AP Computer Science A: Practice Exam 1

Part I (Multiple Choice)

Time: 90 minutes
Number of questions: 40
Percent of total score: 50

Directions: Choose the best answer for each problem. Some problems take longer than others. Consider how much time you have left before spending too much time on any one problem.

Notes:

- You may assume all import statements have been included where they are needed.
- You may assume that the parameters in method calls are not null.
- You may assume that declarations of variables and methods appear within the context of an enclosing class.

1. Consider the following code segment.

```
int myValue = 17;
int multiplier = 3;
int answer = myValue % multiplier + myValue / multiplier;
answer = answer * multiplier;
System.out.println(answer);
```

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

(A) `9`
(B) `10`
(C) `7`
(D) `30`
(E) `21`

2. Assume that `a`, `b`, and `c` have been declared and correctly initialized with `int` values. Consider the following expression.

```
boolean bool = !(a < b || b <= c) && !(a < c || b >= a);
```

Under what conditions does `bool` evaluate to true?

(A) `a = 1, b = 2, c = 3`
(B) `a = 3, b = 2, c = 1`
(C) `a = 3, b = 1, c = 2`
(D) All conditions; `bool` is always `true`.
(E) No conditions; `bool` is always `false`.

3. Consider the following code segment.

```
int[] myArray = {2, 3, 4, 1, 7, 6, 8};
int index = 0;
while (myArray[index] < 7)
{
   myArray[index] += 3;
   index++;
}
```

What values are stored in `myArray` after executing the code segment?

(A) `{2, 3, 4, 1, 7, 6, 8}`
(B) `{2, 3, 4, 1, 7, 9, 11}`
(C) `{5, 6, 7, 4, 7, 9, 8}`
(D) `{5, 6, 7, 4, 7, 6, 8}`
(E) `{5, 6, 7, 4, 0, 9, 11}`

GO ON TO THE NEXT PAGE

4. Consider the following class used by a company to represent the items it has available for online purchase.

```
public class OnlinePurchaseItem
{
        public double getPrice()        // implementation not shown
        public String getItem()          // implementation not shown
        public String getMonth()        // implementation not shown
    // instance variables, constructors, and other methods not shown
}
```

The company bills at the end of the quarter. Until then, it uses an `ArrayList` of `OnlinePurchaseItem` objects to track a customer's purchases.

```
private ArrayList<OnlinePurchaseItem> items;
```

The company decides to offer a 20-percent-off promotion on all items purchased in September. Which of the following code segments properly calculates the correct total price at the end of the quarter?

I.
```
double total = 0.0;
for (int i = 0; i < items.size(); i++)
{
   if (items.get(i).getMonth().equals("September"))
      total += 0.80 * items.get(i).getPrice();
   else
      total += items.get(i).getPrice();
}
```

II.
```
double total = 0.0;
for (OnlinePurchaseItem purchase : items)
{
   if (purchase.get(i).getMonth().equals("September"))
      total = total + 0.80 * purchase.getPrice();
   else
      total += items.get(i).getPrice();

}
```

III.
```
double total = 0.0;
for (int i = items.size(); i >= 0; i--)
{
   if (items.get(i).getMonth().equals("September"))
      total = total + 0.80 * items.get(i).getPrice();
   else
      total += items.get(i).getPrice();
}
```

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

5. Consider the following method.

```
public int mystery(int n)
{
   if (n <= 1)
      return 1;
   return 2 + mystery(n - 1);
}
```

What value is returned by the call `mystery(5)`?

(A) `1`
(B) `7`
(C) `8`
(D) `9`
(E) `10`

6. Consider the following code segment.

```
String myString = "H";
int index = 0;
while (index < 4)
{
   for (int i = 0; i < index; i++)
      myString = "A" + myString + "A";
   index ++;
}
```

What is the value of `myString` after executing the code segment?

(A) `"H"`
(B) `"AHA"`
(C) `"AAAHAAA"`
(D) `"AAAAHAAAA"`
(E) `"AAAAAAHAAAAAA"`

7. Consider the following statement.

```
int var = (int)(Math.random() * 50) + 10;
```

What are the possible values of `var` after executing the statement?

(A) All integers from 1 to 59 (inclusive)
(B) All integers from 10 to 59 (inclusive)
(C) All integers from 10 to 60 (inclusive)
(D) All real numbers from 50 to 60 (not including 60)
(E) All real numbers from 10 to 60 (not including 60)

8. Consider the following statement.

```
System.out.print(13 + 6 + "APCSA" + (9 - 5) + 4);
```

What is printed as a result of executing the statement?

(A) `136APCSA(9 - 5)4`
(B) `19APCSA8`
(C) `19APCSA44`
(D) `136APCSA8`
(E) `136APCSA44`

GO ON TO THE NEXT PAGE

9. Consider the following method.

```
public int guess(int num1, int num2)
{
   if (num1 % num2 == 0)
      return (num2 + num1) / 2;
   return guess(num2, num1 % num2) + (num1 % num2);
}
```

What is the value of num after executing the following code statement?

```
int num = guess(3, 17);
```

(A) 6
(B) 7
(C) 10
(D) Nothing is returned. Modulus by 0 causes an `ArithmeticException`.
(E) Nothing is returned. Infinite recursion causes a stack overflow error.

10. Consider the following class.

```
public class Automobile
{
   private String make, model;
   public Automobile(String myMake, String myModel)
   {
      make = myMake;
      model = myModel;
   }

   public String getMake()
   {  return make;  }

   public String getModel()
   {  return model;  }

   /* Additional implementation not shown */
}
```

Consider the following code segment that appears in another class.

```
Automobile myCar = new Automobile("Ford", "Fusion");
Automobile companyCar = new Automobile("Toyota", "Corolla");
companyCar = myCar;
myCar = new Automobile("Kia", "Spectre");
System.out.println(companyCar.getMake() + " " + myCar.getModel());
```

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

(A) Ford Spectre
(B) Ford Fusion
(C) Kia Spectre
(D) Toyota Corolla
(E) Toyota Spectre

Questions 11–14 refer to the following class definition.

```
public class Depot
{
   private String city;
   private String state;
   private String country = "USA";
   private boolean active = true;

   public Depot(String myCity, String myState, String myCountry, boolean myActive)
   {
      city = myCity;
      state = myState;
      country = myCountry;
      active = myActive;
   }

   public Depot(String myState, String myCity)
   {
      state = myState;
      city = myCity;
   }

   public Depot(String myCity, String myState, boolean myActive)
   {
      city = myCity;
      state = myState;
      active = myActive;
   }

   public String toString()
   {
      return city + ", " + state + " " + country + " " + active;
   }

   /* Additional implementation not shown */
}
```

11. The `Depot` class has three constructors. Which of the following is the correct term for this practice?

(A) Overriding
(B) Procedural abstraction
(C) Encapsulation
(D) Polymorphism
(E) Overloading

12. Consider the following code segment in another class.

```
Depot station = new Depot("Oakland", "California");
System.out.println(station);
```

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

(A) `California, Oakland USA true`
(B) `California, Oakland USA active`
(C) `USA, California USA true`
(D) `Oakland, California USA active`
(E) `Oakland, California USA true`

GO ON TO THE NEXT PAGE

13. Consider the following class definition.

```
public class WhistleStop extends Depot
```

Which of the following constructors compiles without error?

I.
```
public WhistleStop()
{
   super();
}
```

II.
```
public WhistleStop()
{
   super("Waubaushene", "Ontario", "Canada", true);
}
```

III.
```
public WhistleStop(String city, String province)
{
   super(city, province);
}
```

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

14. Assume a correct no-argument constructor has been added to the `WhistleStop` class.

Which of the following code segments compiles without error?

I.
```
ArrayList<Depot> stations = new ArrayList<Depot>();
stations.add(new WhistleStop());
stations.add(new Depot("Mansonville", "Quebec", "Canada", false));
```

II.
```
ArrayList<WhistleStop> stations = new ArrayList<Depot>();
stations.add(new WhistleStop());
stations.add(new Depot("Orinda", "California", true));
```

III.
```
ArrayList<WhistleStop> stations = new ArrayList<WhistleStop>();
stations.add(new WhistleStop());
stations.add(new Depot("Needham", "Massachusetts", true));
```

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

15. Consider the following code segment.

```
String crazyString = "crazy";
ArrayList<String> crazyList = new ArrayList<String>();
crazyList.add("weird");
crazyList.add("enigma");
for (String s : crazyList)
{
   crazyString = s.substring(1, 3) + crazyString.substring(0, 4);
}
System.out.println(crazyString);
```

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

(A) `nicraz`
(B) `nieicr`
(C) `nieicraz`
(D) `einicraz`
(E) `weienicraz`

16. Consider the following method.

```
public int pathways(int n)
{
   int ans;
   if (n < -5)
      ans = 1;
   else if (n < 0)
      ans = 2;
   else if (n > 10)
      ans = 3;
   else
      ans = 4;
   return ans;
}
```

Which of the following sets of data tests every possible path through the code?

(A) `-6, -1, 15, 12`
(B) `-5, -3, 12, 15`
(C) `-8, -5, 8, 10`
(D) `-6, 0, 20, 7`
(E) `-10, -5, 10, 12`

GO ON TO THE NEXT PAGE

Questions 17–18 refer to the following scenario.

A resort wants to recommend activities for its guests, based on the temperature (degrees F) as follows:

76° and above	go swimming
61°–75°	go hiking
46°–60°	go horseback riding
45° and below	play ping pong

17. Consider the following method.

```
public String chooseActivity(int temperature)
{
   String activity;
   if (temperature > 75)
      activity = "go swimming";
   if (temperature > 60)
      activity = "go hiking";
   if (temperature > 45)
      activity = "go horseback riding";
   else
      activity = "play ping pong";
   return activity;
}
```

Consider the following statement.

```
System.out.println("We recommend that you " + chooseActivity(temperature));
```

For which temperature range is the correct suggestion printed (as defined above)?

(A) `temperature > 75`
(B) `temperature > 60`
(C) `temperature > 45`
(D) `temperature <= 60`
(E) Never correct

18. After discovering that the method did not work correctly, it was rewritten as follows.

```
public String chooseActivity(int temperature)
{
   String activity;
   if (temperature > 45)
      activity = "go horseback riding";
   else if (temperature > 60)
      activity = "go hiking";
   else if (temperature > 75)
      activity = "go swimming";
   else
      activity = "play ping pong";
   return activity;
}
```

Consider the following statement.

```
System.out.println("We recommend that you " + chooseActivity(temperature));
```

What is the largest temperature range for which the correct suggestion is printed (as defined above)?

(A) Always correct
(B) Never correct
(C) `temperature <= 75`
(D) `temperature <= 60`
(E) `temperature <= 45`

19. Assume `int[] arr` has been correctly instantiated and is of sufficient size. Which of these code segments results in identical arrays?

I.
```
int i = 1;
while (i < 6)
{
   arr[i / 2] = i;
   i += 2;
}
```

II.
```
for (int i = 0; i < 3; i++)
{
   arr[i] = 2 * i + 1;
}
```

III.
```
for (int i = 6; i > 0; i = i - 2)
{
   arr[(6 - i) / 2] = 6 - i;
}
```

(A) I and II only
(B) II and III only
(C) I and III only
(D) I, II, and III
(E) All three outputs are different.

20. Consider the following code segment.

```
ArrayList<String> nations = new ArrayList<String>();
nations.add("Argentina");
nations.add("Canada");
nations.add("Australia");
nations.add("Cambodia");
nations.add("Russia");
nations.add("France");
for (int i = 0; i < nations.size(); i++)
{
   if (nations.get(i).length() >= 7)
      nations.remove(0);
}
System.out.println(nations);
```

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

(A) `[Canada, Russia, France]`
(B) `[Cambodia, Russia, France]`
(C) `[Australia, Cambodia, Russia, France]`
(D) `[Canada, Cambodia, Russia, France]`
(E) Nothing is printed. `IndexOutOfBoundsException`

21. Consider the following method.

```
public int doStuff(int[] numberArray)
{
   int index = 0;
   int maxCounter = 1;
   int counter = 1;
   for (int k = 1; k < numberArray.length; k++)
   {
      if (numberArray[k] == numberArray[k - 1])
      {
         if (counter <= maxCounter)
         {
            maxCounter = counter;
            index = k;
         }
         counter++;
      }
      else
      {
         counter = 1;
      }
   }
   return numberArray[index];
}
```

Consider the following code segment.

```
int[] intArray = {3, 3, 4, 4, 6};
System.out.println(doStuff(intArray));
```

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

(A) 1
(B) 3
(C) 4
(D) 5
(E) 6

Questions 22–23 refer to the following class.

```
public class Coordinates
{
   private int x, y;

   public Coordinates(int myX, int myY)
   {
      x = myX;
      y = myY;
   }

   public void setX(int myX)
   {  x = myX;  }

   public int getX()
   {  return x;  }

   public void setY(int myY)
   {  y = myY;  }

   public int getY()
   {  return y;  }
}
```

Consider the following code segment.

```
Coordinates c1 = new Coordinates(0, 10);
Coordinates c3 = c1;
Coordinates c2 = new Coordinates(20, 30);
c3.setX(c2.getY());
c3 = c2;
c3.setY(c2.getX());
c2.setX(c1.getX());
```

22. Which of the following calls returns the value 20?

(A) `c1.getX()`
(B) `c1.getY()`
(C) `c2.getX()`
(D) `c2.getY()`
(E) `c3.getX()`

23. Which of the following calls returns the value 30?

(A) `c1.getX()`
(B) `c2.getX()`
(C) `c3.getX()`
(D) All of the above
(E) None of the above

24. Consider the following method.

```
public boolean whatDoesItDo(String st)
{
   for (int i = 0; i < st.length() / 2; i++)
   {
      String sub1 = st.substring(i, i + 1);
      String sub2 = st.substring(st.length() - i - 1, st.length() - i);
      if (!sub1.equals(sub2))
      {
         return false;
      }
   }
   return true;
}
```

What is the purpose of the method `whatDoesItDo`?

(A) The method returns `true` if `st` has an even length, `false` if it has an odd length.
(B) The method returns `true` if `st` contains any character more than once.
(C) The method returns `true` if `st` is a palindrome (spelled the same backward and forward), `false` if it is not a palindrome.
(D) The method returns `true` if `st` begins and ends with the same letter.
(E) The method returns `true` if the second half of `st` contains the same sequence of letters as the first half, `false` if it does not.

25. Consider the following method.

```
public String mixItUp(String entry1, String entry2)
{
   entry2 = entry1;
   entry1 = entry1 + entry2;
   String entry3 = entry1 + entry2;
   return entry3;
}
```

What is returned by the call `mixItUp("AP", "CS")`?

(A) `"APAP"`
(B) `"APCSAP"`
(C) `"APAPAP"`
(D) `"APCSCS"`
(E) `"APAPAPAP"`

Questions 26–28 refer to the following classes.

```
public class Person
{
   private String firstName;
   private String lastName;

   public Person(String fName, String lName)
   {
      firstName = fName;
      lastName = lName;
   }

   public String getName()
   {
      return firstName + " " + lastName;
   }
}

public class Adult extends Person
{
   private String title;

   public Adult(String fname, String lName, String myTitle)
   {
      super(fname, lName);
      title = myTitle;
   }

   /*  toString method to be implemented in question 27  */
}

public class Child extends Person
{
   private int age;

   /*  Constructor to be implemented in question 26  */
}
```

26. Which of the following is a correct implementation of a `Child` class constructor?

(A)
```
public Child(String first, String last, String t, int a)
{
    super(first, last, t);
    age = a;
}
```

(B)
```
public Child()
{
    super();
}
```

(C)
```
public Child(String first, String last, int a)
{
    super(first, last);
    age = a;
}
```

(D)
```
public Child extends Adult (String first, String last, String t, int a)
{
    super(first, last, t);
    age = a;
}
```

(E)
```
public Child extends Person(String first, String last, int a)
{
    super(first, last);
    age = a;
}
```

27. Which of the following is a correct implementation of the `Adult` class `toString` method?

(A)
```
public String toString()
{
    System.out.println(title + " " + super.toString());
}
```

(B)
```
public String toString()
{
    return title + " " + super.toString();
}
```

(C)
```
public String toString()
{
    return title + " " + firstName + " " + lastName;
}
```

(D)
```
public void toString()
{
    System.out.println(title + " " + super.getName());
}
```

(E)
```
public String toString()
{
    return title + " " + super.getName();
}
```

28. Consider the following method declaration in a different class.

```
public void findSomeone(Adult someone)
```

Assume the following variables have been correctly instantiated and initialized with appropriate values.

```
Person p;
Adult a;
Child c;
```

Which of the following method calls compiles without error?

I. `findSomeone(p);`
II. `findSomeone(a);`
III. `findSomeone(c);`
IV. `findSomeone((Adult) p);`
V. `findSomeone((Adult) c);`

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

29. Consider the following method. The method is intended to return `true` if the value `val` raised to the power `power` is within the tolerance `tolerance` of the target value `target`, and `false` otherwise.

```
public boolean similar(double val, double power, double target, double tolerance)
{
   /* missing code */
}
```

Which code segment below can replace `/* missing code */` to make the method work as intended?

(A)
```
double answer = Math.pow(val, power);
if (answer - tolerance >= target)
   return true;
return false;
```

(B)
```
return (Math.abs(Math.pow(val, power))) - target <= tolerance;
```

(C)
```
double answer = Math.pow(Math.abs(val), power);
return answer - target <= tolerance;
```

(D)
```
double answer = Math.pow(val, power) - target;
boolean within = answer - tolerance >= 0;
return within;
```

(E)
```
return (Math.abs(Math.pow(val, power) - target) <= tolerance);
```

30. Consider the following method.

```
public int changeEm(int num1, int num2, int[] values)
{
   num1 = values[num2];
   values[num1] = num2;
   num2++;
   return num2;
}
```

Consider the following code segment.

```
int[] values = {1, 2, 3, 4, 5};
int num1 = 2;
int num2 = 3;
num2 = changeEm(num1, num2, values);
System.out.println("num1 = " + num1 + "  values[" + num2 + "] = " + values[num2]);
```

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

(A) `num1 = 2  values[4] = 4`
(B) `num1 = 2  values[4] = 5`
(C) `num1 = 2  values[4] = 3`
(D) `num1 = 4  values[3] = 3`
(E) `num1 = 4  values[3] = 4`

31. Consider the following method.

```
public int firstLetterIndex(String s2)
{
   String s1 = "abcdefghijklmnopqrstuvwxyz";
   return s1.indexOf(s2.substring(s2.length() - 2));
}
```

What is returned as a result of the call `firstLetterIndex("on your side")`?

(A) `-1`
(B) `0`
(C) `3`
(D) `4`
(E) `5`

32. The following code segment appears in the main method of another class.

```
AnotherClass[] acArray = new AnotherClass[10];
acArray[2] = new AnotherClass("two");
acArray[3] = new AnotherClass("three");
for (int i = 0; i < acArray.length; i++)
{
   /* missing condition */
   {
      System.out.println(acArray[i].getData());
   }
}
```

Which of the following statements should be used to replace `/* missing condition */` so that the code segment will not terminate with a `NullPointerException`?

(A) `if (acArray[i] != null)`
(B) `if (acArray.get(i) != null)`
(C) `if (acArray.getData() != null)`
(D) `if (acArray.getData().equals("two") || acArray.getData().equals("three"))`
(E) Since not all of acArray's entries contain an `AnotherClass` object, there is no way to loop through the array without a `NullPointerException`.

33. A programmer intends to apply the standard Binary Search algorithm on the following array of integers. The standard Binary Search algorithm returns the index of the search target if it is found and -1 if the target is not found. What is returned by the algorithm when a search for 50 is executed?

```
int[] array = {9, 100, 11, 45, 76, 100, 50, 1, 0, 55, 99};
```

(A) `-1`
(B) `0`
(C) `5`
(D) `6`
(E) `7`

34. Consider the following code segment.

```
int[][] nums = { {0, 1, 2}, {3, 4, 5}, {6, 7, 8} };

for (int x = 0; x < nums.length; x++)
{
   int temp = nums[x][0];
   nums[x][0] = nums[x][2];
   nums[x][2] = temp;
}
```

What are the values in array nums after the code segment is executed?

(A) `{ {0, 1, 0}, {3, 4, 3}, {6, 7, 6} }`
(B) `{ {0, 0, 0}, {3, 3, 3}, {6, 6, 6} }`
(C) `{ {2, 1, 0}, {5, 4, 3}, {8, 7, 6} }`
(D) `{ {0, 1, 2}, {3, 4, 5}, {6, 7, 8} }`
(E) `{ {8, 7, 6}, {5, 4, 3}, {2, 1, 0} }`

GO ON TO THE NEXT PAGE

35. Consider the following code segment.

```
ArrayList<Integer> newList = new ArrayList<Integer>();
Integer[] list = {2, 4, 3, 3, 2, 5, 1};
newList.add(list[0]);

for (int i = 1; i < list.length; i++)
{
   if (list[i] > newList.get(newList.size() - 1))
      newList.add(list[i]);
}

System.out.println(newList);
```

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

(A) `[2, 4, 5]`
(B) `[2, 2, 1]`
(C) `[2, 4, 3, 5, 1]`
(D) `[1, 2, 3, 3, 4, 5]`
(E) `[2, 4, 3, 3, 5, 1]`

36. Consider the following code segment.

```
int[][] a = new int[5][5];

for (int r = 0; r < a.length; r++)
   for (int c = r + 1; c < a[0].length; c++)
      a[r][c] = 9;

for (int d = 0; d < a.length; d++)
   a[d][d] = d;

System.out.println(a[1][2] + " " + a[3][3] + " " + a[4][3]);
```

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

(A) `9 9 9`
(B) `0 0 0`
(C) `0 3 9`
(D) `9 3 9`
(E) `9 3 0`

37. Consider the following sorting method.

```
public void mysterySort(int[] arr)
{
   for (int j = 1; j < arr.length; j++)
   {
      int temp = arr[j];
      int index = j;
      while (index > 0 && temp < arr[index - 1])
      {
         arr[index] = arr[index - 1];
         index--;
      }
      arr[index] = temp;
   }
}
```

Consider the following code segment.

```
int[] values = {9, 1, 3, 0, 2};
mysterySort(values);
```

Which of the following shows the elements of the array in the correct order after the second pass through the outer loop of the sorting algorithm?

(A) `{0, 1, 2, 9, 3}`
(B) `{0, 1, 3, 9, 2}`
(C) `{0, 1, 9, 3, 2}`
(D) `{1, 3, 9, 0, 2}`
(E) `{1, 3, 0, 2, 9}`

38. Consider the following method.

```
public int puzzle(int m, int n)
{
   if (n == 1)
      return m;
   return m * puzzle(m, n - 1);
}
```

What is returned by the method call `puzzle(3, 4)`?

(A) `9`
(B) `64`
(C) `81`
(D) `128`
(E) Nothing is returned. Infinite recursion causes a stack overflow error.

39. Consider the following code segment.

```
int[][] table = new int[5][6];
int val = 0;
for (int k = 0; k < table[0].length; k++)
{
   for (int j = table.length - 1; j >= 0; j--)
   {
      table[j][k] = val;
      val = val + 2;
   }
}
```

What is the value of `table[3][4]` after executing the code segment?

(A) `22`
(B) `30`
(C) `34`
(D) `42`
(E) `46`

40. Consider the following method.

```
public void selectionSort(String[] arr)
{
   for (int i = 0; i < arr.length; i++)
   {
      int small = i;
      for (int j = i; j < arr.length; j++)
      {
         /* missing code */
         small = j;
      }
      String temp = arr[small];
      arr[small] = arr[i];
      arr[i] = temp;
   }
}
```

Assume `String[] ray` has been properly instantiated and populated with `String` objects.

Which line of code should replace `/* missing code */` so that the method call `selectionSort(ray)` results in an array whose elements are sorted from least to greatest?

(A) `if (arr[j].equals(arr[small]))`
(B) `if (arr[j] > arr[small])`
(C) `if (arr[j] < arr[small])`
(D) `if (arr[j].compareTo(arr[small]) > 0)`
(E) `if (arr[j].compareTo(arr[small]) < 0)`

STOP. End of Part I.

AP Computer Science A: Practice Exam 1

Part II (Free Response)

Time: 90 minutes
Number of questions: 4
Percent of total score: 50

Directions: Write all of your code in Java. Show all your work.

Notes:

- You may assume all imports have been made for you.
- You may assume that all preconditions are met when making calls to methods.
- You may assume that all parameters within method calls are not null.
- Be aware that you should, when possible, use methods that are defined in the classes provided as opposed to duplicating them by writing your own code.

1. A `Password` class contains methods used to determine information about passwords. You will write two methods of the class.

```
public class Password
{
    private String upper = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
    private String lower = "abcdefghijklmnopqrstuvwxyz";
    private String symbols = "!@#$%^&*";
    private int minLength;
    private int maxLength;

    public Password(int min, int max)
    {
        /* implementation not shown    */
    }

    public boolean isValid(String password)
    {
        /* implemented in part (a)     */
    }

    public String generatePassword()
    {
        /* implemented in part (b)      */
    }

    // There may be instance variables, constructors, and other methods not shown.
}
```

(a) Write the method `isValid`, which returns true if a password is valid and false otherwise. A password is considered valid if

- Its length is a valid length. The length must be at least minimum characters and at most maximum characters, inclusive.
- It contains at least one uppercase letter, one lowercase letter, and one symbol.

Statement	Value Returned	Comment
Password p1 = new Password(3,15);		Passwords can be any length between 3 and 15 inclusive.
p1.isValid("HelloWorld!");	true	Password has length 11, contains at least 1 uppercase letter, 1 lowercase letter, and 1 symbol.
p1.isValid("Aloha")	false	Password does not include a symbol
p1.isValid("#APComputerScience Rocks");	false	Password has length 23

Complete method `isValid` below.

```
public boolean isValid(String password)
```

(b) Write the method `generatePassword`, which returns a `String` representing a valid password. In writing `generatePassword` you must call `isValid`. Assume `isValid` works correctly regardless of what you wrote in part (a).

The `generatePassword` method must randomly choose the length of the password and randomly choose characters from the uppercase, lowercase, and symbol strings.

Statement	Value Returned	Comment
Password p2 = new Password(4,10);		Passwords can be any length between 4 and 10 inclusive.
p2.generatePassword();	Bkb@mHI	Password has length 7, contains at least 1 uppercase letter, 1 lowercase letter, and 1 symbol.
p2.generatePassword();	k&iHBe	Password has length 6, contains at least 1 uppercase letter, 1 lowercase letter, and 1 symbol.
p2.generatePassword();	hY*G	Password has length 4, contains at least 1 uppercase letter, 1 lowercase letter, and 1 symbol.

Complete method `generatePassword` below.

```
public String generatePassword()
```

2. An ISBN (International Standard Book Number) is a numeric book identifier that is assigned to each published book. Prior to 2007, the ISBN numbers were 10 digits in length. The ISBN number is broken down into sections (group, publisher, title, and check digit). The last digit, or check digit, is used for error detection.

To generate a valid check digit, the first nine digits in the ISBN is multiplied by a weight. Each weight is multiplied by the corresponding digit, and then the products are added together.

ISBN with no check digit	1	2	6	0	4	5	4	9	1
Weight	10	9	8	7	6	5	4	3	2
Product	10	18	48	0	24	25	16	27	2
Sum	10 + 18 + 48 + 0 + 24 + 25 + 16 + 27 + 2 = 170								
Check Digit	11 − (sum % 11) = 11 − (170%11) = 6								

ISBN with no check digit	0	3	0	6	4	0	6	1	1
Weight	10	9	8	7	6	5	4	3	2
Product	0	27	0	42	24	0	24	3	2
Sum	0 + 27 + 0 + 42 + 24 + 0 + 24 + 3 + 2 = 122								
Check Digit	11 − (sum % 11) = 11 − (122%11) = 10 = X								

If the value of the check digit is 10, then the check digit becomes "X".

An `ISBN` object is created with a parameter that contains a nine-digit ISBN number. The `ISBN` class provides two methods: `calculateCheckDigit` and `generateNumber`.

- `calculateCheckDigit` will perform the calculations described above to determine the check digit.
- `generateNumber` will join together the nine-digit number and the check digit generated from the method defined above.

For example, consider the following code that is in a class other than the `ISBN` class.

```
ISBN book1 = new ISBN(126045491);
String check1 = book1.calculateCheckDigit();          //  "6" will be returned
String isbnNumber1 = book1.generateNumber();          // "126045491-6" will be returned

ISBN book2 = new ISBN(030640611);
String check2 = book2.calculateCheckDigit();          //  "X" will be returned
String isbnNumber2 = book2.generateNumber();          // "030640611-X" will be returned
```

Write the complete `ISBN` class, including the constructor and any required instance variables and methods. Your implementation must meet all specifications to confirm to the example.

3. A train is composed of an engine and any number of train cars. The engine is rated to pull up to a maximum weight, based on its power. The weight of all the train cars combined, including the engine, must be below this maximum weight or the engine will not be able to move the train.

Trains are represented by the `Train` class and contain an engine followed by any number of cars. An `ArrayList` will be used to store the weight of the engine (in element 0), and the weights of each of the cars (in elements 1, 2, 3, etc.).

The weight of a `Train` object is calculated by adding the weight of the engine to the weight of each of the objects in the `trainCars ArrayList` (to be completed in part (a).

Trains need to be checked to make sure that their weight can be pulled by their engine. If the train is overweight, train operators must remove train cars from the end of the train until the train is within the acceptable weight range (to be completed in part (b)).

```
public class Train
{
   private double maxWeight;
   private ArrayList <Double> trainCars;

   public Train(Double max, ArrayList <Double> tc)
   {
      maxWeight = max;
      trainCars = tc;
   }

   public double getMaximumWeight()
   /*       implementation not shown    */

   /** Finds the total weight of the train.
    *
    *    @return double value containing the sum of the weights of the train
    */
   public double getTotalWeight()
   {     /*     To be implemented in part (a)     */     }


   /** Removes Double objects from the end of the train
    *    until the train can be pulled by the Engine.
    *
    *    @return ArrayList<Double> containing the removed cars in
    *            the order they were removed (the last car is
    *            item 0, etc.). If no cars are removed, the returned
    *            list will be empty.
    */
   public ArrayList<Double> removeExcessTrainCars()
   {     /*     To be implemented in part (b)     */     }

     /* Additional implementation not shown */
   }
```

(a) Write the `getTotalWeight` method that will calculate the total weight of the train by adding the weight of each of the train cars to the weight of the engine (element 0).

```
/** Finds the total weight of the train.
 *
 * @return double value containing the sum of the weights of the train cars
 */
public double getTotalWeight()
{  /*     To be implemented in part (a)     */     }
```

GO ON TO THE NEXT PAGE

(b) Write the `removeExcessTrainCars` method of the `Train` class that removes `Double` objects one at a time from the end of the train until the train is less than or equal to the maximum weight allowed as given by the `getMaximumWeight` method of the `Train` object. The removed train cars are added to the end of an `ArrayList` of `Double` objects as they are removed. This `ArrayList` of removed `Double` objects is returned by the method. If no `Double` objects need to be removed, an empty `ArrayList` is returned. You may assume what you wrote for part (a) works as intended.

Example:
A train is composed of the cars listed below. The engine has a maximum weight rating of 475,000 pounds.

Car	Weight
Engine/ `trainCars[0]`	200,000
`trainCars[1]`	100,000
`trainCars[2]`	150,000
`trainCars[3]`	50,000
`trainCars[4]`	100,000
`trainCars[5]`	60,000

In the example, the train initially weighs 650,000 pounds. Train cars need to be removed one by one from the end of the train until the total weight is under the maximum allowed weight of 475,000.

- Removing the train car from index 4 lowers the weight to 590,000
- Removing the train car from index 3 lowers the weight to 490,000
- Removing the train car from index 2 lowers the weight to 440,000

At 440,000 pounds, the train is now in the acceptable weight range. The following `ArrayList` is returned by the method:

```
[60000.0, 100000.0, 50000.0]
```

Complete the `removeExcessTrainCars` method.

```
/** Removes TrainCar objects from the end of the train
 *  until the train can be pulled by the Engine.
 *
 *  @return ArrayList<TrainCar> containing the removed cars in
 *          the order they were removed (the first car removed is
 *          item 0, etc.) If no cars are removed, the returned
 *          list will be empty.
 */
public ArrayList <Double> removeExcessTrainCars()
```

4. The colors of the pixels on a TV or computer screen are made by combining different quantities of red, green, and blue light. Each of the three colors is represented by a value between 0 and 255, where 0 means none of that color light and 255 means as much of that color as there can be. This way of representing colors is referred to as RGB, for red-green-blue, and the color values are written in ordered triples like this (80, 144, 240). That particular combination means red at a value of 80, green at a value of 144, and blue at a value of 240. The resulting color is a kind of medium blue. The combination (0, 0, 0) produces black and (255, 255, 255) produces white.

Pixels can be modeled by the `Pixel` class shown below.

```
public class Pixel
{
   private int red;
   private int green;
   private int blue;

   public Pixel(int myRed, int myGreen, int myBlue)
   {
      red = myRed;
      green = myGreen;
      blue = myBlue;
   }
   public String toString()
   {
      return "(" + red + ", " + green + ", "  + blue + ")";
   }
}
```

An artist wants to manipulate the pixels on a large computer display. She begins by creating three two-dimensional arrays the size of the screen, `rows` by `columns`. One array contains the red value of each pixel, one contains the blue value of each pixel, and one contains the green value of each pixel. She then calls methods of the `AlterImage` class to generate and then manipulate the data.

```
public class AlterImage

{
   /** Converts 3 arrays of color values into one array
    *  of Pixel objects
    *
    *  @param reds the red color values
    *  @param greens the green color values
    *  @param blues the blue color values
    *  @return the Pixel array generated with information
    *          in the red, green and blue arrays
    *
    *  Precondition:  The three color arrays are all the same
    *                 size and contain int values in the range 0-255.
    *  Postcondition: The returned array is the same size as the
    *                 3 color arrays.
    */
   public Pixel[][] generatePixelArray(int[][] reds, int[][] greens, int[][] blues)
   {  /* to be implemented in part (a) */  }
```

GO ON TO THE NEXT PAGE

```
    /** Flips a 2D array of Pixel objects either
     *  horizontally or vertically
     *
     *  @param image the 2D array of Pixel objects to be processed
     *  @param horiz true: flip horizontally, false: flip vertically
     *  @return the processed image
     */
    public Pixel[][] flipImage(Pixel[][] image, boolean horiz)
    {  /* to be implemened in part (b) */  }

    /* Additional implementation not shown */

}
```

(a) Given the three arrays `reds`, `greens` and `blues`, and the `Pixel` and `AlterImage` classes, implement the `generatePixelArray` method that converts the raw information in the three color arrays into a `rows` by `columns` array of `Pixel` objects.

Precondition: The three color arrays are all the same size and contain `int` values in the range 0–255.
Postcondition: The returned array is the same size as the three color arrays.

```
/** Converts 3 arrays of color values into one array
 *  of Pixel objects
 *
 *  @param reds the red color values
 *  @param greens the green color values
 *  @param blues the blue color values
 *  @return the Pixel array generated with information
 *          from the red, green and blue arrays
 *
 *  Precondition:    The three color arrays are all the same
 *                   size and contain int values in the range 0-255.
 *  Postcondition:   The returned array is the same size as the
 *                   3 color arrays.
 */
public Pixel[][] generatePixelArray(int[][] reds, int[][] greens, int[][] blues)
```

(b) The artist uses various techniques to modify the image displayed on the screen. One of these techniques is to flip the image, either horizontally (along the *x*-axis, top-to-bottom) or vertically (along the *y*-axis, left-to-right), so that a mirror image is produced as shown below.

In this example, the data in the cells are the RGB values stored in the `Pixel` objects. To make the example clearer, red is set to the original row number, green is set to the original column number, and blue is constant at 100.

Original array:

0, 0, 100	0, 1, 100	0, 2, 100	0, 3, 100
1, 0, 100	1, 1, 100	1, 2, 100	1, 3, 100
2, 0, 100	2, 1, 100	2, 2, 100	2, 3, 100
3, 0, 100	3, 1, 100	3, 2, 100	3, 3, 100

Flipped horizontally:

3, 0, 100	3, 1, 100	3, 2, 100	3, 3, 100
2, 0, 100	2, 1, 100	2, 2, 100	2, 3, 100
1, 0, 100	1, 1, 100	1, 2, 100	1, 3, 100
0, 0, 100	0, 1, 100	0, 2, 100	0, 3, 100

Flipped vertically:

0, 3, 100	0, 2, 100	0, 1, 100	0, 0, 100
1, 3, 100	1, 2, 100	1, 1, 100	1, 0, 100
2, 3, 100	2, 2, 100	2, 1, 100	2, 0, 100
3, 3, 100	3, 2, 100	3, 1, 100	3, 0, 100

Write the `flipImage` method, which takes as parameters a 2-D array of `Pixel` objects and a `boolean` direction to flip, where true means horizontally and false means vertically. The method returns the flipped image as a 2-D array of `Pixel` objects.

```
/** Flips a 2D array of Pixel objects either
 *  horizontally or vertically
 *
 *  @param image the 2D array of Pixel objects to be processed
 *  @param horiz true: flip horizontally, false: flip vertically
 *  @return the processed image
 */
public Pixel[][] flipImage(Pixel[][] image, boolean horiz)
```

STOP. End of Part II.