int x = 2;
int y = 4;
if (false || true)
{
    if (x < y)
        x = x * x;
    else
        x = x + y;
}
if ((true && false) || true)
{
    if (x <= y)
        y = y * y;
    else
        y = x + y;
}
if (x <= y)
    x = x + y;
System.out.println("x = " + x);
System.out.println("y = " + y);

What is the output of this code fragment?

a) x = 18
   y = 16
b) x = 6
   y = 4
c) x = 8
   y = 4
d) x = 16
   y = 10
e) x = 20
   y = 16
BST

Public class Node
{
    private Node left;    //left child node
    private Node right;  //right child node
    private int data;

    public Node getLeft() {
        return left;  }
    public Node getRight() {
        return right; }
    public int getData() {
        return data; }
    ...
    ...
}

public void Traverse(Node root)
{
    if ( root.getLeft() != null )
        Traverse ( root.getLeft() );
    if ( root.getRight() != null )
        Traverse ( root.getRight() );
    if ( root.getData() != null )
        System.out.println(
        root.getData() );
}

What is the output of this Traversal method on the above BST, where 9 is passed as the root?

a)  4  b)  3  c)  9  d)  9  e)  9
    6  4  7  7  13
    5  5  13  3  16
    3  6  3  5  14
    7  7  16  4  7
    14  9  5  6  3
    16  13  14  13  5
    13  14  4  16  6
    9  16  6  14  4
```c
int array1 = { 3, 5, 4, 8, 2, 6, 7 };
int array2 = { 2, 1, 3, 4 };

array1[ 3 ] = array2[ 2 ];
    array1[ 2 ] += 3;
array1[ 4 ] = array1[ 6 ];
```

What is the value of array1 after this code is executed?

a) { 3, 9, 1, 6, 6, 6, 7 }  
b) { 3, 5, 4, 3, 7, 8, 7 }  
c) { 3, 5, 8, 3, 7, 8, 7 }  
d) { 3, 5, 5, 3, 7, 8, 7 }  
e) { 3, 5, 8, 8, 7, 8, 7 }

HASH TABLES

Consider the resulting Hash Table that is made when the following < key, data> pairs are added. The hash code form of the key is given in the parenthesis after the pair. The HashTable is of size 5 and uses external chaining. Which is the correct representation of the data stored in the Hash Table?

< “Red” , “Apple” > ( 54 )
< “Yellow” , “Banana” > ( 33 )
< “Green” , “Pear” > ( 79 )
< “Orange” , “Tangerine” > ( 12 )
< “Pink” , ”Grapefruit > ( 10 )
< “Blue”, “Berry” > ( 25 )
< “Purple” , “Raisin” > ( 31 )

| a) | Red       | b) | Grapefruit -> | Apple  |
|    | Yellow    |    | Berry        | Pear   |
|    | Green     |    | Raisin       |        |
|    | Orange    |    | Tangerine    |        |
|    | Pink      |    | Banana       |        |
| d) |          |    |              |        |

| d) |          |    |              |        |
|    | Grapefruit -> | Apple  |
|    | Berry        | Pear   |
|    | Raisin       |        |
|    | Tangerine    |        |
|    | Banana       |        |
|    | Apple        |        |
|    | -> Pear      |        |

| e) |          |    |              |        |
|    | Apple ->   |    |              |        |
|    | Berry       |    |              |        |
|    | Banana ->  |    |              |        |
|    | Raisin      |    |              |        |
|    | Pear        |    |              |        |
|    | Tangerine   |    |              |        |
|    | Grapefruit  |    |              |        |

| c) |          |    |              |        |
|    | Pink ->   |    |              |        |
|    | Blue       |    |              |        |
|    | Purple     |    |              |        |
|    | Orange     |    |              |        |
|    | Yellow     |    |              |        |
|    | Red ->    |    |              |        |
|    | Green      |    |              |        |
The following method “isSorted” should return true if the array “x” is sorted in descending order. Otherwise, the method should return false:

```java
public static Boolean isSorted(int[] x) {
    // missing code
}
```

Which of the following code fragments is the missing code?

a) boolean b = true;
   for (int i = 0; i < x.length - 1; i++) {
       if (x[i] < x[i + 1])
           b = false;
       else
           b = true;
   } return b;

b) for (int i = 0; i < x.length - 1; i++) {
    if (x[i] < x[i + 1])
        return false;
} return true;

c) boolean b = false; // always returns
   for (int i = 0; i < x.length - 1; i++) {
       if (x[i] < x[i + 1])
           b = false;
   } return b;

d) boolean b = false; // will return true
   for (int i = 0; i < x.length - 1; i++) {
       if (x[i] < x[i + 1])
           b = true;
   } return b;

e) for (int i = 0; i < x.length - 1; i++) {
    if (x[i] < x[i + 1])
        return true;
} return false;
public class Trace {
    private int myNumber;

    public Trace ( int n ) {
        this.myNumber = n;
    }

    public int getMyNumber() {
        return myNumber;
    }

    public void setMyNumber( int n ) {
        this.myNumber = n;
    }

    public String toString() {
        return "My number is " + getMyNumber();
    }

    public static void main(String[] args) {
        Trace a = new Trace( 4 );
        Trace b = new Trace( 7 );
        Trace c = new Trace( 2 );

        Trace[] myArray = new Trace[3];
        myArray[0] = c;
        myArray[1] = a;
        myArray[2] = b;

        for( int i = 0; i < myArray.length ;i++ ) {
            System.out.println( myArray[i].getMyNumber() );
        }

        b.setMyNumber( a.getMyNumber() );
        a.setMyNumber( c.getMyNumber() );

        System.out.println( b );
        System.out.println( myArray[ 0 ] );
    }
}
What is the output of the Trace class on the preceding page?

a) 4
   7
   2
   4
   4

b) 4
   7
   2
   My number is 4
   My number is 2

c) 2
   4
   7
   My number is 4
   My number is 4

d) 2
   4
   7
   My number is 4
   My number is 2

e) My number is 2
   My number is 4
   My number is 7
   4
   2
int[] x = { 3, 2, 5, 6, 8, 4 };  
int limit = 11;  
int i = 0;  

while (0 < limit && i < x.length)  
{
    limit -= x[i];  
i++;  
}

What is the value of the variable "i" after the code is executed?

a) 0  
b) 3  
c) 4  
d) 5  
e) -5
public int Eval( String s, char c, int value)
{
    if ( s.length == 0 )
        return value;
    else if( s.charAt( 0 ).equals( c ) )
    {
        value = value++;
        return Eval( s.substring(1 ), c , value );
    }
    else
    {
        value = value * 2;
        return Eval( s.substring(1 ), c , value );
    }
}

What is the value returned by this method call?
Eval( “remember”, ‘e’, 1 );

a) 58  
b) 25  
c) 26  
d) 9  
e) 29
POLYMORPHISM

Given the following class hierarchy:

```java
public abstract class Student
    has: public void goesToGT()

public interface Graduate
    has: public void do()

public class GettingOut extends Student implements Graduate
    has: public void happy()
    has: public void do()

public class Alumni extends GettingOut
    has: public void newDo()

public class GradStudent extends Student
    has: public void something()
```

Determine whether the following statements will compile and run without errors.
If no errors write OK.
If compile error write COMP
If run time error write RUN

1) Student t = new Student();
2) Student t = new GettingOut();
3) Student t = new GradStudent();
   t.something();
4) Student t = new Alumni();
   ((GradStudent)t.something();
5) Alumni t = new GettingOut();

<table>
<thead>
<tr>
<th>a)</th>
<th>b)</th>
<th>c)</th>
<th>d)</th>
<th>e)</th>
</tr>
</thead>
<tbody>
<tr>
<td>OK</td>
<td>COMP</td>
<td>OK</td>
<td>OK</td>
<td>COMP</td>
</tr>
<tr>
<td>COMP</td>
<td>COMP</td>
<td>RUN</td>
<td>RUN</td>
<td>OK</td>
</tr>
<tr>
<td>RUN</td>
<td>RUN</td>
<td>OK</td>
<td>COMP</td>
<td>OK</td>
</tr>
<tr>
<td>OK</td>
<td>COMP</td>
<td>OK</td>
<td>OK</td>
<td>COMP</td>
</tr>
</tbody>
</table>
SEARCHING

```java
int array[] = { 13, 11, 10, 8, 5, 7, 3};
int i = 0;

while( ( i < array.length - 1 ) && ( array[ i + 1 ] < array[ i ] ) )
{
    i++;
}
System.out.println( i );
```

What is the output of the code above?

a) 5  
b) 4  
c) 6  
d) 8  
e) 7
DYNAMIC BINDING

public class Object
{
    public void printer()
    {
        System.out.println("Printer method in Object class");
    }
}

public class myObject extends Object
{
    public void printer()
    {
        System.out.println("Printer method in myObject class");
    }
}

public class aObject extends myObject
{
    public void printer()
    {
        System.out.println("Printer method in aObject class");
    }

    public static void main( String[] args )
    {
        Object temp = new Object();
        temp.printer();
        aObject hTemp = new aObject();

        temp = hTemp;
        temp.printer();

        myObject mTemp = new myObject();
        mTemp.printer();

        mTemp = (aObject)temp;
        mTemp.printer();
    }
}
What is the output of the preceding code?

a) Printer method in Object class
   Printer method in Object class
   Printer method in myObject class
   Printer method in myObject class

b) Printer method in Object class
   Printer method in aObject class
   Printer method in Object class
   Printer method in aObject class

c) Printer method in aObject class
   Printer method in Object class
   Printer method in Object class
   Printer method in aObject class

d) Printer method in aObject class
   Printer method in aObject class
   Printer method in myObject class
   Printer method in myObject class

e) Printer method in Object class
   Printer method in aObject class
   Printer method in myObject class
   Printer method in aObject class
public class myFrame extends Frame
{
    public static void main(String argv[])
    {
        myFrame f = new myFrame();
        f.setSize(300,200);
        f.setVisible(true);
    }
}

How would you set the frame surface color to blue?

a) f.setBackground(Color.blue);
b) f.setColor(BLUE);
c) f.Background(blue);
d) f.color=Color.blue;
e) f setColor(Color.blue);
public class ListNode
{
    private ListNode next;
    private String data;
    ...
}

Assume that a linked list exists and that the variable "head" is used to maintain the beginning of the list. Also assume that the variable "position" and "head" are objects of the class ListNode.

ListNode head = new ListNode();
ListNode position = new ListNode();

Which of the following code segments correctly adds "position" to the beginning of the list? The code must maintain the "head" variable as the beginning of the list.

a) position = head;
   head = position;

b) position.next = head.next;
   head = position.next;

c) position.next = head;
   head = position;

d) position = head.next;
   head = position.next;

e) position = head.next;
   head = position;