Name: ____ AP Chapter 5 Review

1. Refer to method **match** below:

```
/* @param v an array of int sorted in increasing order
   @param w an array of int sorted in increasing order
   @param N the number of elements in array v
   @param M the number of elements in array w
   @return ture if there is an integer k that occurs in both arrays; otherwise returns false
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] */
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.

2. Let list be an ArrayList<Integer> containing the 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(6, 9);`
d. `Object x = list.remove(6);`
e. `Object y = list.set(6, 8);`

3. Consider this class:

```
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:

```
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.

```
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.

4. Consider this program segment:

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

5. 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 store 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.

6. Refer to the following code segment. You may assume that array arr1 contains elements arr1[0], arr1[1], …, arr[N-1], where N = arr1.length.

```
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];
```

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, 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

7. Consider a class MatrixStuff that has a private instance variable:

```
private int[][] mat;
```

Refer to method alter below that occurs in the MatrixStuff class. (The lines are numbered for reference)

```
1//precondition: mat is initialized with integers
2//postcondition: column c has been removed and the
3//         last column is filled with zeros
4public void alter(int[][] mat, int c)
5{
6     for(int i = 0; i < mat.length; i++)
7           for(int j = c; j < mat[0].length; j++)
8                 mat[i][j] = mat[i][j+1];
9     //code to insert zeros in rightmost column
10     . . .
11}
```

The intent of the method alter is to remove column c. Thus, if the input matrix mat is

```
2 6 8 9
1 5 4 3
0 7 3 2
```

The method call alter(mat, 1) should change mat to

```
2 8 9 0
1 4 3 0
0 3 2 0
```

The method does not work as intended. Which of the following changes will correct the problem?

I. Change line 7 to
```
for(int j = c; j < mat[0].length - 1; j++)
```
And make no other changes

II. Change lines 7 and 8 to
```
for(int j = c + 1; j < mat[0].length; j++)
      mat[i][j-1] = mat[i][j];
```
And make no other changes

III. Change lines 7 and 8 to
```
for(int j = mat[0].length - 1; j > c; j--)
      mat[i][j-1] = mat[i][j];
```

And make no other changes

a. I only
b. II only
c. III only
d. I and II only
e. I, II, and III

8. This question refers to the following method:

```
public static boolean isThere(String[][] mat, int row, int col, String symbol)
{
        boolean yes;
        int i, count =0;
        for(i=0; i<SIZE; i++)
                if(mat[i][col].equals(symbol))
                        count++;
        yes = (count == SIZE);
        count = 0;
        for(i = 0; i<SIZE; i++)
                if(mat[row][i].equals(symbol))
                        count++;
        return (yes || count == SIZE);
}
```

Now consider this code segment:

```
public final int SIZE = 8;
String[][] mat = new String[SIZE][SIZE];
```

Which of the following conditions on a matrix mat of the type declared in the code segment will by itself guarantee that

isThere(mat, 2, 2, “$”)

will have the value true when evaluated?

I. The element in row 2 and column 2 is “$”
II. All elements in both diagonals are “$”
III. All elements in column 2 are “$”

a. I only
b. III only
c. I and II only
d. I and III only
e. II and III only

9. 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.

10. The following code fragment is intended to find the smallest value in arr[0]…arr[n – 1].

```
//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];
}
```

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

I. Change the line

```
int i = 1;
```

To

```
int i = 0;
```

Make no other changes

II. Change the body of the while loop to

```
{
      if(arr[i] < min)
            min = arr[i];
      i++;
}
```

Make no other changes

III. Change the test for the while loop to:

```
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

Refer to the following class for Questions 11-14

```
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;}
}

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:

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

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

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

Which is a correct replacement for /*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:

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

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

I.
```
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.
```
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.
```
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].getAddress().getName());`

14. Here is a method that locates the Student with the id number:

```
//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

15. Refer to the method insert below:

```
/*Precondition: List list is an ArrayList that contains
 *          Comparable values sorted in decreasing order
 * Postcondition: Element inserted in its correct position
 *          in list
 */
public void insert(List<Comparable> list, Comparable element)
{
      int index = 0;
      while(elemeent.compareTo(list.get(index)) < 0)
            index++;
      list.add(index, element);
}
```

Assuming that the type of element is compatible with the objects in the list, which is a true statement about the insert method?

a. It works as intended for all values of element.
b. It fails for all values of element
c. It fails if element is greater than the first item in list and works in all other cases.
d. It fails if element is smaller than the last item in list and works in all other cases.
e. It fails if element is either greater than the first item or smaller than the last item in list and works in all other cases.

16. Which of the following initializes an 8 x 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 c = 0; c < mat[r].length; c++)
      for(int r = 0; r < mat.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

Questions 17 and 18 are based on the Coin and Purse classes given below:

```
public class Coin
{
      private double myValue;
      private String myName;

      public Coin(double value, String name)
      {
            myValue = value;
            myName = name;
      }

      public double getValue()
      {
            return myValue;
      }

      public String getName();
      {
            return myName;
      }

      //define equals method for Coin objects
      public boolean equals(Object obj)
      {
            /*implementation not shown*/
      }
      //other methods not shown
}

public class Purse
{
      private List<Coin> coins;

      public Purse()
      {
            coins = new ArrayList<Coin>();
      }

      public void add(Coin aCoin)
      {
            coins.add(aCoin);
      }

      public double getTotal()
      {
            /*implemenation not shown*/
      }
}
```

17. Here is the getTotal method from the Purse class:

```
//returns total value of coins in purse
public double getTotal()
{
      double total = 0;
      /*more code*/
      return total;
}
```

Which of the following is a correct replacement for /*more code*/?

a.
```
for(Coin c : coins)
{     c = coins.get(i);
      total += c.getValue();
}
```

b.
```
for(Coin c : coins)
{     Coin value = c.getValue();
      total += value;
}
```

c.
```
for(Coin c : coins)
{     Coin c = coins.get(i);
      total += c.getValue();
}
```

d.
```
for(Coin c: coins)
      total += coins.getValue();
```

e.
```
for(Coin c : coins)
      total += c.getValue();
```

18. A boolean method find is added to the Purse class:

```
/*returns true if the purse has a coin that matches aCoin, false otherwise*/
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

19. Consider the following method that will alter the matrix mat:

```
//precondition: mat is initialized
public static void matStuff(int[][] mat, int row)
{
      int numCols = mat[0].length;
      for(int col = 0; col < numCols; col++)
            mat[row][col] = row;
}
```

Suppose mat is originally

```
            1 4 9 0
            2 7 8 6
            5 1 4 3
```

After the method call matStuff(mat, 2), matrix mat will be

a. 1 4 9 0
   2 7 8 6
   2 2 2 2

b. 1 4 9 0
   2 2 2 2
   5 1 4 3

c. 2 2 2 2
   2 2 2 2
   2 2 2 2

d. 1 4 2 0
   2 7 2 6
   5 1 2 3

e. 1 2 9 0
   2 2 8 6
   5 2 4 3

20. Consider a Clown class that has a default constructor. Suppose a list ArrayList<Clown> list is initialized. Which of the following will **not** cause an IndexOutOfBoundsException to be thrown?

a. for(int i=0; i<=list.size(); i++)
   list.set(i, new Clown());
b. list.add(list.size(), new Clown());
c. Clown c = list.get(list.size());
d. Clown c = list.remove(list.size());
e. List.add(-1, new Clown());

21. Assume that a square matrix mat is defined by

```
int[][] mat = new int[SIZE][SIZE];
//SIZE is an integer constant >=2
```

What does the following code segment do?

```
for(int i=0; i< SIZE -1; i++)
        for(int j=0; j < SIZE – i – 1; j++)
                swap(mat, i, j, SIZE – j – 1, SIZE – i – 1);
```

You may assume the existence of this method:

```
/** Interchange mat[a][b] and mat[c][d]. */
public void swap(int[][] mat, int a, int b, int c, int d)
```

a. Reflects mat through its major diagonal. For example,

$$\begin{matrix} 2 & 6 \\ 4 & 3 \end{matrix} \rightarrow \begin{matrix} 2 & 4 \\ 6 & 3 \end{matrix}$$

b. Reflects mat through its minor diagonal. For example,

$$\begin{matrix} 2 & 6 \\ 4 & 3 \end{matrix} \rightarrow \begin{matrix} 3 & 6 \\ 4 & 2 \end{matrix}$$

c. Reflects mat through a horizontal line of symmetry. For example,

$$\begin{matrix} 2 & 6 \\ 4 & 3 \end{matrix} \rightarrow \begin{matrix} 4 & 3 \\ 2 & 6 \end{matrix}$$

d. Reflects mat through a vertical line of symmetry. For example,

$$\begin{matrix} 2 & 6 \\ 4 & 3 \end{matrix} \rightarrow \begin{matrix} 6 & 2 \\ 3 & 4 \end{matrix}$$

e. Leaves mat unchanged.

22. The method changeNegs below should replace every occurrence of a negative integer in its matrix parameter with 0.

```
/* @param mat the matrix
   Precondition: mat is initialized with integers.
   Postcondition: All negative values in mat replaced with 0.  */

public static void changeNegs(int[][] mat)
{  /*code */ }
```

Which is correct replacement for /*code */?

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

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

III.
```
for(int[] row: mat)
        for(int element: row)
                if(element < 0)
                        element = 0;
```

a. I only
b. II only
c. III only
d. I and II only
e. I, II, and III

23. Consider the following instance variable and method.

```
private int[] nums;
//Precondition: nums contains int values in no particular order

public int getValue()
{
        for(int k=0; k<nums.length; k++)
        {
                if(nums[k] % 2 != 0)
                        return k;
        }
        return -1;
}
```

Suppose the following statement is executed:

```
        int j= getValue();
```

If the value returned in j is a positive integer, which of the following best describes the contents of nums?

a. The only odd int in nums is at position j.
b. All values in positions 0 through j-1 are odd.
c. All values in positions 0 through j-1 are even.
d. All values in positions nums.length -1 down to j+1 are odd
e. All values in positions nums.length -1 down to j+1 are even.

24. Refer to the following class, containing the mystery method.

```
public class SomeClass
{
        private int[] arr;

        /*Constructor.  Initializes arr to contain nonnegative integers k such that 0 <= k < = 9 */
        public SomeClass()
        {        /*implementation not shown */ }

        public int mystery()
        {
                int value = arr[0];
                for(int i=1; i< arr.length; i++)
                        value = value * 10 + arr[i];
                return value;
        }
}
```

Which best describes what the mystery method does?

a. It sums the elements of arr.
b. It sums the products 10*arr[0]+ 10*arr[1] + … + 10 * arr[arr.length-1].
c. It builds an integer of form $d_1 d_2 d_3 \ldots d_n$ where d_1 = arr[0], d_2 = arr[1], … d_n = arr[arr.length-1].
d. It builds an integer of the form $d_1 d_2 d_3 \ldots d_n$ where d_1 = arr[arr.length-1], d_2 = arr[arr.length-2], … d_n = arr[0].
e. It converts the elements of arr to base-10.

25. A matrix is declared as

```
int [] [] mat = new int[2][3];
```

Consider the following method:

```
public static void changeMatrix(int[][] mat)
{
        for(int r=0; r<mat.length; r++)
                for(int c=0; c<mat[r].length; c++)
                        if(r==c)
                                mat[r][c] = Math.abs(mat[r][c]);
}
```

If mat is initialized to be

```
-1  -2  -6
-2  -4   5
```

Which matrix will be the result of a call to changeMatrix(mat)?

a.
```
 1  -2  -6
-2   4   5
```

b.
```
-1   2  -6
 2  -4   5
```

c.
```
-1  -2  -6
-2  -4  -5
```

d.
```
1  2  -6
2  4   5
```

e.
```
1  2  6
2  4  5
```