MULTIPLE-CHOICE QUESTIONS ON ARRAYS AND ARRAY LISTS

1. Which of the following correctly initializes an array `arr` to contain four elements each with value 0?
   I int[] arr = {0, 0, 0, 0};
   II int[] arr = new int[4];
   III int[] arr = new int[4];
      for (int i = 0; i < arr.length; i++)
         arr[i] = 0;

   (A) I only
   (B) III only
   (C) I and III only
   (D) II and III only
   (E) I, II, and III

2. The following program segment is intended to find the index of the first negative integer in `arr[0] ... arr[N-1]`, where `arr` is an array of `N` integers.

   int i = 0;
   while (arr[i] >= 0)
   {
     i++;
   }
   location = i;

   This segment will work as intended
   (A) always.
   (B) never.
   (C) whenever `arr` contains at least one negative integer.
   (D) whenever `arr` contains at least one nonnegative integer.
   (E) whenever `arr` contains no negative integers.

3. Refer to the following code segment. You may assume that `arr` is an array of int values.

   int sum = arr[0], i = 0;
   while (i < arr.length)
   {
     i++;
     sum += arr[i];
   }

   Which of the following will be the result of executing the segment?
   (A) Sum of `arr[0], arr[1], ..., arr[arr.length-1]` will be stored in `sum`.
   (B) Sum of `arr[1], arr[2], ..., arr[arr.length-1]` will be stored in `sum`.
   (C) Sum of `arr[0], arr[1], ..., arr[arr.length]` will be stored in `sum`.
   (D) An infinite loop will occur.
   (E) A run-time error will occur.
4. The following code fragment is intended to find the smallest value in 
arr[0] ... arr[n-1].

```java
// Precondition: arr[0]...arr[n-1] initialized with integers.
// arr is an array, arr.length = n.
// Postcondition: min = smallest value in arr[0]...arr[n-1].
int min = arr[0];
int i = 1;
while (i < n)
{
    i++;
    if (arr[i] < min)
        min = arr[i];
}
```

This code is incorrect. For the segment to work as intended, which of the following modifications could be made?

I Change the line
```java
int i = 1;
```
to
```java
int i = 0;
```
Make no other changes.

II Change the body of the while loop to
```java
{
    if (arr[i] < min)
        min = arr[i];
    i++;
}
```
Make no other changes.

III Change the test for the while loop as follows:
```java
while (i <= n)
```
Make no other changes.

(A) I only
(B) II only
(C) III only
(D) I and II only
(E) I, II, and III

5. If arr contains
```java
int[] arr;
for (int i = 0; i < n; i++)
{
    ...
}
```
which of the following will arr contain?
(A) arr[0]
(B) arr[1]
(C) arr[2]
(D) arr[n]
(E) arr[n+1]

6. The expression
```java
if (arr[i] < min)
```
will be true if
(A) arr[i] > min
(B) arr[i] = min
(C) arr[i] < min
(D) arr[i] < min
(E) arr[i] > min

7. The code fragment above will find the minimum value in an array.

(A) True
(B) False
Questions 5 and 6 refer to the following code segment. You may assume that array arr1 contains elements arr1[0], arr1[1], ..., arr1[N-1], where N = arr1.length.

```java
int count = 0;
for (int i = 0; i < N; i++)
    if (arr1[i] != 0)
        { 
            arr1[count] = arr1[i];
            count++;
        }
int[] arr2 = new int[count];
for (int i = 0; i < count; i++)
    arr2[i] = arr1[i];
```

5. If array arr1 initially contains the elements 0, 6, 0, 4, 0, 0, 2 in this order, what will arr2 contain after execution of the code segment?
   (A) 6, 4, 2
   (B) 0, 0, 0, 6, 4, 2
   (C) 6, 4, 2, 4, 0, 0, 2
   (D) 0, 6, 0, 4, 0, 0, 2
   (E) 6, 4, 2, 0, 0, 0, 0.

6. The algorithm has run time
   (A) O(N^2)
   (B) O(N)
   (C) O(1)
   (D) O(log N)
   (E) O(N log N)

7. Consider this program segment:

```java
for (int i = 2; i <= k; i++)
    if (arr[i] < someValue)
        System.out.print("SMALL");
```

What is the maximum number of times that SMALL can be printed?
   (A) 0
   (B) 1
   (C) k - 1
   (D) k - 2
   (E) k
8. What will be output from the following code segment, assuming it is in the same class as the doSomething method?

```java
int[] arr = {1, 2, 3, 4};
doSomething(arr);
System.out.print(arr[1] + " ");
System.out.print(arr[3]);
...

public void doSomething(int[] list)
{
    int[] b = list;
    for (int i = 0; i < b.length; i++)
        b[i] = i;
}
```

(A) 0 0  
(B) 2 4  
(C) 1 3  
(D) 0 2  
(E) 0 3

9. Consider writing a program that reads the lines of any text file into a sequential list of lines. Which of the following is a good reason to implement the list with an ArrayList of String objects rather than an array of String objects?

(A) The get and set methods of ArrayList are more convenient than the [] notation for arrays.  
(B) The size method of ArrayList provides instant access to the length of the list.  
(C) An ArrayList can contain objects of any type, which leads to greater generality.  
(D) If any particular text file is unexpectedly long, the ArrayList will automatically be resized. The array, by contrast, may go out of bounds.  
(E) The String methods are easier to use with an ArrayList than with an array.

10. Consider writing a program that produces statistics for long lists of numerical data. Which of the following is the best reason to implement each list with an array of int (or double), rather than an ArrayList of Integer (or Double) objects?

(A) An array of primitive number types is more efficient to manipulate than an ArrayList of wrapper objects that contain numbers.  
(B) Insertion of new elements into a list is easier to code for an array than for an ArrayList.  
(C) Removal of elements from a list is easier to code for an array than for an ArrayList.  
(D) Accessing individual elements in the middle of a list is easier for an array than for an ArrayList.  
(E) Accessing all the elements is more efficient in an array than in an ArrayList.
Refer to the following classes for Questions 11-14.

```java
public class Address
{
    private String myName;
    private String myStreet;
    private String myCity;
    private String myState;
    private String myZip;

    //constructors
    ...

    //accessors
    public String getName()
    { return myName; }
    public String getStreet()
    { return myStreet; }
    public String getCity()
    { return myCity; }
    public String getState()
    { return myState; }
    public String getZip()
    { return myZip; }
}
```

```java
public class Student
{
    private int idNum;
    private double gpa;
    private Address myAddress;

    //constructors
    ...

    //accessors
    public Address getAddress()
    { return myAddress; }
    public int getIdNum()
    { return idNum; }
    public double getGpa()
    { return gpa; }
}
```
11. A client method has this declaration, followed by code to initialize the list:

```java
Address[] list = new Address[100];
```

Here is a code segment to generate a list of *names only*.

```java
for (Address a : list)
    /* line of code */
```

Which is a correct /* line of code */?

- (A) `System.out.println(Address[i].getName());`
- (B) `System.out.println(list[i].getName());`
- (C) `System.out.println(a[i].getName());`
- (D) `System.out.println(a.getName());`
- (E) `System.out.println(list.getName());`

12. The following code segment is to print out a list of addresses:

```java
for (Address addr : list)
{
    /* more code */
}
```

Which is a correct replacement for /* more code */?

- I
  ```java
  System.out.println(list[i].getName());
  System.out.println(list[i].getStreet());
  System.out.println(list[i].getCity() + ", ");
  System.out.println(list[i].getState() + " ");
  System.out.println(list[i].getZip());
  ```
- II
  ```java
  System.out.println(addr.getName());
  System.out.println(addr.getStreet());
  System.out.println(addr.getCity() + " ");
  System.out.println(addr.getState() + " ");
  System.out.println(addr.getZip());
  ```
- III
  ```java
  System.out.println(addr);
  ```

- (A) I only
- (B) II only
- (C) III only
- (D) I and II only
- (E) I, II, and III
13. A client method has this declaration:

   Student[] allStudents = new Student[NUM_STUDS]; //NUM_STUDS is
   //an int constant

Here is a code segment to generate a list of Student names only. (You may assume
that allStudents has been initialized.)

   for (Student student : allStudents)
       /* code to print list of names * /

Which is a correct replacement for /* code to print list of names */?
(A) System.out.println(allStudents.getName());
(B) System.out.println(student.getName());
(C) System.out.println(student.getAddress().getName());
(D) System.out.println(allStudents.getAddress().getName());
(E) System.out.println(student[i].getName());
14. Here is a method that locates the Student with the highest idNum:

```java
//Precondition: Array stuArr of Student is initialized.
//Postcondition: Student with highest idNum has been returned.
public static Student locate(Student[] stuArr)
{
    /* method body */
}
```

Which of the following could replace /* method body */ so that the method works as intended?

I int max = stuArr[0].getIdNum();
    for (Student student : stuArr)
        if (student.getIdNum() > max)
            { max = student.getIdNum();
              return student;
        }
    return stuArr[0];

II Student highestSoFar = stuArr[0];
    int max = stuArr[0].getIdNum();
    for (Student student : stuArr)
        if (student.getIdNum() > max)
            { max = student.getIdNum();
              highestSoFar = student;
            }
    return highestSoFar;

III int maxPos = 0;
    for (int i = 1; i < stuArr.length; i++)
        if (stuArr[i].getIdNum() > stuArr[maxPos].getIdNum())
            maxPos = i;
    return stuArr[maxPos];

(A) I only
(B) II only
(C) III only
(D) I and III only
(E) II and III only
Multiple Choice Questions on Arrays and Array Lists

Questions 15-17 refer to the Ticket and Transaction classes below.

```java
public class Ticket {
    private String myRow;
    private int mySeat;
    private double myPrice;

    //constructor
    public Ticket(String row, int seat, double price) {
        myRow = row;
        mySeat = seat;
        myPrice = price;
    }

    //accessors getRow(), getSeat(), and getPrice()
    ...
}

public class Transaction {
    private int myNumTickets;
    private Ticket[] tickList;

    //constructor
    public Transaction(int numTicks) {
        myNumTickets = numTicks;
        tickList = new Ticket[numTicks];
        String row;
        int seat;
        double price;
        for (int i = 0; i < numTicks; i++) {
            //read user input for row, seat, and price
            ...

            /* more code */
        }
    }

    //Returns total amount paid for this transaction.
    public double totalPaid() {
        double total = 0.0;
        /* code to calculate amount */
        return total;
    }
}
```
15. Which of the following correctly replaces /* more code */ in the Transaction constructor to initialize the tickList array?

(A) tickList[i] = new Ticket(getRow(), getSeat(), getPrice());
(B) tickList[i] = new Ticket(row, seat, price);
(C) tickList[i] = new tickList(getRow(), getSeat(), getPrice());
(D) tickList[i] = new tickList(row, seat, price);
(E) tickList[i] = new tickList(numTicks);

16. Which represents correct /* code to calculate amount */ in the totalPaid method?

(A) for (Ticket t : tickList)
    total += t.myPrice;
(B) for (Ticket t : tickList)
    total += tickList.getPrice();
(C) for (Ticket t : tickList)
    total += t.getPrice();
(D) Transaction T;
    for (Ticket t : T)
    total += t.getPrice();
(E) Transaction T;
    for (Ticket t : T)
    total += t.myPrice;

17. Suppose it is necessary to keep a list of all ticket transactions. A suitable declaration would be

(A) Transaction[] listOfSales = new Transaction[NUMSALES];
(B) Transaction[] listOfSales = new Ticket[NUMSALES];
(C) Ticket[] listOfSales = new Transaction[NUMSALES];
(D) Ticket[] listOfSales = new Ticket[NUMSALES];
(E) Transaction[] Ticket = new listOfSales[NUMSALES];
18. Refer to method match below:

```java
// Precondition: v[0]..v[N-1] and w[0]..w[M-1] initialized with
// integers. v[0] < v[1] < ... < v[N-1] and
// w[0] < w[1] < ... < w[M-1].
// Postcondition: Returns true if there is an integer k that occurs
// in both arrays, otherwise returns false.
public static boolean match(int[] v, int[] w, int N, int M)
{
    int vIndex = 0, wIndex = 0;
    while (vIndex < N && wIndex < M)
    {
        if (v[vIndex] == w[wIndex])
            return true;
        else if (v[vIndex] < w[wIndex])
            vIndex++;
        else
            wIndex++;
    }
    return false;
}
```

Assuming that the method has not been exited, which assertion is true at the end of every execution of the while loop?

(A) v[0]..v[vIndex-1] and w[0]..w[wIndex-1] contain no common value,
    vIndex ≤ N and wIndex ≤ M.
(B) v[0]..v[vIndex] and w[0]..w[wIndex] contain no common value,
    vIndex ≤ N and wIndex ≤ M.
(C) v[0]..v[vIndex-1] and w[0]..w[wIndex-1] contain no common value,
    vIndex ≤ N-1 and wIndex ≤ M-1.
(D) v[0]..v[vIndex] and w[0]..w[wIndex] contain no common value,
    vIndex ≤ N-1 and wIndex ≤ M-1.
(E) v[0]..v[N-1] and w[0]..w[M-1] contain no common value,
    vIndex ≤ N and wIndex ≤ M.
19. Consider this class:

```java
public class Book{

    private String myTitle;
    private String myAuthor;
    private boolean myCheckoutStatus;

    //constructor
    public Book(String title, String author){
        myTitle = title;
        myAuthor = author;
        myCheckoutStatus = false;
    }

    //Change checkout status.
    public void changeStatus()
    { myCheckoutStatus = !myCheckoutStatus; }

    //other methods not shown ...
}
```

A client program has this declaration:

```java
Book[] bookList = new Book[SOME_NUMBER];
```

Suppose `bookList` is initialized so that each `Book` in the list has a title, author, and checkout status. The following piece of code is written, whose intent is to change the checkout status of each book in `bookList`.

```java
for (Book b : bookList)
    b.changeStatus();
```

Which is true about this code?

(A) The `bookList` array will remain unchanged after execution.
(B) Each book in the `bookList` array will have its checkout status changed, as intended.
(C) A `NullPointerException` may occur.
(D) A run-time error will occur because it is not possible to modify objects using the for-each loop.
(E) A logic error will occur because it is not possible to modify objects in an array without accessing the indexes of the objects.
Consider this class for Questions 20 and 21:

```java
public class BingoCard
{
    private int[] myCard;

    /* Default constructor: Creates BingoCard with
    * 20 random digits in the range 1 - 90. */
    public BingoCard()
    { /* implementation not shown */ }

    /* Display BingoCard. */
    public void display()
    { /* implementation not shown */ }
}
```

A program that simulates a bingo game declares an array of BingoCard. The array has NUMPLAYERS elements, where each element represents the card of a different player. Here is a code segment that creates all the bingo cards in the game:

```java
/* declare array of BingoCard */
/* construct each BingoCard */
```

20. Which of the following is a correct replacement for

```java
/* declare array of BingoCard */
```

(A) int[] BingoCard = new BingoCard[NUMPLAYERS];
(B) BingoCard[] players = new int[NUMPLAYERS];
(C) BingoCard[] players = new BingoCard[20];
(D) BingoCard[] players = new BingoCard[NUMPLAYERS];
(E) int[] players = new BingoCard[NUMPLAYERS];

21. Assuming that players has been declared as an array of BingoCard, which of the following is a correct replacement for

```java
/* construct each BingoCard */
```

I for (BingoCard card : players)
    card = new BingoCard();

II for (BingoCard card : players)
    players[card] = new BingoCard();

III for (int i = 0; i < players.length; i++)
    players[i] = new BingoCard();

(A) I only
(B) II only
(C) III only
(D) I and III only
(E) I, II, and III
22. Which declaration will cause an error?

I. `ArrayList<String> stringList = new ArrayList<String>();`

II. `ArrayList<Integer> intList = new ArrayList<Integer>();`

III. `ArrayList<Comparable> compList = new ArrayList<Comparable>();`

(A) I only
(B) II only
(C) III only
(D) I and III only
(E) II and III only

23. Consider these declarations:

```java
ArrayList<String> stringList = new ArrayList<String>();
String ch = " ";
Integer intOb = new Integer(5);
```

Which statement will cause an error?

(A) `stringList.add(ch);`
(B) `stringList.add(new String("handy andy"));`
(C) `stringList.add(intOb.toString());`
(D) `stringList.add(ch + 8);`
(E) `stringList.add(intOb + 8);`

24. Let `list` be an `ArrayList<Integer>` containing these elements:

```
2 5 7 6 0 1
```

Which of the following statements would *not* cause an error to occur? Assume that each statement applies to the given list, independent of the other statements.

(A) `Object ob = list.get(6);`
(B) `Integer intOb = list.add(3.4);`
(C) `list.add(5, 9);`
(D) `Object x = list.remove(6);`
(E) `Object y = list.set(6, 8);`
28. A boolean method find is added to the Purse class:

```java
/* Returns true if the purse has a coin that matches aCoin, */
public boolean find(Coin aCoin)
{
    for (Coin c : coins)
    {
        /* code to find match */
    }
    return false;
}
```

Which is a correct replacement for /* code to find match */?

I if (c.equals(aCoin))
    return true;

II if ((c.getName()).equals(aCoin.getName()))
    return true;

III if ((c.getValue()).equals(aCoin.getValue()))
    return true;

(A) I only
(B) II only
(C) III only
(D) I and II only
(E) I, II, and III

29. Which of the following initializes an 8 × 10 matrix with integer values that are perfect squares? (0 is a perfect square.)

I int[][] mat = new int[8][10];

II int[][] mat = new int[8][10];
    for (int r = 0; r < mat.length; r++)
        for (int c = 0; c < mat[r].length; c++)
            mat[r][c] = r * r;

III int[][] mat = new int[8][10];
    for (int r = 0; r < mat.length; r++)
        for (int c = 0; c < mat[r].length; r++)
            mat[r][c] = c = c;

(A) I only
(B) II only
(C) III only
(D) I and II only
(E) I, II, and III
### ANSWER KEY

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4. B</td>
<td>17. A</td>
<td>30. A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8. C</td>
<td>21. C</td>
<td>34. D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10. A</td>
<td>23. E</td>
<td>36. E</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### ANSWERS EXPLAINED

1. (E) Segment I is an initializer list which is equivalent to

```java
int[] arr = new int[4];
arr[0] = 0;
arr[1] = 0;
arr[2] = 0;
arr[3] = 0;
```

Segment II creates four slots for integers, which by default are initialized to 0. The for loop in segment III is therefore unnecessary. It is not, however, incorrect.

2. (C) If `arr` contains no negative integers, the value of `i` will eventually exceed `N-1`, and `arr[i]` will cause an `ArrayIndexOutOfBoundsException` to be thrown.

3. (E) The intent is to sum elements `arr[0]`, `arr[1]`, ..., `arr[arr.length-1]`. Notice, however, that when `i` has the value `arr.length-1`, it is incremented to `arr.length` in the loop, so the statement `sum += arr[i]` uses `arr[arr.length]`, which is out of range.

4. (B) There are two problems with the segment as given:

   1. `arr[1]` is not tested.
   2. When `i` has a value of `n-1`, incrementing `i` will lead to an out-of-range error for the `if(arr[i] < min)` test.

Modification II corrects both these errors. The change suggested in III corrects neither of these errors. The change in I corrects (1) but not (2).

5. (A) The code segment has the effect of removing all occurrences of 0 from array `arr1`. Then the nonzero elements are transferred to array `arr2`. 
6. (B) The algorithm is linear. It passes once through the array, making a single assignment if a nonzero element is found.

7. (C) If \( a[i] < \text{someValue} \) for all \( i \) from 2 to \( k \), SMALL will be printed on each iteration of the for loop. Since there are \( k-1 \) iterations, the maximum number of times that SMALL can be printed is \( k-1 \).

8. (C) Array \( arr \) is changed by \text{doSomething}. Here are the memory slots:

   Just before \text{doSomething} is called:
   
   \[
   \begin{array}{c}
   \text{arr} \\
   \rightarrow \begin{array}{cccc}
   1 & 2 & 3 & 4 \\
   \end{array}
   \end{array}
   \]

   Just after \text{doSomething} is called, but before the for loop is executed:
   
   \[
   \begin{array}{c}
   \text{arr} \\
   \rightarrow \begin{array}{cccc}
   1 & 2 & 3 & 4 \\
   \end{array}
   \end{array}
   \]

   Just before exiting \text{doSomething}:
   
   \[
   \begin{array}{c}
   \text{arr} \\
   \rightarrow \begin{array}{cccc}
   0 & 1 & 2 & 3 \\
   \end{array}
   \end{array}
   \]

   Just after exiting \text{doSomething}:
   
   \[
   \begin{array}{c}
   \text{arr} \\
   \rightarrow \begin{array}{cccc}
   0 & 1 & 2 & 3 \\
   \end{array}
   \end{array}
   \]

9. (D) Arrays are of fixed length and do not shrink or grow if the size of the data set varies. An \text{ArrayList} automatically resizes the list. Choice A is false: The \( [] \) notation is compact and easy to use. Choice B is not a valid reason because an array \( arr \) also provides instant access to its length with the quantity \( arr.length \). Choice C is invalid because an array can also contain objects. Also, generality is beside the point in the given program: The list \text{must} hold \text{String} objects. Choice E is false: Whether a \text{String} object is \( arr[i] \) or \( list.get(i) \), the \text{String} methods are equally easy to invoke.

10. (A) In order for numerical elements to be added to an \text{ArrayList}, each element must be wrapped in a wrapper class before insertion into the list. Then, to retrieve a numerical value from the \text{ArrayList}, the element must be unboxed using the \text{intValue} or \text{doubleValue} methods. Even though these operations can be taken care of with auto-boxing and unboxing, there are efficiency costs. In an array, you simply use the \( [] \) notation for assignment (as in \( arr[i] = \text{num} \)) or retrieval (\( \text{value} = arr[i] \)). Note that choices B and C are false statements: Both insertion and deletion for an array involve writing code to shift elements. An \text{ArrayList} automatically takes care of this through its \text{add} and \text{remove} methods. Choice D is a poor reason for choosing an array. While the get and set methods of \text{ArrayList} might be slightly more awkward than using the \( [] \) notation, both mechanisms work pretty easily. Choice E is false: Efficiency of access is roughly the same.

11. (D) For each \text{Address} object \( a \) in \text{list}, access the name of the object with \( a.getName() \).
12. (B) Since the Address class does not have a toString method, each data field must explicitly be printed. Segment III would work if there were a toString method for the class (but there isn’t, so it doesn’t!). Segment I fails because of incorrect use of the for-each loop: The array index should not be accessed.

13. (C) Each Student name must be accessed through the Address class accessor getName(). The expression student.getAddress() accesses the entire address of that student. The myName field is then accessed using the getName() accessor of the Address class.

14. (E) Both correct solutions are careful not to lose the student who has the highest idNum so far. Segment II does it by storing a reference to the student whose idNum is greater than max, not necessarily the student with the highest idNum in the list.

15. (B) For each i, tickList[i] is a new Ticket object that must be constructed using the Ticket constructor. Therefore eliminate choices C, D, and E. Choice A is wrong because getRow(), getSeat(), and getPrice() are accessors for values that already exist for some Ticket object. Note also the absence of the do member construct.

16. (C) To access the price for each Ticket in the tickList array, the getPrice() accessor in the Ticket class must be used, since myPrice is private to that class. This eliminates choices A and E. Choice B uses the array name incorrectly. Choices D and E incorrectly declare a Transaction object. (The method applies to an existing Transaction object.)

17. (A) An array of type Transaction is required. This eliminates choices C and D. Additionally, choices B and D incorrectly use type Ticket on the right-hand side. Choice E puts the identifier listOfSales in the wrong place.

18. (A) Notice that either vIndex or wIndex is incremented at the end of the loop. This means that, when the loop is exited, the current values of v[vIndex] and w[wIndex] have not been compared. Therefore, you can only make an assertion for values v[0] . . . v[vIndex-1] and w[0] . . . w[wIndex-1]. Also, notice that if there is no common value in the arrays, the exiting condition for the while loop will be that the end of one of the arrays has been reached, namely vIndex equals N or wIndex equals M.

19. (B) Objects in an array can be changed in a for-each loop by using mutator methods of the objects’ class. The changeStatus method, a mutator in the Book class, will work as intended in the given code. Choice C would be true if it were not given that each Book in bookList was initialized. If any given b had a value of null, then a NullPointerException would be thrown.

20. (D) The declaration must start with the type of value in the array, namely BingoCard. This eliminates choices A and E. Eliminate choice B: The type on the right of the assignment should be BingoCard. Choice C is wrong because the number of slots in the array should be NUMPLAYERS, not 20.

21. (C) Segment III is the only segment that works, since the for-each loop cannot be used to replace elements in an array. After the declaration

```java
BingoCard[] players = new BingoCard[NUMPLAYERS];
```

each element in the players array is null. The intent in the given code is to
22. (B) The type parameter in a generic `ArrayList` must be a class type, not a primitive.

23. (E) All elements added to `strList` must be of type `String`. Each choice satisfies this except choice E. Note that in choice D, since `ch` is a `String`, the expression `ch + 8` becomes a `String` (just one of the operands needs to be a `String` to convert the whole expression to a `String`). In choice E, neither `int` nor `8` is a `String`.

24. (C) The effect of choice C is to adjust the size of the list to 7 and to add the `Integer` 9 to the last slot (i.e., the slot with index 6). Choices A, D, and E will all cause an `IndexOutOfBoundsException` because there is no slot with index 6; the last slot has index 5. Choice B will cause a compile-time error, since it is attempting to add an element of type `Double` to a list of type `Integer`.

25. (D) If `element` is smaller than the last item in the list, it will be compared with every item in the list. Eventually `index` will be incremented to a value that is out of bounds. To avoid this error, the test in the `while` loop should be

```java
while (index < list.size() &&
       element.compareTo(list.get(index)) < 0)
```

Notice that if `element` is greater than or equal to at least one item in `list`, the test as given in the problem will eventually be false, preventing an out-of-range error.

26. (A) Recall that `add(index, obj)` shifts all elements, starting at `index`, one unit to the right, then inserts `obj` at position `index`. The `set(index, obj)` method replaces the element in position `index` with `obj`. So here is the state of `list` after each change:

```
<table>
<thead>
<tr>
<th>i</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6</td>
<td>0</td>
<td>1</td>
<td>8</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>6</td>
<td>0</td>
<td>1</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>8</td>
<td></td>
</tr>
</tbody>
</table>
```

27. (E) The value of each `Coin` in `coins` must be accessed with `c.getValue()`. This eliminates choice D. Eliminate choices A and B: The loop accesses each `Coin` in the `coins ArrayList`, which means that there should not be any statements attempting to get the next `Coin`. Choice B would be correct if the first statement in the loop body were

```java
double value = c.getValue();
```

28. (D) The `equals` method is defined for objects only. Since `getValue` returns a `double`, the quantities `c.getValue()` and `aCoin.getValue()` must be compared either using `==`, or as described in the box on p. 122 (better).

29. (D) Segment II is the straightforward solution. Segment I is correct because it initializes all slots of the matrix to 0, a perfect square. (By default, all arrays of `int` or `double` are initialized to 0.) Segment III fails because `r` is undefined in the condition `c < mat[r].length`. In order to do a column-by-column traversal, you need to get the number of columns in each row. The outer for loop could be

```java
for (int c = 0; c < mat[0].length; c++)
```