**iLAB OVERVIEW**

**Scenario/Summary**

In this lab, you will create one project that reads from a file, one project that writes to a file, and one project drawing a snowman.

**Deliverables**

Program files for each of the following programs.

1. Write out client information
2. Read in client information
3. Draw a snowman

At the beginning of all your programs, put a comment box that includes the program name, your name, and a brief description of the program.

**Example:**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
Program Name: ProgramName.java
Programmer's Name: Student Name
Program Description: Describe here what this program will do
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**How to submit your assignment:**

1. The programs *must* have the same names as the assignment title.
2. Each Java source file (\*.java) must include a corresponding class file (\*.class) program as evidence of success.
3. In addition to the program source code files and byte code files, put all your program source code files and screen shots of your program output files into a Word document.
4. You must use a zipped folder to send your weekly assignment to the Dropbox. Do not send subfolders within your zipped folder. Place all of the .java and .class files for the week into the one zipped folder. The zip folder should be named CIS355A\_YourLastName\_iLab\_Week5, and this zip folder will contain all the weekly programming assignments.

**Required Software**

**Eclipse**

Access the software at [https://lab.devry.edu](https://lab.devry.edu/).
Steps: 1, 2, and 3

**iLAB STEPS**

**STEP 1: Writing out Client Information**

[Back to Top](https://devry.equella.ecollege.com/file/9097403e-52ff-40f5-af1a-ded82e0a807a/33/CIS355A_W5_iLab.html#top)

1)      Create the following GUI, so that when your program is running, your user can input information regarding a client and hit the save button to save the information out to a file.

2)  Every time the user hits the save button, that information should be saved out to a file called client.txt; each new client's information should append to the information already saved onto the file client.txt.

3) The data in the client.txt file should be formatted like the following.

|  |
| --- |
| **Client Activity Report** |
| **Client Name**  |  **Client ID**  |  **Starting Balance**  |  **Closing Balance** |
| **XXXXXXXXX**  |  **9999999**  |  **99999.99**  |  **99999.99** |
| **XXXXXXXXX**  |  **9999999** |  **99999.99**  |  **99999.99** |
| **XXXXXXXXX**  |  **9999999**  |  **99999.99** |  **99999.99** |
| **Writing Client Information** | **Points** | **Description** |
| **Standard header included** | 1 | Must contain program name, student’s name, and description of the program |
| **Program Compiles** | 2 | Program does not have any error |
| **Program Executes** | 2 | Program runs without any error |
| **Data Output is Correct** | 5 | The data is written and formatted correctly |
| **GUI is Correct** | 5 | The GUI has the correct fields and displays information in correct format |
| **Subtotal** | 15 |   |

**STEP 2: Reading in Client Information**

[Back to Top](https://devry.equella.ecollege.com/file/9097403e-52ff-40f5-af1a-ded82e0a807a/33/CIS355A_W5_iLab.html#top)

1)     Create a class called Client, the Client class must contain attributes for Client name, Client ID, starting balance, and closing balance, and all other accessor/mutator/constructor functions as necessary.

2)      Assume you have a client.txt file with the following sample information.

**Charles Smith|100235|5700.75|1200.00**

**James Peterson|320056|349.56|4005.56**

**Francis Lewis|400556|7500.00|456.23**

**William Burgess|45399|5000.00|1245.56**

**Philip Wilson|10090|10000.00|2300.75**

**James Brown|34291|25000.45|31454.86**

3)      Create a Client ArrayList to process input records in main().

4)      Use a *for* loop to read in the information from client.txt.

5)      The GUI to this program should look similar to this:

6)     Once the user hits the display button, everything read in from the file should display in the Console window in this format.

|  |
| --- |
| **Client Activity Report** |
| **Client Name**  |  **Client ID**  |  **Starting Balance**  |  **Closing Balance** |
| **XXXXXXXXX**  |  **9999999**  |  **99999.99**  |  **99999.99** |
| **XXXXXXXXX**  |  **9999999** |  **99999.99**  |  **99999.99** |
| **XXXXXXXXX**  |  **9999999**  |  **99999.99** |  **99999.99** |
| **Reading in Client Information** | **Points** | **Description** |
| **Standard header included** | 1 | Must contain program name, student’s name, and description of the program |
| **Program Compiles** | 2 | Program does not have any error |
| **Program Executes** | 2 | Program runs without any error |
| **Client ArrayList used** | 3 | Created an ArrayList to read in the records from client.txt |
| **Data Read in Correctly** | 3 | The data is read correctly and saved correctly using the correct attributes |
| **GUI is Correct** | 4 | The GUI has the correct fields and displays information in correct format |
| **Subtotal** | 15 |   |

**STEP 3: Snowman!**

[Back to Top](https://devry.equella.ecollege.com/file/9097403e-52ff-40f5-af1a-ded82e0a807a/33/CIS355A_W5_iLab.html#top)

Use the many draw methods provided to you by Java and draw a Snowman—be as creative or as basic as you would like, as long as the final result resembles a snowman. It doesn't have to necessarily look exactly like this, but this is the minimum you should achieve with your drawing.

1) You must have at least three circles in your project.

2) You must have at least a line, a polygon, an oval, or a rectangle.

3) In addition to your snowman, you should also use drawString to draw some text.

4) Use draw or fill and the color class as you see fit.

**Hint:** frame.getContentPane().setBackground(Color.blue); //This is the code you need to set the frame's background color.

**Have Fun!**

| **Snowman** | **Points** | **Description** |
| --- | --- | --- |
| **Standard header included** | 1 | Must contain program name, student’s name, and description of the program |
| **Program Compiles** | 1 | Program does not have any error |
| **Program Executes** | 1 | Program runs without any error |
| **Program Output** | 3 | Drawing resembles a Snowman |
| **Correct use of Classes/Methods** | 4 | Program contains background color, the shapes  asked for, and uses the drawString method |
| **Subtotal** | 10 |   |