**CIS 255 - Java Programming**
**Exam #3**

**Name:** _____

**Date:** _____

**1.** (11 points) **Given the declared and initialized LinkedList object below, display all the values in the list after each of the statement(s) is/are executed. If the list is empty, then write "empty". If the statement causes an error, then write "error". (Each statement builds on the one before)**

**List<String> list = new LinkedList<String>();**

**a.)**
```
list.add("Jim");
```

**b.)**
```
list.add("Bill");
```

**c.)**
```
list.add("Karen");
```

**d.)**
```
list.remove(3);
```

**e.)**
```
list.add("Jim");
```

**f.)**
```
list.add("Sam");
```

**g.)**
```
list.remove("Jim");
```

**h.)**
```
list.clear();
```

**i.)**
```
list.set(0, "Cindy");
```

**j.)**
```
List<String> fix =
        new LinkedList<String>();

fix.add("Adam");
list.addAll(fix);
```

**k.)**
```
list.add("Adam");
```

**2.** (9 points) **Given the declared and initialized LinkedHashMap object below, display all the keys and values in the map after each of the statement(s) is/are executed. If the map is empty, then write "empty". If the statement causes an error, then write "error". (Each statement builds on the one before)**

**Map<String, String> map= new LinkedHashMap<String, String>();**

**a.)**
```
map.put("AL", "Alabama");
```

**b.)**
```
map.put("NY", "New York");
```

**c.)**
```
Map<String, String> fix1 =
      new LinkedHashMap<String, String>();

    fix1.put("HI", "Hawaii");
    fix1.put("TN", "Tennessee");
    map.putAll(fix1);
```

**d.)**
```
map.remove("Al");
```

**e.)**
```
map.clear();
```

**f.)**
```
map.put("Tn", "Tennessee");
```

**g.)**
```
map.remove("AK");
```

**h.)**
```
map.put("AK", "Alaska");
```

**i.)**
```
Map<String, String> fix2 = null;

   fix2.put("AK", "Hawaii");
   fix2.put("TN", "Tennessee");
   map.putAll(fix2);
```

**3. Given the following Certification object class, answer the questions that follow.**

```
package edu.calhoun.cis.java.intro.exam.exam3;

import java.util.Date;

public class Certification {
    private String certName = null;
    private double examLengthInHours = 0;
    private Date issueDate = null;

    public Certification() {
    }

    public Certification(String certName, double examLengthInHours, Date issueDate) {
        this.certName = certName;
        this.examLengthInHours = examLengthInHours;
        this.issueDate = issueDate;
    }

    public String getCertName() {
        return certName;
    }

    public void setCertName(String certName) {
        this.certName = certName;
    }

    public double getExamLengthInHours() {
        return examLengthInHours;
    }

    public void setExamLengthInHours(int examLengthInHours) {
        this.examLengthInHours = examLengthInHours;
    }

    public Date getIssueDate() {
        return issueDate;
    }

    public void setIssueDate(Date issueDate) {
        this.issueDate = issueDate;
    }

    public void showLevelsForDoD() {
        System.out.println("Not qualified to work for DoD");
    }

    @Override
    public String toString() {
        return "SecurityCertification [certName=" + certName
                + ", examLength=" + examLengthInHours
                + ", issueDate=" + issueDate + "]";
    }
}
```

**a.** (10 points) **Write a Java class SecurityPlus that that extends the Certification object.**

- **Ensure this class resides in the package edu.calhoun.cis.java.intro.exam.exam3.**
- **Add the member variable yearsExperience as an integer.**
- **Add getters and setters for the member variables.**
- **Override the showLevelsForDoD() method to output "IAT Level II, IAM Level I".**
- **The SecurityPlus object class should override the toString() method and display all variables (including the inherited ones).**

**b.** (10 points) **Write a Java class CISSP that that extends the Certification object.**

- **Ensure this class resides in the package edu.calhoun.cis.java.intro.exam.exam3.**
- **Add the member variables yearsExperience as an integer and attendedBootcamp as a boolean.**
- **Add getters and setters for the member variables.**
- **Override the showLevelsForDoD() method to output "IAT Level III, IAM Level III, IASAE II".**
- **The CISSP object class should override the toString() method and display all variables (including the inherited ones).**

**c.** (10 points) **Write the Java class Exam3Q3 to test the Certification, SecurityPlus, and CISSP classes.**

- **Ensure class Exam3Q3 resides in the package edu.calhoun.cis.java.intro.exam.exam3.**
- **Exam3Q3 should create 1 instance of the Certification object using the default constructor.**
  - **Output the created Certification object's values to the screen using a System.out.println().**
  - **After displaying the Certification object's data, execute the showLevelsForDoD() method.**
- **Exam3Q3 should create 1 instance of the SecurityPlus object using the overloaded constructor.**
  - **Pass the following values to the constructor:**
    - `"Security+"`
    - `1.5`
    - `new Date()`
    - `2`
  - **Output the created SecurityPlus object's values to the screen using a System.out.println().**
  - **After displaying the Certification object's data, execute the showLevelsForDoD() method.**
- **Exam3Q3 should create 1 instance of the CISSP object using the overloaded constructor.**
  - **Pass the following values to the constructor:**
    - `"CISSP"`
    - `6.0`
    - `new Date()`
    - `11`
    - `true`
  - **Output the created CISSP object's values to the screen using a System.out.println().**
  - **After displaying the Certification object's data, execute the showLevelsForDoD() method.**

**4. Given the following Clearance interface, answer the questions that follow.**

```
package edu.calhoun.cis.java.intro.exam.exam3;

public interface Clearance {
    public void showAcronym();
    public void showDescription();
    public void showImpactIfMadePublic();
}
```

**a.** (10 points) **Write a Java class Confidential that that implements the Clearance interface.**

- **Ensure class Confidential resides in the package edu.calhoun.cis.java.intro.exam.exam3.**
- **Add the private member variables for** `acronym`**,** `description`**, and** `impactIfMadePublic`**.**
- **Create the default constructor method which sets the member variables to the following values:**
  - **`acronym =`** `"C"`
  - **`description =`** `"Confidential"`
  - **`impactIfMadePublic =`** `"Damage or be prejudicial to national security"`
- **Implement the showAcronym(), showDescription(), and showImpactIfMadePublic() methods using a System.out.println() to output the appropriate variables to the screen.**

**b.** (10 points) **Write a Java class Secret that that implements the Clearance interface.**

- **Ensure class Secret resides in the package edu.calhoun.cis.java.intro.exam.exam3.**
- **Add the private member variables for** `acronym`**,** `description`**, and** `impactIfMadePublic`**.**
- **Create the default constructor method which sets the member variables to the following values:**
  - **`acronym =`** `"S"`
  - **`description =`** `"Secret"`
  - **`impactIfMadePublic =`** `"Serious damage to national security"`
- **Implement the showAcronym(), showDescription(), and showImpactIfMadePublic() methods using a System.out.println() to output the appropriate variables to the screen.**

**c.** (10 points) **Write a Java class TopSecret that that implements the Clearance interface.**

- **Ensure class TopSecret resides in the package edu.calhoun.cis.java.intro.exam.exam3.**
- **Add the private member variables for** `acronym`**,** `description`**, and** `impactIfMadePublic`**.**
- **Create the default constructor method which sets the member variables to the following values:**
  - **`acronym =`** `"TS"`
  - **`description =`** `"Top Secret"`
  - **`impactIfMadePublic =`** `"Exceptionally grave damage to national security"`
- **Implement the showAcronym(), showDescription(), and showImpactIfMadePublic() methods using a System.out.println() to output the appropriate variables to the screen.**

**d.** (10 points) **Write the Java class Exam3Q4 to test the classes that implement the Clearance interface.**

- **Ensure class Exam3Q4 resides in the package edu.calhoun.cis.java.intro.exam.exam3.**
- **Exam3Q4 should create 1 Confidential object, 1 Secret object, and 1 TopSecret object.**
- **Each Clearance object after creation should be added to a List that will contain Clearance objects.**
- **Loop through the list and do the following at each iteration:**
  - **Execute the interface method** `showAcronym()`
  - **Execute the interface method** `showDescription()`
  - **Execute the interface method** `showImpactIfMadePublic()`

**5. Given the following text file, ATMOSPHERES.DAT:**

```
1 Troposphere 0 7
2 Stratosphere 7 31
3 Mesosphere 31 to 50
4 Thermosphere 50 440
```

**And the following Atmosphere Java class, answer the questions that follow:**

```
package edu.calhoun.cis.java.intro.exam.exam3;

public class Atmosphere {
    private String id = null;
    private String name = null;
    private String startHeightInMiles = null;
    private String endHeightInMiles = null;

    public Atmosphere() {
    }

    public Atmosphere(String id, String name,
            String startHeightInMiles, String endHeightInMiles) {
        this.id = id;
        this.name = name;
        this.startHeightInMiles = startHeightInMiles;
        this.endHeightInMiles = endHeightInMiles;
    }

    public String getId() {
        return id;
    }

    public void setId(String id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getStartHeightInMiles() {
        return startHeightInMiles;
    }

    public void setStartHeightInMiles(String startHeightInMiles) {
        this.startHeightInMiles = startHeightInMiles;
    }

    public String getEndHeightInMiles() {
        return endHeightInMiles;
    }

    public void setEndHeightInMiles(String endHeightInMiles) {
        this.endHeightInMiles = endHeightInMiles;
    }
```

```
    @Override
    public String toString() {
        return "Atmosphere [id=" + id
                + ", name=" + name
                + ", startHeightInMiles=" + startHeightInMiles
                + ", endHeightInMiles=" + endHeightInMiles + "]";
    }
}
```

**a.** (10 points) **Write the Java class Exam3Q5.**

- **Ensure class Exam3Q5 resides in the package edu.calhoun.cis.java.intro.exam.exam3.**
- **Create method readAtmospheres(list) in the Exam3Q5 class that that reads the file ATMOSPHERES.DAT.**
    i. **Each line in the data file will be stored into an Atmosphere Java class object and added to a list of Atmosphere objects.**
- **Once the file has been read and saved into a list of Atmosphere objects, create another method displayAtmospheres(list) which iterates the list and displays the information for each atmosphere.**
- **Add the method writeAtmosphere() to the Exam3Q5 class that that writes the Atmosphere objects to a new data file ATMOSPHERES_NEW.DAT.**
- **Before writing the atmospheres to the new file, add a new Atmosphere object to the list by creating the method addAtmosphere(List<Atmosphere> atmospheres, String id, String name, String startHeightInMiles, String endHeightInMiles) and passing it the following values:**

    **ID:** 5
    **Name:** Exosphere
    **Start Height in Miles:** 440
    **End Height in Miles:** 6200

- **Show the output generated by executing this program.**
- **Outside of the program, copy and paste the contents of the newly created ATMOSPHERES_NEW.DAT to illustrate a successful write.**