Instructions: This is an open note exam and you have 75 minutes to complete the test. The exam is worth 120 points (12% of your total grade). There are four questions. Write your answers as neatly as you can and write them in the space provided. You are not allowed to access any electronic devices (computers, calculators, phones etc.) during the exam.

Name:

Problem 1 (20 points):

Write a method named `verifyEmail` that takes a string as the only parameter and checks if the string is a valid email address. The method returns `true` if the string is found to be a valid email address, and `false` otherwise. Assume that a valid email address has two parts: username and domain, separated by one @ sign. For example, if the username is “john-smith” and domain is “uiowa.edu”, the complete email address would be “john-smith@uiowa.edu”. For this problem you should make the following assumptions:

1. A username can have only letters, digits, dots, underscores and dashes.
2. A domain can have only letters and dot(s).
3. A domain can have multiple dots, but cannot have two consecutive dots. Also, a domain cannot end with a dot.
4. An email address is not case sensitive.

The following table shows what your function should return for different parameters.

<table>
<thead>
<tr>
<th>Method call</th>
<th>Value returned</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>verifyEmail(&quot;john-smith@uiowa.edu&quot;)</code></td>
<td><code>true</code></td>
</tr>
<tr>
<td><code>verifyEmail(&quot;john-smith123@uiowa.edu&quot;)</code></td>
<td><code>true</code></td>
</tr>
<tr>
<td><code>verifyEmail(&quot;john-smith@uiowa.edu&quot;)</code></td>
<td><code>false</code></td>
</tr>
<tr>
<td><code>verifyEmail(&quot;john-smith@uiowa2.edu&quot;)</code></td>
<td><code>false</code></td>
</tr>
<tr>
<td><code>verifyEmail(&quot;john-smith@uiowa.edu&quot;)</code></td>
<td><code>false</code></td>
</tr>
<tr>
<td><code>verifyEmail(&quot;john-smith@abc.com.us.&quot;)</code></td>
<td><code>false</code></td>
</tr>
<tr>
<td><code>verifyEmail(&quot;john-smith@uiowa..edu&quot;)</code></td>
<td><code>false</code></td>
</tr>
<tr>
<td><code>verifyEmail(&quot;john@smith@uiowa.edu&quot;)</code></td>
<td><code>false</code></td>
</tr>
</tbody>
</table>
public static boolean verifyEmail(String e) {
    e = e.toLowerCase(); // email address is not case sensitive
    String[] parts = e.split("@");
    if (parts.length != 2 || parts[0].length() == 0 || parts[1].length() == 0) {
        return false;
    }
    char currentChar;
    // process hostname part
    for (int i = 0; i < parts[0].length(); i++) {
        currentChar = parts[0].charAt(i);
        if (Character.isLetter(currentChar) == false &&
            Character.isDigit(currentChar) == false &&
            currentChar != '.' && currentChar != '_' && currentChar != '-')
            return false;
    }
    // process the domain part
    if (parts[1].contains("..") || parts[1].endsWith(".")
        || parts[1].contains(".") == false) {
        return false;
    }
    for (int i = 0; i < parts[1].length(); i++) {
        currentChar = parts[1].charAt(i);
        if (Character.isLetter(currentChar) == false && currentChar != '.'){
            return false;
        }
    }
    return true;
}
Problem 2 (35 points):

You may find the following definition from Wikipedia useful for this problem.

**Matrix addition**: The usual matrix addition is defined for two matrices of the same dimensions. The sum of two \( m \times n \) (pronounced "m by n") matrices \( A \) and \( B \), denoted by \( A + B \), is again an \( m \times n \) matrix computed by adding corresponding elements.

\[
A + B = \begin{bmatrix}
a_{11} & a_{12} & \cdots & a_{1n} \\
a_{21} & a_{22} & \cdots & a_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
a_{m1} & a_{m2} & \cdots & a_{mn}
\end{bmatrix}
+ \begin{bmatrix}
b_{11} & b_{12} & \cdots & b_{1n} \\
b_{21} & b_{22} & \cdots & b_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
b_{m1} & b_{m2} & \cdots & b_{mn}
\end{bmatrix}
= \begin{bmatrix}
a_{11} + b_{11} & a_{12} + b_{12} & \cdots & a_{1n} + b_{1n} \\
a_{21} + b_{21} & a_{22} + b_{22} & \cdots & a_{2n} + b_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
a_{m1} + b_{m1} & a_{m2} + b_{m2} & \cdots & a_{mn} + b_{mn}
\end{bmatrix}
\]

For example:

\[
\begin{bmatrix}1 & 3 \\1 & 0 \\1 & 2 \end{bmatrix} + \begin{bmatrix}0 & 0 \\7 & 5 \\2 & 1 \end{bmatrix} = \begin{bmatrix}1+0 & 3+0 \\1+7 & 0+5 \\1+2 & 2+1 \end{bmatrix} = \begin{bmatrix}1 & 3 \\8 & 5 \\3 & 3 \end{bmatrix}
\]

You are given an incomplete program (Matrix.java) with few methods unimplemented. Your task is to provide code for those methods to make the program compile and generate the given output. Method descriptions are given as comments before the method signatures in the program.

Here is the incomplete program Matrix.java:

```java
public class Matrix {
    private int m, n; // number of rows and columns
    private double[][] data; // 2d array for a m-by-n matrix

    // create a m-by-n matrix of zeros
    public Matrix(int m, int n) {
        // implement this method
        this.m = m;
        this.n = n;
        data = new double[m][n];
    }
}
```
// Create a m-by-n matrix based on a 2d array.
// The matrix created should copy the elements from the given 2d array
public Matrix(double[][] data) {
    //implement this method
    m = data.length;
    n = data[0].length;
    this.data = new double[m][n];
    for (int i = 0; i < m; i++)
        for (int j = 0; j < n; j++)
            this.data[i][j] = data[i][j];
}

// Copy constructor
public Matrix(Matrix m) {
    //implement this method
    this(m.data);
}

// Swap rows i and j of the invoking matrix
private void swapRows(int i, int j) {
    //implement this method
    double[] temp = data[i];
    data[i] = data[j];
    data[j] = temp;
}
// Check if the given matrix m1 has the same
dimension (i.e. same number of rows and columns) as the invoking matrix.
// Return null if the dimensions are not the same.
// Add matrix m1 and the invoking matrix if the dimensions are same,
// and return the resultant matrix.
public Matrix plus(Matrix m1) {
    //implement this method
    if (m1.m != this.m || m1.n != this.n)
        return null;
    Matrix mat = new Matrix(m, n);
    for (int i = 0; i < m; i++)
        for (int j = 0; j < n; j++)
            mat.data[i][j] = this.data[i][j] + m1.data[i][j];
    return mat;
}

//Return true if the given matrix m1 is exactly same as the invoking matrix.
//Return false otherwise.
public boolean isEqual(Matrix m1) {
    //implement this method
    if (m1.m != this.m || m1.n != this.n)
        return false;
    for (int i = 0; i < m; i++)
        for (int j = 0; j < n; j++)
            if (this.data[i][j] != m1.data[i][j])
                return false;
    return true;
}

// print matrix
public void printMatrix() {
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++)
            System.out.printf("%6.2f ", data[i][j]);
        System.out.println();
    }
}
// main method
public static void main(String[] args) {
    double[][] d = { { 1, 2, 3 }, { 4, 5, 6 }, { 7, 8, 9 }};
    Matrix mat1 = new Matrix(d);
    mat1.printMatrix();
    System.out.println();

    Matrix mat2 = new Matrix(d);
    mat2.swapRows(1, 2);
    mat2.printMatrix();
    System.out.println();

    mat1.plus(mat2).printMatrix();
    System.out.println();

    System.out.println(mat1.isEqual(mat2));
}

Desired output:
1.00 2.00 3.00
4.00 5.00 6.00
7.00 8.00 9.00

1.00 2.00 3.00
7.00 8.00 9.00
4.00 5.00 6.00

2.00 4.00 6.00
11.00 13.00 15.00
11.00 13.00 15.00

False
**Problem 3 (30 points):**

Check the following Java programs carefully and list as many errors (both compile-time and run-time) as you can. While listing errors, you should indicate the line number for statement that is the source of error. You should also provide a short description of the errors.

(a) Source file **MyClass.java**: (10 points)

```java
public class MyClass {
    final static int[] arr = new int[10];
    void sayHello(){
        System.out.println("Hello!");
    }
    static class InnerClass{
        InnerClass(){
            int a;
        }
    }
    public static void main(String[] args) {
        arr[0] = 10;
        for (int i=0; i<=10; i++){
            System.out.println(arr[i]);
        }
        MyClass myClass = new MyClass();
        MyClass.sayHello();
        myClass.InnerClass inner = new MyClass.InnerClass();
    }
}
```

List the errors here (line number and short description):

**Line 13:** Array index `i` out of bound, `i` cannot be 10.
**Line 16:** Instance method `sayHello()` cannot be referenced without an object. (`MyClass` is not an object)
**Line 17:** `InnerClass` is a static nested class, which needs to be referenced by using the class name (not an object).

Note that line 11 is not an error-- the reference is final here, not the array.
(b) Source file Y.java: (5 points)

```java
public class Y {
    class X extends Y {
        void f2() {}
        public static void main(String args[]) {
            Y y = new X();
            y.f2();
        }
    }
}
```

List the error(s) here:

**Line 6:** y needs to be type casted to X to get access to the method f2.

Give the corrected code to make this program compile:

**At line 6:** ((X)y).f2();
(c) Source file Test.java: (15 points)

```java
1  Public interface MyInterface {
2      int a;
3      void func1(int i, String s);
4      void func2(int i);
5  }
6  
7  Public interface MyInterface1 {
8      void func3(int i, String s);
9      void func4(int i);
10  }
11  abstract class MyClass implements MyInterface {
12      
13      void func5(int i, String s);
14      void func6(int i) {}
15      void func1(int i, String s){}
16      final void func7(){}
17      MyClass(String s){}
18  }
19  class Test extends MyClass implements MyInterface1 {
20      
21      void func2(int i) {}
22      void func3(int i, String s) {}
23      void func4(int i) {}
24      void func5(int i, String s) {}
25      void func6(int i) {}
26      void func7(){
27      Test() {}
28      Test(MyInterface1 mf) {}
29      
30      public static void main(String[] args) {
31          Test t1, t2;
32          t1 = new Test(new Test());
33          t2 = new Test(new MyClass());
34      }
35  }
```

List the errors here (line number and short description):

- **Line 1 and 7:** Cannot be public. Check the source file name.
- **Line 19:** The class Test needs to be public, since the source file name is Test.java
- **Line 2:** needs an assignment, (e.g. `a = 10;`)
- **Line 13:** missing method body, or needs to be declared as abstract
- **Line 15:** `func1` in interface `MyInterface` is public. An implementation or overridden method cannot have weaker access. This method should be public here as well.
- **Line 20, 21, 22:** needs to be public, for the same reason that is described for line 15
- **Line 25:** final method cannot be overridden
- **Line 26:** class `MyClass` does not have a no parameter constructor
- **Line 31:** `MyClass` is an abstract class, cannot be instantiated
Problem 4 (35 points):
Write the output of the following Java programs:
(a) Program **Cube.java** (15 points)

```java
public class Cube {
    static int n = 1;
    int side;
    Cube(int n) {
        side = n;
    }
    int getVolume() {
        return side * side * side;
    }
    public static void main(String[] args) {
        Cube[] cubes = new Cube[5];
        Cube cube = new Cube(n);
        for (int i = 0; i < cubes.length; i++) {
            cubes[i] = cube;
            n++;
            cubes[i].side = n;
            System.out.println(cubes[i].getVolume());
        }
        System.out.println(cube.getVolume());
        for (Cube c : cubes) {
            System.out.println(c.getVolume());
        }
    }
}
```

What is the Output?

8
27
64
125
216
216
216
216
216
216
216
216
(b) Program `SuperClass.java` (20 points)

```java
public class SuperClass {
    static int a, b;
    static {
        a = b = 2;
        for (int i = 0; i < 3; i++) {
            b += a;
        }
    }
    int num1 = 5;
    SuperClass() {
        a = 5;
    }
    SuperClass(int n) {
        a = n;
    }
    void method1() {
        System.out.println("Super method1");
    }
    static void method2() {
        System.out.println("Super method2");
    }
}

class SubClass extends SuperClass {
    SubClass() {
        this(10);
        a = 15;
    }
    SubClass(int n) {
        super(n);
        a = 20;
    }
    void method1() {
        System.out.println("Sub method1");
    }
    static void method2() {
        System.out.println("Sub method2");
    }
    public static void main(String[] args) {
        SuperClass sup = new SuperClass();
        SubClass sub = new SubClass();
        sup = sub;
        sub.method1();
        sup.method1();
        SuperClass.method2();
        SubClass.method2();
        System.out.println(sub.num1 + " " + SuperClass.a + " " + SuperClass.b);
    }
}

What is the output?
Sub method1
Sub method1
Super method2
Sub method2
5 15 8
```