

# ***Solutions to Quick Check Questions***

## **8**

## ***Characters and Strings***

---

### **8.1 Characters**



#### ***Quick Check***

1. Determine the output of the following statements:

a. `System.out.println( (char) 65 );`

*A*

b. `System.out.println( (int) 'C' );`

*67*

c. `System.out.println( 'Y' );`

*Y*

d. `if ( 'A' < '?' )  
    System.out.println( 'A' );  
else`

```
System.out.println( '?' );
```

?

2. How many distinct characters can you represent by using eight bits?

$$2^8 = 256$$

## 8.2 Strings



### Quick Check

1. Determine the output of the following code:

```
a.   String str = "Programming";
      for (int i = 0; i < 9; i+=2) {
          System.out.print( str.charAt( i ) );
      }
```

*Pormi*

```
b.   String str = "World Wide Web";
      for (int i = 0; i < 10; i ++ ) {
          if ( str.charAt(i) == 'W' ) {
              System.out.println( 'M' );
          }
          else {
              System.out.print( str.charAt(i) );
          }
      }
```

*M  
orld  
M  
ide*

2. Write a loop that prints out a string in reverse. If the string is Hello then the code outputs olleH. Use System.out.

```
int max = str.length()-1;
for (int i = max; i >= 0; i--)
    System.out.print( str.charAt(i) );
}
```

3. Assuming two String objects str1 and str2 are initialized as follows:

```
String str1 = "programming";
String str2 = "language";
```

Determine the value of each of the following expressions if they are valid. If they are not valid, state the reason why.

a. `str1.compareTo( str2 )`

*positive number*

b. `str2.compareTo( str2 )`

*0*

c. `str2.substring( 1, 1 )`

*“” //empty string*

d. `str2.substring( 0, 7 );`

*“languag”*

e. `str2.charAt( 11 );`

*invalid —out of bounds error*

f. `str1.length( ) + str2.length( )`

*19*

4. What is the difference between the two String methods equals and equalsIgnoreCase?

*equals is a case-sensitive comparison while equalsIgnoreCase is not.*

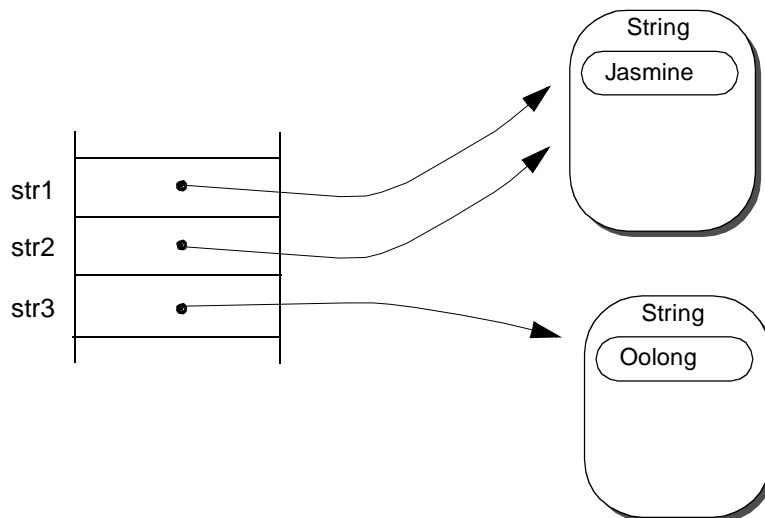
### 8.3 Primitive versus Reference Types



#### Quick Check

1. Show the state of memory after the following statements are executed:

```
String str1, str2, str3;  
str1 = "Jasmine";  
str2 = "Oolong";  
str3 = str2;  
str2 = str1;
```



### 8.4 StringBuffer



#### Quick Check

1. Determine the value of str after the following statements are executed:

```
a.    StringBuffer str  
      = new StringBuffer( "Caffeine" );
```

```
str.insert(0, "Dr. ");
```

*“Dr. Caffeine”*

```
b.    String      str    = "Caffeine";
      StringBuffer str1   =
          new StringBuffer( str.substring(1, 3) );
      str1.append('e');
      str = "De" + str1;
```

*“Deafe”*

```
c.    String      str    = "Caffeine";
      StringBuffer str1   =
          new StringBuffer( str.substring(4, 8) );
      str1.insert(3, 'f');
      str = "De" + str1;
```

*“Deeinfo”*

2. Assume a String object str is assigned as

```
String str = inputBox.getString();
```

Write a code segment to replace all occurrences of lowercase vowels in a given string to the letter C by using String and StringBuffer objects.

```
StringBuffer strBuf = new StringBuffer( "" );
int max = str.length();
char letter;

for (int i = 0; i < max; i++) {
    letter = str.charAt( i );

    if (letter == 'a' || letter == 'e' ||
        letter == 'i' || letter == 'o' ||
        letter == 'u' ) {

        strBuf.append('C');
    }
    else {
        strBuf.append( letter );
    }
}
```

```
    }
    str = strBuf.toString();
```

3. Find the errors in the following code:

```
String      str      = "Caffeine";
1 → StringBuffer str1 = str.substring(1, 3);
    str1.append('e');
2 → System.out(str1);
    str1 = str1 + str;
```

1. Cannot assign a String value to a StringBuffer variable.

2. Method print or println is missing.

## 8.5 Passing Objects as Parameters



### Quick Check

1. Draw the state-of-memory diagram for the following code:

```
String      str1 = "Daisy",
            str2 = "Iris";
Tester      tester = new Tester( );

tester.exchange( str1, str2);
...
class Tester
{
    public void exchange( String one, String two )
    {
        String temp;
        temp  = one;
        one   = two;
        two   = temp;
    }
}
```

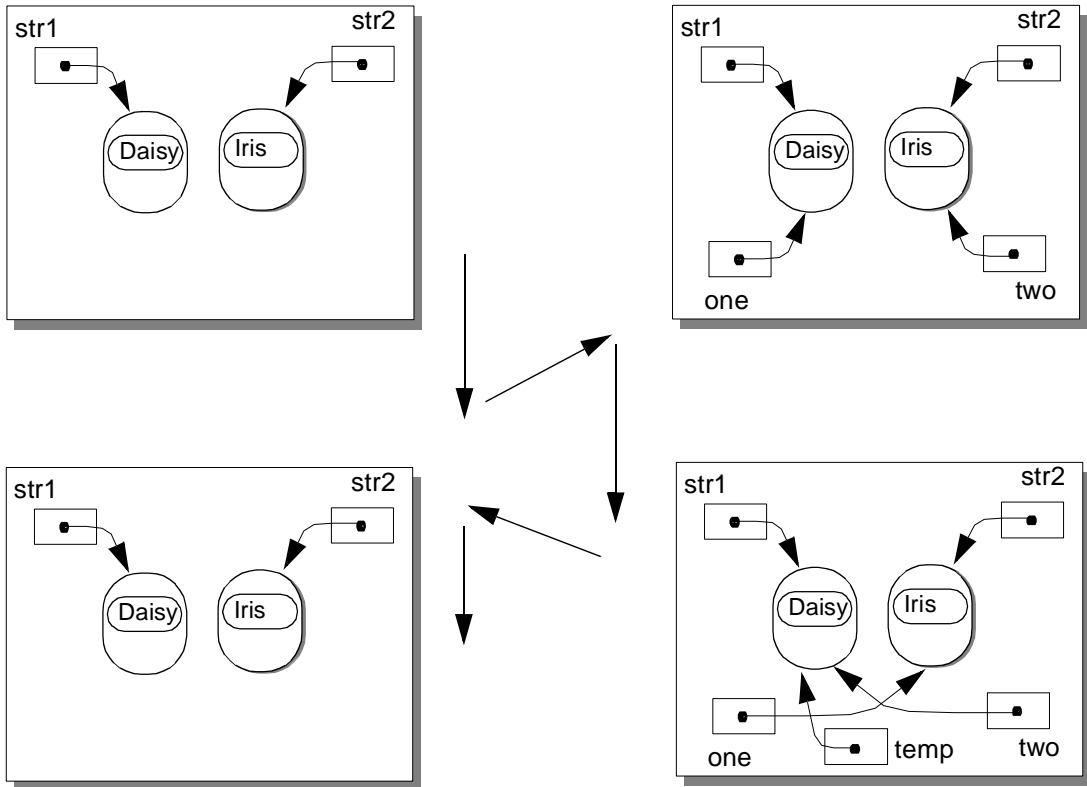
## 8.6 Returning an Object from Methods



### Quick Check

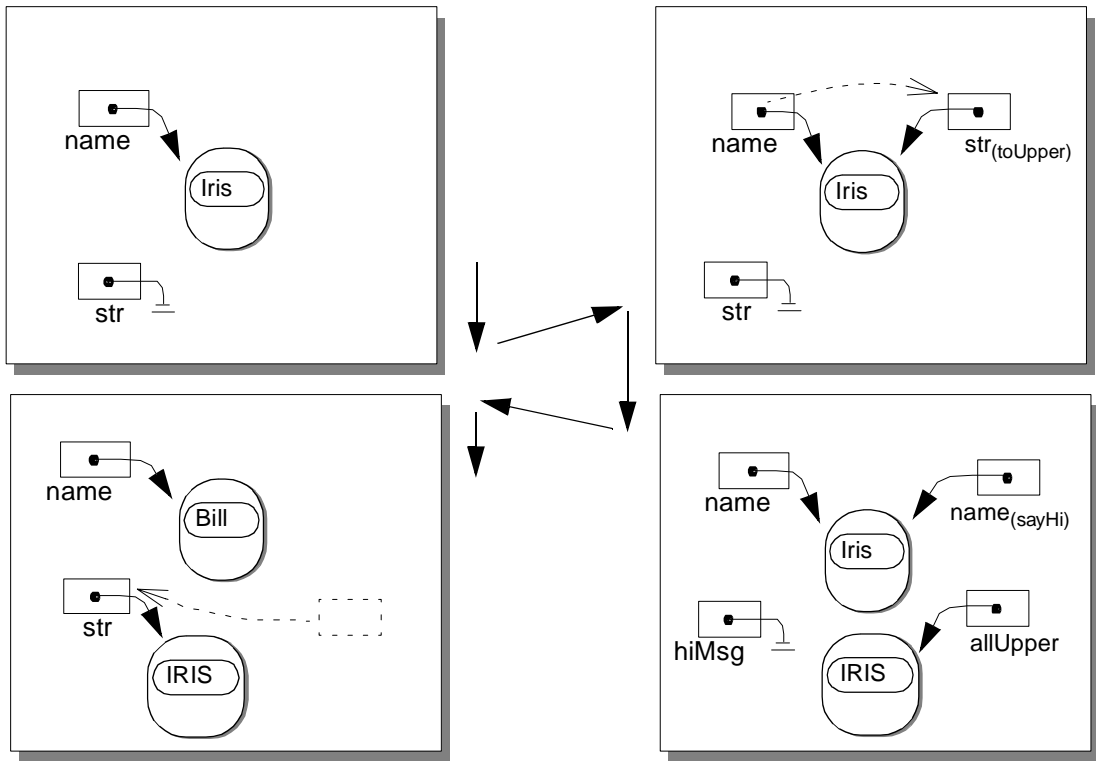
1. Draw the state-of-memory diagram for the following code:

## State of Memory



```
String    str, name = "Iris";
Tester    tester = new Tester( );
str        = tester.toUpper( name );
...
class Tester
{
    public String toUpper( String str )
    {
        String allUpper;
        allUpper = str.toUpperCase();
        return allUpper;
    }
}
```

}

**State of Memory**

2. The following code uses the `Tester` class from question 1. Determine the output.

```
String    str = "Internet";
Tester    tester = new Tester( );

System.out.println( str + " is " +
```



```
tester.toUpper( str ) );
```

```
Internet is INTERNET
```

## 8.7 Sample Program: Word Play

---

No Quick Check Questions.

