = count;  }    **public** String substring(**int** beginIndex, **int** endIndex) {  *//check boundary*  **return** **new** String(offset + beginIndex, endIndex - beginIndex, value);  } |

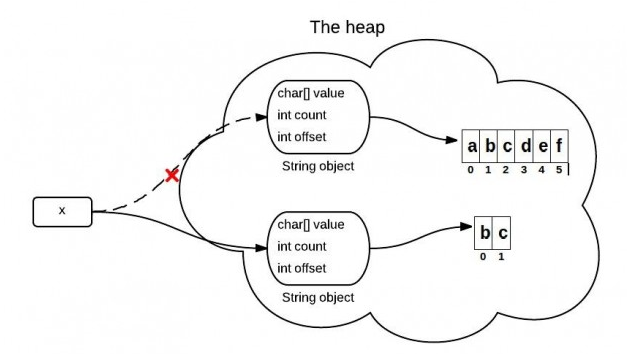
4. A problem caused by substring() in JDK 6

If you have a VERY long string, but you only need a small part each time by using substring(). This will cause a performance problem since you need only a small part, you keep the whole thing. For JDK 6, the solution is using the following, which will make it point to a real substring:

|  |
| --- |
| x = x.substring(x, y) + "" |

5. substring() in JDK 7

This is improved in JDK 7. In JDK 7, the substring() method actually create a new array in the heap.



|  |
| --- |
| *//JDK 7*  **public** String(**char** value[], **int** offset, **int** count) {  *//check boundary*  **this**.value = Arrays.copyOfRange(value, offset, offset + count);  }    **public** String substring(**int** beginIndex, **int** endIndex) {  *//check boundary*  **int** subLen = endIndex - beginIndex;  **return** **new** String(value, beginIndex, subLen);  } |

**1) What is String in Java? Is String is data type?**  
String in Java is not a primitive data type like int, long or double. The string is a class or in more simple term a user defined type. This is confusing for someone who comes from C background. String is defined in java.lang package and wrappers its content in a character array. String provides [equals() method](http://javarevisited.blogspot.sg/2011/02/how-to-write-equals-method-in-java.html) to compare two String and provides various other methods to operate on String like toUpperCase() to convert String into upper case, replace() to [replace String contents](http://javarevisited.blogspot.sg/2011/12/java-string-replace-example-tutorial.html), substring() to get substring, split() to [split long String](http://javarevisited.blogspot.sg/2011/09/string-split-example-in-java-tutorial.html) into multiple String.  
  
  
  
**2) Why is String final in Java?**  
The string is final by design in Java, some of the points which make sense why String is final is Security, optimization and to maintain a pool of String in Java. for details on each of this point see [Why String is final in Java](http://javarevisited.blogspot.sg/2010/10/why-string-is-immutable-in-java.html).

**3) What is the difference between String and StringBuffer in Java?**  
This is probably the most common question on String I have seen in Java interviews. Though String and StringBuffer are two different class they are used in the context of concatenating two Strings, Since String is immutable in Java every operation which changes String produces new String, which can be avoided by using StringBuffer. See [String vs StringBuffer](http://javarevisited.blogspot.sg/2011/07/string-vs-stringbuffer-vs-stringbuilder.html)for more details.

### StringBuffer and StringBuilder have the same methods with one difference and that’s of synchronization. StringBuffer is synchronized( which means it is thread safe and hence you can use it when you implement threads for your methods) whereas StringBuilder is not synchronized( which implies it isn’t thread safe). String vs StringBuffer

Since String is immutable in java, whenever we do String manipulation like concat, substring etc, it generates a new String and discard the older String for garbage collection.

These are heavy operations and generate a lot of garbage in heap. So Java has provided StringBuffer and StringBuilder class that should be used for String manipulation.

StringBuffer and StringBuilder are mutable objects in java and provide append(), insert(), delete() and substring() methods for String manipulation.

### //CITI interview questions

### StringBuffer vs StringBuilder

StringBuffer was the only choice for String manipulation till Java 1.4 but it has one disadvantage that all of its public methods are synchronized. StringBuffer provides Thread safety but on a performance cost.

In most of the scenarios, we don’t use String in multithreaded environment, so Java 1.5 introduced a new class StringBuilder that is similar with StringBuffer except thread safety and synchronization.

So if you are in a single threaded environment or don’t care about thread safety, you should use StringBuilder else use StringBuffer. See this post for [performance benchmarking between StringBuffer and StringBuilder](https://www.journaldev.com/137/stringbuffer-vs-stringbuilder).

### String vs StringBuffer vs StringBuilder

1. String is immutable whereas StringBuffer and StringBuider are mutable classes.
2. StringBuffer is thread safe and synchronized whereas StringBuilder is not, thats why [StringBuilder is more faster than StringBuffer](https://www.journaldev.com/137/stringbuffer-vs-stringbuilder).
3. String concat + operator internally uses StringBuffer or StringBuilder class.
4. For String manipulations in non-multi threaded environment, we should use StringBuilder else use StringBuffer class.

That’s all for a quick roundup of difference between String vs StringBuffer vs StringBuilder. In most of the scenarios for string manipulation, StringBuilder is better suited than StringBuffer.

**4) What is the difference in String on C and Java?**  
If you have mentioned C in your resume, then you are likely to face this String interview question. Well, C String and Java String are completely different to each other, C String is a null terminated [character array](http://javarevisited.blogspot.sg/2012/02/how-to-convert-char-to-string-in-java.html) while String in Java is an Object. Also, String is more feature rich in Java than C.  
  
  
  
**5) Why char array is better than String for storing password?**  
This String interview question is debatable and you might not agree with interviewer but this is also a chance to show that how deep and differently you can think of. One of the reasons which people give Why you should store a password in char array over String is related to immutability since it's not possible to erase contents of String but you can erase contents of a char array. See [Why char array preferred over String for a password](http://javarevisited.blogspot.sg/2012/03/why-character-array-is-better-than.html) for a complete discussion.  
  
  
  
**6) How do you compare two String in Java?**  
This is another common String interview question which appears on fresher level interviews. There are multiple ways to compare two String like equals() method, equalsIgnoreCase() etc, You can also see [4 ways to compare String in Java](http://javarevisited.blogspot.sg/2012/03/how-to-compare-two-string-in-java.html) for more examples. The main thing which interviewer checks is that whether candidate mentioned equality operator or not "==", comparing String with equality operator is a common mistake which works in some case and doesn't work in other. next String interview question is follow-up up of this.  
  
  
  
**7) Can we compare String using == operator? What is the risk?**  
As discussed in previous String question, You can compare String using equality operator but that is not suggested or advised because equality operator is used to compare primitives and equals() method should be used to compare objects. As we have seen in the [pitfall of autoboxing in Java](http://javarevisited.blogspot.sg/2012/07/auto-boxing-and-unboxing-in-java-be.html) that how equality operator can cause a subtle issue while comparing primitive to Object, anyway String is free from that issue because it doesn't have a corresponding primitive type and not participate in autoboxing.  
  
Almost all the time comparing String means comparing contents of String i.e. characters and equals() method is used to perform character-based comparison. equals() return true if two String points to the same object or two String has same contents while == operator returns true if two String object points to the same object but return false if two different String object contains same contents. That explains why sometimes it works and sometimes it doesn't.  
  
In short [always use equals method in Java](http://javarevisited.blogspot.sg/2011/02/how-to-write-equals-method-in-java.html) to check equality of two String object.  
StringBuffer  
StringBuffer is mutable means one can change the value of the object . The object created through StringBuffer is stored in the heap. StringBuffer has the same methods as the StringBuilder , but each method in StringBuffer is synchronized that is StringBuffer is thread safe . Due to this it does not allow two threads to simultaneously access the same method . Each method can be accessed by one thread at a time . But being thread safe has disadvantages too as the performance of the StringBuffer hits due to thread safe property . Thus StringBuilder is faster than the StringBuffer when calling the same methods of each class. String Buffer can be converted to the string by using toString() method.

StringBuffer demo1 = new StringBuffer("Hello") ; // The above object stored in heap and its value can be changed . demo1=new StringBuffer("Bye"); // Above statement is right as it modifies the value which is allowed in the StringBuffer

StringBuilder

StringBuilder is same as the StringBuffer , that is it stores the object in heap and it can also be modified . The main difference between the StringBuffer and StringBuilder is that StringBuilder is also not thread safe. StringBuilder is fast as it is not thread safe . StringBuilder demo2= new StringBuilder("Hello"); // The above object too is stored in the heap and its value can be modified demo2=new StringBuilder("Bye"); // Above statement is right as it modifies the value which is allowed in the StringBuilder  
  
**8) How does substring method work in Java?**  
This is one of the [tricky Java question](http://java67.blogspot.sg/2012/09/top-10-tricky-java-interview-questions-answers.html) relate to String and until you are familiar with the internals of String class, it's difficult to answer. Substring shares same character array as original String which can create a memory leak if original String is quite big and not required to retain in memory but unintentionally retained by substring which is very small in size and prevents large array from begin claimed during [Garbage collection in Java](http://javarevisited.blogspot.sg/2011/04/garbage-collection-in-java.html).

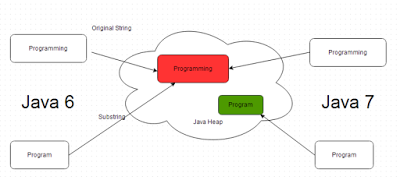
|  |
| --- |
| **Answer:** |
| String in java is a sequence of characters. String is more like a utility class which works on that character sequence. This character sequence is maintained as a array called value[], for example  *private final char value[];*  String internally defines two private variables called offset and count to manage the char array. The declarations can be as shown below:  */\*\* The offset is the first index of the storage that is used. \*/ private final int offset;  /\*\* The count is the number of characters in the String. \*/ private final int count;*  Everytime we create a substring from any string object, substring() method assigns the new values of offset and count variables. The internal char array is unchanged. This is a possible source of memory leak if substring() method is used without care. |

# [Memory Leak Fixed in JDK 1.7](http://javarevisited.blogspot.in/2011/10/how-substring-in-java-works.html)

**memory leak in Java** is a situation where some objects are not used by the application any

more, but GC fails to recognize them as unused.

<http://javarevisited.blogspot.in/2011/10/how-substring-in-java-works.html>

[](http://javarevisited.blogspot.sg/2011/10/how-substring-in-java-works.html)

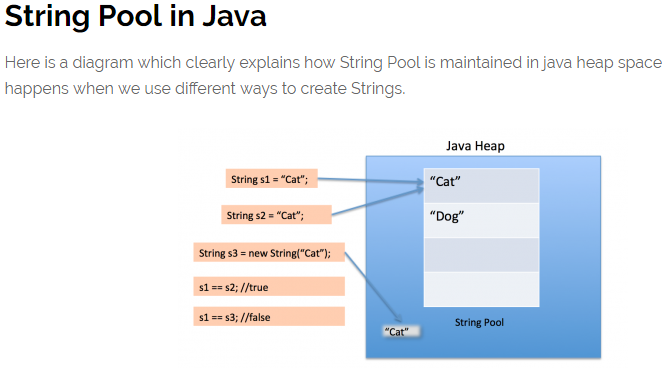
**9) What is String pool in Java?**

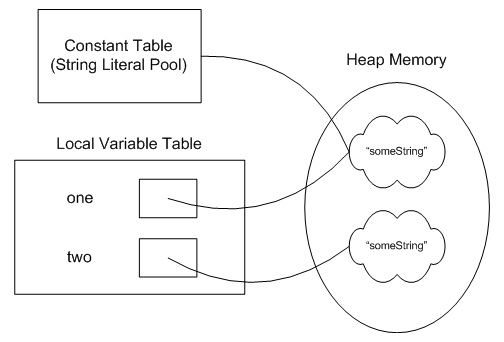
**Summary**

1) use == to compare primitive e.g. boolean, int, char etc, while use equals() to compare objects in Java.

2) == return true if two reference are of same object. Result of equals() method depends on overridden implementation.

3) For comparing String use equals() instead of  == equality operator.

  
Another [tough Java question](http://java67.blogspot.sg/2012/09/top-10-tough-core-java-interview-questions-answers.html) asked in String interview. String pool is a special storage area in Java heap, mostly located on PerGen space, to store String literals like "ABC". When Java program creates a new String using String literal, JVM checks for that String in the pool and if String literal is already present in the pool than the same object is returned instead of creating a whole new object. String pool check is only performed when you create String as literal, if you create [String using new() operator](http://java67.blogspot.sg/2014/08/difference-between-string-literal-and-new-String-object-Java.html" \t "_blank), a new String object will be created even if String with the same content is available in the pool.

[](https://1.bp.blogspot.com/-f_I0OwMN3KQ/VvUErQpn6HI/AAAAAAAAFXo/ZEn0nEP4du8HuRcdeXc_hsRIwPE7ZdFfQ/s1600/String+Pool+in+Java.jpg)

**10) What does intern() method do in Java?**  
As discussed in previous String interview question, String object created by new() operator is by default not added in String pool as opposed to String literal. The [intern method](http://javarevisited.blogspot.com/2015/12/when-to-use-intern-method-of-string-in-java.html" \t "_blank) allows putting a [String object](http://javarevisited.blogspot.sg/2012/08/convert-collection-to-string-in-java.html) into a pool.  
  
  
**11) Is string thread-safe in Java?**  
If you are familiar with the concept of immutability and [thread-safety](http://javarevisited.blogspot.sg/2011/07/java-multi-threading-interview.html) you can easily answer this String interview question in Java. Since [String is immutable](http://java67.blogspot.com/2014/01/why-string-class-has-made-immutable-or-final-java.html" \t "_blank), it is thread-safe and it can be shared between multiple threads without external synchronization.  
  
If you are seriously preparing for Java interviews and do not want to leave any stone unturned, I strongly suggest you go through questions given in [Java Programming Interview Exposed](http://aax-us-east.amazon-adsystem.com/x/c/QjFvp72EjjkP0DK42Cqt9eIAAAFd_h_CWgEAAAFKAV8uSgs/https:/assoc-redirect.amazon.com/g/r/http:/www.amazon.com/Java-Programming-Interviews-Exposed-Markham/dp/1118722868/ref=as_at?creativeASIN=1118722868&linkCode=w61&imprToken=dv2UCdJKBdHZBm9Nv-faXw&slotNum=0&tag=javamysqlanta-20" \t "_blank), one of the rare book which covers all important topics for Java interviews.

## String based Coding Questions

### These questions are most based upon Java's implementation of String and  you can only answer them well if you have good knowledge of java.lang.String class. But, String is very general data structure and you will find it in almost all programming and script language e.g. C, C++, C#, Python, Perl or Ruby. That's why I aWhy String is popular HashMap key in Java?

Since String is immutable, its hashcode is cached at the time of creation and it doesn’t need to be calculated again. This makes it a great candidate for key in a Map and it’s processing is fast than other HashMap key objects. This is why String is mostly used Object as HashMap keys.

### String Programming Questions

1. What is the output of below program?
2. package com.journaldev.strings;
3. public class StringTest {
4. public static void main(String[] args) {
5. String s1 = new String("pankaj");
6. String s2 = new String("PANKAJ");
7. System.out.println(s1 = s2);
8. }

}

It's a simple yet tricky program, it will print "PANKAJ" because we are assigning s2 String to s1. Don't get confused with == comparison operator.

1. What is the output of below program?

package com.journaldev.strings;

public class Test {

public void foo(String s) {

System.out.println("String");

}

public void foo(StringBuffer sb){

System.out.println("StringBuffer");

}

public static void main(String[] args) {

new Test().foo(null);

}

}

The above program will not compile with error as "The method foo(String) is ambiguous for the type Test". For complete clarification read [Understanding the method X is ambiguous for the type Y error](http://www.journaldev.com/9107/the-method-is-ambiguous-for-the-type-java-ambiguous-method-call-null-error).

1. What is the output of below code snippet?
2. String s1 = new String("abc");
3. String s2 = new String("abc");

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

It will print **false** because we are using new operator to create String, so it will be created in the heap memory and both s1, s2 will have different reference. If we create them using double quotes, then they will be part of string pool and it will print true.

1. What will be output of below code snippet?

String s1 = "abc";

StringBuffer s2 = new StringBuffer(s1);

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

It will print false because s2 is not of type String. If you will look at the equals method implementation in the String class, you will find a check using **instanceof** operator to check if the type of passed object is String? If not, then return false.

1. What will be output of below program?

String s1 = "abc";

String s2 = new String("abc");

s2.intern();

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

It's a tricky question and output will be **false**. We know that intern() method will return the String object reference from the string pool, but since we didn't assigned it back to s2, there is no change in s2 and hence both s1 and s2 are having different reference. If we change the code in line 3 to s2 = s2.intern(); then output will be true.

1. How many String objects got created in below code snippet?
2. String s1 = new String("Hello");

String s2 = new String("Hello");

Answer is 3.  
First - line 1, "Hello" object in the string pool.  
Second - line 1, new String with value "Hello" in the heap memory.  
Third - line 2, new String with value "Hello" in the heap memory. Here "Hello" string from string pool is reused.

m going to share some more [String based coding question](http://javarevisited.blogspot.com/2015/01/top-20-string-coding-interview-question-programming-interview.html), which is not Java specific. You can solve these question in any programming language as they are mostly logic based programming question.  
 **1) Write a Java program to reverse String in Java without using any API?** ([solution](http://java67.blogspot.com/2012/12/how-to-reverse-string-in-java-stringbuffer-stringbuilder.html" \t "_blank))  
This means you can not use StringBuffer's reverse() method or any of String utility method, all you can have is a character array for reversing contents.  
  
  
**2) Write a Program to check if a String is a palindrome or not?**([solution](http://java67.blogspot.com/2012/09/palindrome-java-program-to-check-number.html" \t "_blank))  
For example, a String e.g. "madam" is a palindrome but "book" is not a palindrome. You also need to solve this question without taking any help from Java String API.  
  
  
**3) Write a Java program to check if two String are Anagram or not?**([solution](http://javarevisited.blogspot.com/2013/03/Anagram-how-to-check-if-two-string-are-anagrams-example-tutorial.html))  
You need to write method e.g. isAnagram(String first, String second) which will return true if second String is an anagram of the first string. An anagram must contain the same number of characters and exactly same characters but in different order e.g. top and pot, or army and mary.  
  
 **4) Write a method in Java to remove any character from String?** ([solution](http://javarevisited.blogspot.com/2016/03/how-to-remove-first-and-last-character-from-String-in-java-example.html))  
For example, you need to write method remove(String word, char removeThis), this method should return a String without character, which is asked to remove. you can use indexOf(), substring() and similar methods from String class, but your method must handle corner cases e.g. passing null or empty String, String containing just one character etc.  
  
  
**5) Write a method to split a comma separated String in Java?**([solution](http://java67.blogspot.com/2016/01/how-to-split-string-by-comma-in-java-with-example.html))  
  
  
**6) Write Java program to print all permutations of a String** e.g. passing "ABC" will print all permutations like "BCA", "CBA" etc ([solution](http://javarevisited.blogspot.com/2015/08/how-to-find-all-permutations-of-string-java-example.html" \t "_blank))  
  
If you are hungry for more String based coding question, you can also check the [Cracking the Coding Interview](http://aax-us-east.amazon-adsystem.com/x/c/QjFvp72EjjkP0DK42Cqt9eIAAAFd_h_CWgEAAAFKAV8uSgs/https:/assoc-redirect.amazon.com/g/r/http:/www.amazon.com/dp/098478280X/ref=as_at?creativeASIN=098478280X&linkCode=w61&imprToken=dv2UCdJKBdHZBm9Nv-faXw&slotNum=2&tag=javamysqlanta-20" \t "_blank) book, a collection of 189 programming questions and solutions from various programming job interviews of reputed tech companies like Amazon, Google, Facebook, and Microsoft.

That's all about **Java String interview question and answers**. In Summary, there are a lot of specifics about String which needs to be known for anyone who has started Java programming and these String question will not just help to perform better on Java Interviews but also opens the new door of learning about String. I didn't know much String related concepts until I come across these question which motivated to research and learns more about String in Java.  
  
  
Other **Java String tutorials and questions** from Javarevisited Blog

1. [How to convert Date to String in Java](http://javarevisited.blogspot.sg/2011/09/convert-date-to-string-simpledateformat.html)
2. [How to convert Enum to String in Java](http://javarevisited.blogspot.sg/2011/12/convert-enum-string-java-example.html)
3. [How to create comma separated String from Collection in Java](http://javarevisited.blogspot.sg/2012/08/convert-collection-to-string-in-java.html)
4. [How to convert String to Double in Java](http://javarevisited.blogspot.sg/2011/10/convert-double-to-string-example.html)
5. [How to reverse String in Java with recursion](http://javarevisited.blogspot.sg/2012/01/how-to-reverse-string-in-java-using.html)
6. [How to format String in Java](http://javarevisited.blogspot.sg/2012/08/how-to-format-string-in-java-printf.html)

[by Taboola](http://popup.taboola.com/en/?template=colorbox&utm_source=javarevisited&utm_medium=referral&utm_content=thumbnails-a:Below%20Article%20Thumbnails:" \t "_blank)

[Sponsored Links](http://popup.taboola.com/en/?template=colorbox&utm_source=javarevisited&utm_medium=referral&utm_content=thumbnails-a:Below%20Article%20Thumbnails:" \t "_blank)

**You May Like**

**What is String in Java? String is a data type?**

String is a Class in java and defined in java.lang package. It’s not a primitive data type like int and long. String class represents character Strings. String is used in almost all the Java applications and there are some interesting facts we should know about String. String in immutable and final in Java and JVM uses String Pool to store all the String objects.  
Some other interesting things about String is the way we can instantiate a String object using double quotes and overloading of “+” operator for concatenation.

**What are different ways to create String Object?**

We can create String object using new operator like any normal java class or we can use double quotes to create a String object. There are several constructors available in String class to get String from char array, byte array, StringBuffer and StringBuilder.

String str = new String("abc");

String str1 = "abc";

When we create a String using double quotes, JVM looks in the String pool to find if any other String is stored with same value. If found, it just returns the reference to that String object else it creates a new String object with given value and stores it in the String pool.  
When we use new operator, JVM creates the String object but don’t store it into the String Pool. We can use intern() method to store the String object into String pool or return the reference if there is already a String with equal value present in the pool.

**Write a method to check if input String is Palindrome?**

A String is said to be Palindrome if it’s value is same when reversed. For example “aba” is a Palindrome String.  
String class doesn’t provide any method to reverse the String but StringBuffer and StringBuilder class has reverse method that we can use to check if String is palindrome or not.

private static boolean isPalindrome(String str) {

if (str == null)

return false;

StringBuilder strBuilder = new StringBuilder(str);

strBuilder.reverse();

return strBuilder.toString().equals(str);

}

Sometimes interviewer asks not to use any other class to check this, in that case we can compare characters in the String from both ends to find out if it’s palindrome or not.

private static boolean isPalindromeString(String str) {

if (str == null)

return false;

int length = str.length();

System.out.println(length / 2);

for (int i = 0; i < length / 2; i++) {

if (str.charAt(i) != str.charAt(length - i - 1))

return false;

}

return true;

}

**Write a method that will remove given character from the String?**

We can use replaceAll method to replace all the occurance of a String with another String. The important point to note is that it accepts String as argument, so we will use Character class to create String and use it to replace all the characters with empty String.

private static String removeChar(String str, char c) {

if (str == null)

return null;

return str.replaceAll(Character.toString(c), "");

}

**How can we make String upper case or lower case?**

We can use String class toUpperCase and toLowerCase methods to get the String in all upper case or lower case. These methods have a variant that accepts Locale argument and use that locale rules to convert String to upper or lower case.

**What is String subSequence method?**

Java 1.4 introduced CharSequence interface and String implements this interface, this is the only reason for the implementation of subSequence method in String class. Internally it invokes the String substring method.  
Check this post for [String subSequence](https://www.journaldev.com/813/java-string-subsequence-example) example.

**How to compare two Strings in java program?**

Java String implements Comparable interface and it has two variants of compareTo() methods.

compareTo(String anotherString) method compares the String object with the String argument passed lexicographically. If String object precedes the argument passed, it returns negative integer and if String object follows the argument String passed, it returns positive integer. It returns zero when both the String have same value, in this case equals(String str) method will also return true.

compareToIgnoreCase(String str): This method is similar to the first one, except that it ignores the case. It uses String CASE\_INSENSITIVE\_ORDER Comparator for case insensitive comparison. If the value is zero then equalsIgnoreCase(String str) will also return true.  
Check this post for [String compareTo](https://www.journaldev.com/810/java-string-compareto-examples) example.

**How to convert String to char and vice versa?**

This is a tricky question because String is a sequence of characters, so we can't convert it to a single character. We can use use charAt method to get the character at given index or we can use toCharArray()method to convert String to character array.  
Check this post for sample program on converting [String to character array to String](https://www.journaldev.com/794/string-char-array-java).

**How to convert String to byte array and vice versa?**

We can use String getBytes() method to convert String to byte array and we can use String constructor new String(byte[] arr) to convert byte array to String.  
Check this post for [String to byte array](https://www.journaldev.com/770/string-byte-array-java) example.

**Can we use String in switch case?**

This is a tricky question used to check your knowledge of current Java developments. Java 7 extended the capability of switch case to use Strings also, earlier java versions doesn't support this.  
If you are implementing conditional flow for Strings, you can use if-else conditions and you can use switch case if you are using Java 7 or higher versions.

Check this post for [Java Switch Case String](https://www.journaldev.com/588/java-switch-case-string) example.

**Write a program to print all permutations of String?**

This is a tricky question and we need to use recursion to find all the permutations of a String, for example "AAB" permutations will be "AAB", "ABA" and "BAA".  
We also need to use Set to make sure there are no duplicate values.  
Check this post for complete program to [find all permutations of String](https://www.journaldev.com/526/java-program-to-find-all-permutations-of-a-string).

**Write a function to find out longest palindrome in a given string?**

A String can contain palindrome strings in it and to find longest palindrome in given String is a programming question.  
Check this post for complete program to find longest [palindrome in a String](https://www.journaldev.com/530/longest-palindrome-string-java).

**Difference between String, StringBuffer and StringBuilder?**

String is immutable and final in java, so whenever we do String manipulation, it creates a new String. String manipulations are resource consuming, so java provides two utility classes for String manipulations - StringBuffer and StringBuilder.  
StringBuffer and StringBuilder are mutable classes. StringBuffer operations are thread-safe and synchronized where StringBuilder operations are not thread-safe. So when multiple threads are working on same String, we should use StringBuffer but in single threaded environment we should use StringBuilder.  
StringBuilder performance is fast than StringBuffer because of no overhead of synchronization.

Check this post for extensive details about [String vs StringBuffer vs StringBuilder](https://www.journaldev.com/538/string-vs-stringbuffer-vs-stringbuilder).  
Read this post for benchmarking of [StringBuffer vs StringBuilder](https://www.journaldev.com/137/stringbuffer-vs-stringbuilder).

**Why String is immutable or final in Java**

There are several benefits of String because it's immutable and final.

* String Pool is possible because String is immutable in java.
* It increases security because any hacker can't change its value and it's used for storing sensitive information such as database username, password etc.
* Since String is immutable, it's safe to use in multi-threading and we don't need any synchronization.
* Strings are used in [java classloader](https://www.journaldev.com/349/java-classloader) and immutability provides security that correct class is getting loaded by Classloader.

Check this post to get more details [why String is immutable in java](https://www.journaldev.com/802/string-immutable-final-java).

**How to Split String in java?**

We can use split(String regex) to split the String into String array based on the provided regular expression.  
Learn more at [java String split](https://www.journaldev.com/791/java-string-split).

**Why Char array is preferred over String for storing password?**

String is immutable in java and stored in String pool. Once it's created it stays in the pool until unless garbage collected, so even though we are done with password it's available in memory for longer duration and there is no way to avoid it. It's a security risk because anyone having access to memory dump can find the password as clear text.  
If we use char array to store password, we can set it to blank once we are done with it. So we can control for how long it's available in memory that avoids the security threat with String.

**How do you check if two Strings are equal in Java?**

There are two ways to check if two Strings are equal or not - using "==" operator or using equals method. When we use "==" operator, it checks for value of String as well as reference but in our programming, most of the time we are checking equality of String for value only. So we should use equals method to check if two Strings are equal or not.  
There is another function equalsIgnoreCase that we can use to ignore case.

String s1 = "abc";

String s2 = "abc";

String s3= new String("abc");

System.out.println("s1 == s2 ? "+(s1==s2)); //true

System.out.println("s1 == s3 ? "+(s1==s3)); //false

System.out.println("s1 equals s3 ? "+(s1.equals(s3))); //true

**What is String Pool?**

As the name suggests, String Pool is a pool of Strings stored in Java heap memory. We know that String is special class in java and we can create String object using new operator as well as providing values in double quotes.  
Check this post for more details about [String Pool](https://www.journaldev.com/797/what-is-java-string-pool).

**What does String intern() method do?**

When the intern method is invoked, if the pool already contains a string equal to this String object as determined by the equals(Object) method, then the string from the pool is returned. Otherwise, this String object is added to the pool and a reference to this String object is returned.  
This method always return a String that has the same contents as this string, but is guaranteed to be from a pool of unique strings.

**Does String is thread-safe in Java?**

Strings are immutable, so we can't change it's value in program. Hence it's thread-safe and can be safely used in multi-threaded environment.  
Check this post for [Thread Safety in Java](https://www.journaldev.com/1061/thread-safety-in-java).

**Why String is popular HashMap key in Java?**

Since String is immutable, its hashcode is cached at the time of creation and it doesn’t need to be calculated again. This makes it a great candidate for key in a Map and it’s processing is fast than other HashMap key objects. This is why String is mostly used Object as HashMap keys.

**String Programming Questions**

1. What is the output of below program?
2. package com.journaldev.strings;
3. public class StringTest {
4. public static void main(String[] args) {
5. String s1 = new String("pankaj");
6. String s2 = new String("PANKAJ");
7. System.out.println(s1 = s2);
8. }
9. }

It's a simple yet tricky program, it will print "PANKAJ" because we are assigning s2 String to s1. Don't get confused with == comparison operator.

1. What is the output of below program?
2. package com.journaldev.strings;
3. public class Test {
4. public void foo(String s) {
5. System.out.println("String");
6. }
7. public void foo(StringBuffer sb){
8. System.out.println("StringBuffer");
9. }
10. public static void main(String[] args) {
11. new Test().foo(null);
12. }
13. }

The above program will not compile with error as "The method foo(String) is ambiguous for the type Test". For complete clarification read [Understanding the method X is ambiguous for the type Y error](https://www.journaldev.com/9107/the-method-is-ambiguous-for-the-type-java-ambiguous-method-call-null-error).

1. What is the output of below code snippet?
2. String s1 = new String("abc");
3. String s2 = new String("abc");
4. System.out.println(s1 == s2);

It will print **false** because we are using *new* operator to create String, so it will be created in the heap memory and both s1, s2 will have different reference. If we create them using double quotes, then they will be part of string pool and it will print true.

1. What will be output of below code snippet?
2. String s1 = "abc";
3. StringBuffer s2 = new StringBuffer(s1);
4. System.out.println(s1.equals(s2));

It will print false because s2 is not of type String. If you will look at the equals method implementation in the String class, you will find a check using **instanceof** operator to check if the type of passed object is String? If not, then return false.

1. What will be output of below program?
2. String s1 = "abc";
3. String s2 = new String("abc");
4. s2.intern();
5. System.out.println(s1 ==s2);

It's a tricky question and output will be **false**. We know that intern() method will return the String object reference from the string pool, but since we didn't assigned it back to s2, there is no change in s2 and hence both s1 and s2 are having different reference. If we change the code in line 3 to s2 = s2.intern(); then output will be true.

1. How many String objects got created in below code snippet?
2. String s1 = new String("Hello");
3. String s2 = new String("Hello");

Answer is 3.  
First - line 1, "Hello" object in the string pool.  
Second - line 1, new String with value "Hello" in the heap memory.  
Third - line 2, new String with value "Hello" in the heap memory. Here "Hello" string from string pool is reused.

I hope that the questions listed here will help you in java interviews, please let me know if I have missed anything.

Make sure to check out [Java Programming Questions](https://www.journaldev.com/370/java-programming-interview-questions)

Filed Under: [Interview Questions](https://www.journaldev.com/dev/interview-questions), [Java](https://www.journaldev.com/dev/java)

### String in Java

1. String class represents character strings, we can instantiate String by two ways.  
   String str = "abc"; or String str = new String ("abc");
2. String is immutable in java, so its easy to share it across different threads or functions.
3. When we create a String using double quotes, it first looks for the String with same value in the JVM string pool, if found it returns the reference else it creates the String object and then place it in the String pool. This way JVM saves a lot of space by using same String in different threads. But if new operator is used, it explicitly creates a new String in the heap memory.
4. + operator is overloaded for String and used to concatenate two Strings. Although internally it uses StringBuffer to perform this action.
5. String overrides equals() and hashCode() methods, two Strings are equal only if they have same characters in same order. Note that equals() method is case sensitive, so if you are not looking for case sensitive checks, you should use equalsIgnoreCase() method.
6. A String represents a string in the UTF-16 format
7. String is a final class with all the fields as final except “private int hash”. This field contains the hashCode() function value and created only when hashCode() method is called and then cached in this field. Furthermore, hash is generated using final fields of String class with some calculations, so every time hashCode() method is called, it will result in same output. For caller, its like calculations are happening every time but internally it’s cached in hash field.