1. Program Traces (40 points, 55 minutes)

a. (12 points) When executed, what does each class display on the screen?

```
public class C1 {
    public static void main (String [] args)
    {
        mystery1();
        mystery2();
    }
    public static void mystery1()
    {
        System.out.println("m1");
        mystery2();
    }
    public static void mystery2()
    {
        System.out.println("m2");
    }
}
```

```
public class C2 {
    public static void mystery1(int x)
    {
        if(x<10) {
            mystery2("x=\" + x);
        }
        System.out.println("m1");
    }
    public static void mystery2(String s)
    {
        if(s.length()>5) {
            mystery1(s.length());
        }
        System.out.println(s);
    }
    public static void main (String [] args)
    {
        mystery1(7);
        mystery2("goodness");
        mystery2("me");
    }
}
```

Output:

```
m1
m2
```

Output:

```
x=7
m1
x=8
m1
goodness
me
```
b. (18 points) Answer the questions below about each class.

<table>
<thead>
<tr>
<th>Class</th>
<th>Code</th>
<th>Questions</th>
</tr>
</thead>
</table>
| **public class C1** | public class C1 { 
  public static void mystery (int a, boolean b) { 
    // POINT 1 
    a += 2; 
    b = !b; 
    // POINT 2 
  } 
  public static void main (String [] args) { 
    int x = 3; 
    boolean y = false; 
    mystery(x, y); 
    // POINT 3 
  } 
} | 1. What are the values of a and b at POINT 1, when this program is executed? 
3, false |
| | | 2. What are the values of a and b at POINT 2? 
5, true |
| | | 3. What are the values of x and y at POINT 3? 
3, false |
| **public class C2** | public class C2 { 
  public static void mystery1 (double [] d) { 
    for(int i=1; i<d.length; i++) { 
      d[i] = d[i] - d[i-1]; 
    } 
  } 
  public static double [] mystery2 (double [] d) { 
    double [] ret = new double[d.length]; 
    for(int i=0; i<ret.length; i++) { 
      ret[i] = d[i]-1; 
      d[i] += 1; 
    } 
    // POINT 2 
    return ret; 
  } 
  public static void main (String [] args) { 
    double [] d1 = {1.0, -1.0, 3.0}; 
    double [] d2 = {5.5, 6.5, 7.5}; 
    mystery1(d1); 
    // POINT 1 
    double [] d3 = mystery2(d2); 
    // POINT 3 
  } 
} | 1) Draw what d1 looks like at POINT 1. 
{1.0, -2.0, 5.0} |
| | 2) Draw what d, ret, and d2 look like at POINT 2. 
{6.5, 7.5, 8.5} (d, d2) 
{4.5, 5.5, 6.5} |
| | 3) Draw what d2 and d3 look like at POINT 3. 
{6.5, 7.5, 8.5} 
{4.5, 5.5, 6.5} |
c. (10 points) For each snippet of Java code on the left, write down the value of the variables after the code is finished executing.

<table>
<thead>
<tr>
<th>Code</th>
<th>Class definition</th>
<th>Value of variable at end of Code in LEFT COLUMN</th>
</tr>
</thead>
</table>
| public class Mystery1 { public double a; public void update() { a += 2.5; } }  
Mystery1 m = new Mystery1(); m.a = 1.0; m.update(); | m = 3.5 |
| public class Mystery2 { private int a = 0, b = 1; public int getVal() { a = a + b; return a; } }  
Mystery2 m1 = new Mystery2(); int x = m1.getVal();  
Mystery2 m2 = new Mystery2(); x = m2.getVal(); x = m2.getVal(); | x = 2  
m1 = (1, 1)  
m2 = (2, 1) |
| public class Mystery3 { private String s = "ba"; public String getStr(int num) { n = num; for(int i=0;i<n;i++) { s = s + n; } return s; } }  
Mystery3 m1 = new Mystery3(); String str1 = m1.getStr(1);  
Mystery3 m2 = new Mystery3(); String str2 = m2.getStr(2); | m1 = ("ba1", 1)  
str1 = "ba1"  
m2 = ("ba22", 2)  
str2 = "ba22" |
| public class Mystery4 { public int [] a = null; public Mystery4 (int len, int val){ a = new int[len]; for(int i=0;i<len; i++) { a[i] = val; } } }  
Mystery4 m1 = new Mystery4(2,3);  
Mystery4 m2 = new Mystery4(3,1); m2.a[1] = 0; | m1 = (3, 3)  
m2 = (1, 0, 1) |

2) Writing short programs (42 points, 40 minutes)

a. (8 points) Write an alternative version of the class below. Your version should contain exactly two static methods: a main method and a drawLine method. The output of your class should be identical to the output of the class below, but the main method should contain exactly three calls to the drawLine method.

```java
public class MyVersion {
    public static void main(String [] args) {
        System.out.println("*****");
        System.out.println("***");
    }
}
```
System.out.println("**********");
}
}

public class AlternativeVersion {

public static void main(String[] args) {
    drawline(5);
    drawline(3);
    drawline(10);
}

public static void drawline(int x) {
    for (int i = 0; i < x; i++)
        System.out.print("*");
    System.out.println();
}

}

b. (10 points) Write a static method named DigitSum that receives an integer value passed by the caller and returns the sum of the digits of that number. You may assume that the number is non-negative.

Example:
A nonnegative number: 29107 will return 19 (i.e., 2+9+1+0+7)

Hint: Use the % operator to extract the last digit of a number.

Public static int DigitSum(int n){
    int sum = 0;
    while (n > 0){
        sum += n % 10; // add last digit to sum
        n = n / 10; // remove last digit
    }
    return sum;
}

c. (12 points) Write a static method named printGrid that accepts two integer parameters rows and cols. The output is a comma-separated grid of numbers where the first parameter (rows) represents the number of rows of the grid and the second parameter (cols) represents the number of columns. The numbers count up from 1 to (rows x cols). The output are displayed in column-major order, meaning that the numbers shown increase sequentially down each column and wrap to the top of the next column to the right once the bottom of the current column is reached. You may assume that both parameters passed to your method are greater than 0.

Here are some example calls to your method and their expected results:

<table>
<thead>
<tr>
<th>Call</th>
<th>printGrid(3,6);</th>
<th>printGrid(5,3);</th>
<th>printGrid(4,1);</th>
<th>printGrid(1,3);</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output</td>
<td>1, 4, 7, 10, 13, 16</td>
<td>1, 6, 11</td>
<td>1</td>
<td>1, 2, 3</td>
</tr>
<tr>
<td></td>
<td>2, 5, 8, 11, 14, 17</td>
<td>2, 7, 12</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3, 6, 9, 12, 15, 18</td>
<td>3, 8, 13</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4, 9, 14</td>
<td>4, 14</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5, 10, 15</td>
<td>5, 10, 15</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
d. (12 points) Write a static method that takes an integer badLength and a String array words as
arguments. Your method should find all of the Strings in the array that have a length equal to
badLength. For each one of those Strings, it should change that array position to be just the first half
of the word. For example, if words is {"harry", "potter", "is", "a", "wizard"} and badLength is 6, then
the method should change words to be {"harry", "pot", "is", "a", "wiz"}.

```java
public static void process (int badLength, String [] words){
    for (int i=0; i<words.length; i++){
        if(words[i].length()==badLength)
            words[i]=words[i].substring(0,badLength/2);
    }
}
```

3) Challenge Problem (18 points, 25 minutes)

Complete the Month class. The class should have a private integer field named monthNumber that holds
the number value of the month. For example, January would be 1, February would be 2, and so forth. In
additional, you must provide the following methods:

- A no-arg constructor that sets the monthNumber filed to a default 1.
- A constructor that accepts the number of the month as an argument. It should set the
  monthNumber filed to the value passed as the argument. If a value is out of range [1..12],
  the constructor should set to 1.
- A constructor that accepts the name of the month, such as “January” or “Feburay” as an
  argument. It should set the monthNumber field to the correct corresponding value.
- A setMonthNumber mutator that accepts an integer argument, which is assigned to the
  monthNumber field. If a value is out of range, the method should use 1 as the default
  value.
- A getMonthNumber accessor that returns the integer value in the monthNumber field.
- A toString accessor that returns the name of the month. For example, if the
  monthNumber is 1, then this method should return String “January”, not 1!
- An equals method that accepts a Month object as an argument. If the argument object
  holds the same data as the calling object, this method should return true. Otherwise, it
  should return false.

Then this Month class should support the following application class Date:
public class Date{
    public static void main(String[] args){
        Month a = new Month(), b = new Month(1);
        Month c = new Month("January"), d = new Month("JANUARY");
        if(a.equals(b)) System.out.println("Same number value.");
        if(c.equals(b) && c.equals(d))
            System.out.println("String has the same number value.");
        a.setMonthNumber(-1);
        System.out.println("The value after set (-1) is: "+a.getMonthNumber());
        System.out.println("The string value will be: "+a.toString());
    }
}

public class Month{
    private int monthNumber; // 1.5 points
    public Month(){ // 1.5 points
        monthNumber = 1;
    }
    public Month(int i){ // 1 point
        // cannot call setMonthNumber, must be private method to call // 0.5 point
        if (i>=1 && i<=12)
            monthNumber = i;
        else
            monthNumber = 1;
    }
    public Month(String a){ // 4 point
        switch (a.toUpperCase())
        {
            case "FEBRUARY":
                monthNumber = 2;
                break;
            case "MARCH":
                monthNumber = 3;
                break;
            case "APRIL":
                monthNumber = 4;
                break;
            case "MAY":
                monthNumber = 5;
                break;
            case "JUNE":
                monthNumber = 6;
                break;
            case "JULY":
                monthNumber = 7;
                break;
            case "AUGUST":
                monthNumber = 8;
                break;
            case "SEPTEMBER":
                monthNumber = 9;
                break;
            case "OCTOBER":
                monthNumber = 10;
                break;
        }
    }
}
case "NOVEMBER":
    monthNumber = 11;
    break;
  case "DECEMBER":
    monthNumber = 12;
    break;
  default:
    monthNumber = 1;
    break;
}

public void setMonthNumber(int i){ // 2 points
    if (i>=1 && i<=12)
        monthNumber = i;
    else
        monthNumber =1;
}

public int getMonthNumber (){  // 1.5 points
    return monthNumber;
}

public String toString(){  // 4 points
    String result;
    if (monthNumber == 2)
        result = " FEBRUARY ";
    else if (monthNumber == 3)
        result = "MARCH";
    else if (monthNumber == 4)
        result = "APRIL";
    else if (monthNumber == 5)
        result = "MAY";
    else if (monthNumber == 6)
        result = "JUNE";
    else if (monthNumber == 7)
        result = "JULY";
    else if (monthNumber == 8)
        result = "AUGUST";
    else if (monthNumber == 9)
        result = "SEPTEMBER";
    else if (monthNumber == 10)
        result = "OCTOBER";
    else if (monthNumber == 11)
        result = "NOVEMBER";
    else if (monthNumber == 12)
        result = "DECEMBER";
    else
        result = "JANUARY";

    return result;
}

public boolean equals ( Month right){ // 2 points
    return (monthNumber == right.getMonthNumber());
}