Q>

```
class A {
    public String toString() {
        return null;
    }
}
public class Test {
    public static void main(String [] args) {
        String text = null;
        text = text + new A(); //Line n1   // JVM  text = null + "null"  text =
"nullnull"
        System.out.println(text.length()); //Line n2
    }
}
```

A. Line n1 causes compilation error
B. Line n1 causes Runtime error
C. Line n2 causes RunTime error
D. 0
E. 4
G. 8

Answer: G

Consider below code:

```
//Test.java
class SpecialString {
    String str;
    SpecialString(String str) {
        this.str = str;
    }
}
public class Test {
    public static void main(String[] args) {
        Object [] arr = new Object[4];
        for(int i = 1; i <=3; i++) {
            switch(i) {
                case 1:
                    arr[i] = new String("Java");
                    break;
                case 2:
                    arr[i] = new StringBuilder("Java");
                    break;
                case 3:
                    arr[i] = new SpecialString("Java");
                    break;
            }
        }
        for(Object obj : arr) {
            System.out.println(obj);
        }
    }
}
```

What will be the result of compiling and executing Test class?

A. Java
   Java
   Java
B. Java
   Java

```
   Some text with @symbol
C. Java
   Some text with @symbol
   Some text with @symbol
D. null
   Java
   Java
   Java
E. null
   Java
   Java
   Some text with @symbol
F. null
   Java
   Some text with @symbol
   Some text with @symbol
G. Java
   Java
   Java
   null
H. Java
   Java
   Some text with @symbol
   null
I. Java
   Some text with @symbol
   Some text with @symbol
   null

Answer: E

Q> .
class MyStringClass extends String
{
    String name;
}

Output: CompileTime Error

Q>
String name = "sachinrameshtendulkar".substring(4);
System.out.println(name);//inrameshtendulkar

Q> .
String s = "1".repeat(5);
System.out.println(s);//11111

Q>.
System.out.println("1".concat("2").repeat(5).charAt(7));
    1212121212.charAt(7) - > 2

Q>
To which of the following classes, you can create objects without using new
operator?
String
StringBuffer
StringBuilder
Answer: String
```

Q> .
```
String string = "string".replace('i', '0');
System.out.println(string.substring(2, 5));
```

string = "strOng";
output: rOn

Q> .
In my application, I want mutable and thread safe string objects. Which class do you refer me to use? String or StringBuffer or StringBuilder?
Answer StringBuffer(synchronized)

Q> .
```
System.out.println("Java" == new String("Java"));//false
```

Q>
```
 String str = "   Ineuron\tTechnology\tPrivateLimited\tKnown\tfor\tjava
".strip();
 System.out.println(str);// Ineuron   Technology    PrivateLimited   Know   for
java
```

Q> .
```
 if("string".toUpperCase() == "STRING")
  {
      System.out.println(true);
  }
  else
   {
       System.out.println(false);
   }
```
Answer: false(comparison happened b/w heap area object and SCP)

Q> .
String, StringBuffer and StringBuilder – all these three classes are final classes. True or False?
Answer: Yes

Q> .
```
String str1 = "1";
String str2 = "22";
String str3 = "333";
System.out.println(str1.concat(str2).concat(str3).repeat(3));
```

Answer: "122".concat("333")
"122333".repeat(3)
122333122333122333

Q>Ronaldo is developing an application in which string concatenation is very frequent.
Which string class do you refer him to use? And also he doesn’t need code to be thread safe.
StringBuilder(1.5V)

Q>.
```
System.out.println("Java"+1000+2000+3000); // "java1000"+2000+3000 =>
"java10002000" + 3000 => "java100020003000"
```

Q>.
```
System.out.println(1000+2000+3000+"Java");//3000+3000+"java" => 6000+"java"
```

=>"6000java"

Q>.
System.out.println(7.7+3.3+"Java"+3.3+7.7);//11.0 + "java" + 3.3 +7.7 => "11.0 java"+3.3 => "11.0java3.3" + 7.7 => "11.0java3.37.7"

Q>.
System.out.println("ONE"+2+3+4+"FIVE");//"ONE2" + 3+4+"FIVE" => "ONE23" + 4 +"FIVE"=> "ONE234+" Five" => "ONE234Five"

Q>.
```
String s1=" ";
System.out.println(s1.isBlank());//true
System.out.println(s1.isEmpty());//false
```

Q>.
```
String s2="sachin ramesh tendulkar";
System.out.println(s2.substring(8, 4));
```

A. CE
B. rame
C. in ram
D. NullPointerException
E. StringIndexOutOfBoundsException
F. ArrayIndexOutOfBoundsException
Answer: E

Q>.
```
String s1 = new String("JAVA");
String s2 = new String("JAVA");
System.out.println(s1 == s2);
System.out.println(s1.equals(s2));
System.out.println(s1 == s2.intern());
System.out.println(s1.intern() == s2.intern());
System.out.println(s1.intern() == s2);
```