12/06/12 APCS – Monta Vista High School ArrayList & Sorting Name_____

1. Describe the action on the ArrayList (or array) elements for the **Merge** SORT.

F	R	O	S	T	Y	S	N	O	W	M	A	N

2. Describe the action on the ArrayList (or array) elements through the larger **for** loop of the **Selection** SORT.

81	23	41	92	16	25	78	34	59

3. Consider the following method.

```
public ArrayList <Integer> workonList (int num)
{
        ArrayList <Integer> sequence = new ArrayList <Integer> ();

        for (int i = 1; i <= num; i++)
                sequence.add(new Integer(i * i + 2));

        return sequence;
}
```

What is printed as a result of the following statement?

```
System.out.println(workonList(6));
```

(A) [3, 8, 15, 24, 35]
(B) [3, 8, 15, 24, 35, 48]
(C) [0, 3, 6, 11, 18, 27]
(D) [3, 6, 11, 18, 27]
(E) [3, 6, 11, 18, 27, 38]

4. Consider the following interface and class declarations.

```
public interface Vehicle
{
        double getMileage();                    //  returns the mileage traveled by this Vehicle
}

public class Fleet
{
        private ArrayList <Vehicle> myVehicles;

        public double getTotalMileage ( )       //  returns the mileage traveled by all vehicles in this Fleet
        {
                double sum = 0.0;

                for (Vehicle v : myVehicles)    //  this for-each loop traverses the Vehicles in the ArrayList
                {
                        sum += /*  expression  */  ;
                }
                return sum;
        }

        //  there may be instance variables, constructors, and methods not shown
}
```

Which of the following can be used to replace **/* expression */** so that **getTotalMileage** returns the total of the miles traveled for all vehicles in the fleet?

(A) **getMileage(v)**
(B) **myVehicles[v].getMileage()**
(C) **Vehicle.get(v).getMileage()**
(D) **myVehicles.get(v).getMileage()**
(E) **v.getMileage()**

5. Consider the following code segment.

```
ArrayList<Integer> list = new ArrayList<Integer>();
list.add(2);
list.add(1);
list.add(0);
int n = list.size();
for (int i = 0; i < n; i++)
{
        int value = list.get(i);
        if (value > 0)
                list.add(0,value);
}
System.out.println(list);
```

What is printed as a result of running the code segment?

(A) [2, 1, 0]
(B) [2, 1, 0, 1, 2]
(C) [2, 1, 0, 2, 1]
(D) [2, 1, 2, 1, 0]
(E) [2, 2, 2, 2, 1, 0]

6. What is the output of the following code segment?

```
ArrayList<String> mylist = new ArrayList<String>();
mylist.add("V");
mylist.add("W");
mylist.add("X");
mylist.add("Y");
mylist.add("Z");

for (int count = 1; count <= 2; count++)
{
        mylist.remove(count);
}
for (int count = 1; count <= 3; count++)
{
        mylist.add(2,"Q");
}
for (String letter : mylist)
{
        System.out.print(letter + "   ");
}
```

(A) **V Q Q Q X Z**
(B) **V X Q Q Q Z**
(C) **Q Q Q V W X**
(D) **V W X Q Q Q**
(E) **ArrayIndexOutOfBoundsException**

For problems 7 and 8, consider the following partial class definitions:

```
public abstract class MVStudent
{
        public MVStudent()  { . . . }
}

public class APStudent extends MVStudent
{
        private int yearinschool;
        public APStudent(int year)
        {
                super();
                yearinschool = year;
        }
}

public class APCompSciStudent extends APStudent
{
        private String accountname;
        public APCompSciStudent(String acct, int yr)
        {
                /*  missing statement(s) */
        }
}
```

7. Which of the following is an acceptable replacement for **/* missing statement(s) */** in **APCompSciStudent**'s constructor?

I.
```
yearinschool = yr;
accountname = acct;
```

II.
```
super(yr);
accountname = acct;
```

III.
```
super(acct, yr);
```

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

8. Which of the following declarations are valid for the partial class definitions above?

I.
```
MVStudent student = new APStudent(11);
```

II.
```
APCompSciStudent student = new APStudent(11);
```

III.
```
APStudent student = new APCompSciStudent("ss5005547",11);
```

(A) I and II only
(B) II and III only
(C) I and III only
(D) I, II and III
(E) None of the three

9. Which of the following is **not** a method of **java.util.ArrayList<String>**?

(A) **add (String x);**
(B) **add (int index, String x);**
(C) **remove (String x);**
(D) **insert (int index, String x);**
(E) **set (int index, String x);**

10. Assume that **myList** is an **ArrayList** that has been correctly constructed and populated with objects. Which of the following expressions produces a valid random index for **myList**?

(A) **(int)(Math.random() * myList.size()) - 1**
(B) **(int)(Math.random() * myList.size())**
(C) **(int)(Math.random() * myList.size()) + 1**
(D) **(int)(Math.random() * (myList.size() + 1))**
(E) **Math.random(myList.size())**

11. The class **CourseList** provides methods that allow you to represent and manipulate a list of high school courses, but you are not concerned with how these operations work or how the list is stored in memory. You only know how to initialize and use **CourseList** objects and have no direct access to the implementation of the **CourseList** class or its private data fields. This is an example of

(A) encapsulation
(B) overriding
(C) inheritance
(D) polymorphism
(E) method overloading

12. A "worst case" situation for an insertion sort on 100 elements (taking the most steps to order the list) would be

(A) A list sorted in reverse order
(B) A list in random order
(C) A list in correct sorted order
(D) A list with the first half in sorted order, and the second half in random order
(E) A list with the first 98 elements in random order, and the last 2 elements the smallest in the list

13. Which of the following statements will compile with no errors?

I. **ArrayList<Double> vals = new List<Double>();**
II. **ArrayList<Double> vals = new ArrayList<Double>();**
III. **List<Double> vals = new ArrayList<Double>();**

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

14. Determine what will be printed from the following code fragment:

```
String [] fishy = {"One", "fish", "two", "fish", "red", "fish", "blue", "fish"};
ArrayList <String> rhyme = new ArrayList <String> ();

for (int i = 0; i < fishy.length; i++)
{
        rhyme.add(fishy[i]);                                    // Printed:
}

int index = rhyme.indexOf("fish");
for (int k = 0; k < 4; k++)
{
        rhyme.remove(index + k);
}

System.out.println("\n\n" + rhyme + "\n\n");
```

15. Determine what will be printed from the following code fragment:

```
ArrayList <Integer> mylist = new ArrayList <Integer> ();
mylist.add(7);                                                  // Printed:
mylist.add(8);
mylist.add(9);

int len = mylist.size();
for (int i = 0; i < len; i++)
{
        mylist.add(i + 1, new Integer(i));
        mylist.set(i, new Integer(i + 2));
}

System.out.println("\n\n" + mylist + "\n\n");
```