在**Java**中**String**是什么？**String**是一种数据类型？

String是在java和在java.lang包中定义的一个类，不像基本数据类型int和long。 String类代表字符的字符串。String几乎用在所有的Java应用程序中。String是不可变的，final的，在Java和JVM使用字符串池来存储所有的String对象。 我们可以使用双引号和“+”操作符的重载串连实例化一个String对象的方式。

有什么不同的方法来创建**String**对象的？

我们可以像任何普通的Java类一样使用new运算符来创建String对象，或者我们可以使用双引号来创建一个String对象。String类提供了从char数组，字节数组的StringBuffer和StringBuilder String对象的几个构造函数。

String str = new String(“abc”);

String str1 = “abc”;

当我们使用双引号创建一个字符串时，JVM会先在字符串池中查找是否其他任何字符串存储具有相同的值。如果找到了，返回的它引用，否则创建一个新的String对象，并将其值存储在字符串池中。

当我们使用new运算符时创建String对象时，但JVM不会将其存储到String Pool中。除非我们使用intern（）方法将String对象存储到字符串池中，值如果已经存在String 则返回引用。

写一个方法来检查输入的字符串是否是回文（对称）？

```
private static boolean isPalindrome(String str) {
if (str == null)
return false;
StringBuilder strBuilder = new StringBuilder(str);
strBuilder.reverse();
return strBuilder.toString().equals(str);
}
```

有时候，面试官要求不使用任何其他类来检查这一点，在这种情况下，我们可以从两端String中的字符进行比较，以找出是否是回文与否。

```
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;
}
```

写一个方法从字符串中删除给定的字符？

我们可以使用的replaceAll方法替换字符串的中的字符。

```
private static String removeChar(String str, char c) {
if (str == null)
return null;
return str.replaceAll(Character.toString(c), “”);
}
```

怎样才能转换字符串大写或小写？

我们可以使用String类的toUpperCase和toLowerCase方法来转换大小写。

如何在**java**程序比较两个字符串？

Java的String实现Comparable接口，使用对象的compaareTo 方法比较两个字符串的大小。

我们可否在**switch** 条件中使用字符串？

这是一个用来检查你的当前Java发展的知识一个棘手的问题。 在Java 7是可以的，而早期的Java版本不支持这个。

编写一个程序，打印字符串的所有排列？

这是一个比较的题目，我们需要使用递归找到一个字符串的所有排列，如“AAB”的排列将是“AAB”，“ABA”和“BAA”。
我们还需要检查，以确保没有重复的值。

```
package com.jd.first;

import java.util.HashSet;
import java.util.Set;

/**
* Java Program to find all permutations of a String
* @author pankaj
*
*/
public class StringHelper {
public static Set<String> permutationFinder(String str) {
Set<String> perm = new HashSet<String>();
//Handling error scenarios
if (str == null) {
return null;
} else if (str.length() == 0) {
perm.add("");
return perm;
}
char initial = str.charAt(0); // first character
String rem = str.substring(1); // Full string without first character
```

```
Set<String> words = permutationFinder(rem);
for (String strNew : words) {
for (int i = 0;i<=strNew.length();i++){
perm.add(charInsert(strNew, initial, i));
}
}
return perm;
}

public static String charInsert(String str, char c, int j) {
String begin = str.substring(0, j);
String end = str.substring(j);
return begin + c + end;
}

public static void main(String[] args) {
String s = “AAC”;
String s1 = “ABC”;
String s2 = “ABCD”;
System.out.println(“\nPermutations for ” + s + ” are: \n” +
permutationFinder(s));
System.out.println(“\nPermutations for ” + s1 + ” are: \n” +
permutationFinder(s1));
System.out.println(“\nPermutations for ” + s2 + ” are: \n” +
permutationFinder(s2));
}
}
```