MAT 2170: Laboratory 1

Key Concepts

The purpose of this lab is to acquaint you with the equipment and software in Old Main 3041 that you will be using this semester:

1. `NetBeans`: creating projects and programs
2. WEB: publishing and checking your personal web site
3. Syntax errors: recognition and correction
4. SUB-MIT: electronic submission of your finished lab

Software Overview

We will need several pieces of software to create Java programs and display applets for our labs this semester. All of these packages are available for your use in the Old Main 3041 computer lab*.

- The Java development kit (or JDK)
- An integrated development environment (or IDE) — *NetBeans*
- A Java-enabled web browser — *Safari*
- The Extended ACM Java Task Force library — *acmLibrary.jar*

Getting Ready

Check off each box as you complete the following tasks in the order they are given:

☐ 1. Complete Worksheet #1 prior to lab. Your professor may ask to see it during lab, so wait until the end of lab to turn it in.

☐ 2. Using your EIU email user id and password, login to your Math/CS account. If there is a red dot inside the NAME response box, wait until it disappears to login.

One Time Only — System Modifications & Obtaining Software

☐ 3. The scroll wheel on the mouse works contrary to most systems. To change back to the usual setting, under the apple icon at the top left of the screen, choose **System Preferences**, then click the mouse icon. Uncheck the box for: `scroll direction:  normal`, then close the window.

☐ 4. Fixing up **Finder**. With the **Finder** active, you should see `Finder` on the top menu bar. Click on **Finder** → **Preferences**.

- In **New Finder Window Show:**, select your home directory (it has your user id).
- Click on the **Sidebar** button, then uncheck `All My Files`, and check your home directory. Close the **Preferences** window.

☐ 5. Obtain software to facilitate electronic submission — One time only.

- Open a new Finder window (Command-n).
- In the sidebar on the Finder window's left, locate `Applications` and click on it. You should see multiple files and folders, beginning with `Affinic Debugger`, `App Store`, etc.

*This software is available at no cost from several sources on the Internet. The Java JDK and the IDE known as NetBeans is made available by Oracle; the browser we use is Safari, developed by Apple Inc. and included with the Mac OS X and iOS operating systems. The extended ACM Java libraries are available on our course web site.

- Locate the applications MAKEHTML and SUBMIT (icons shown at right), and drag them to the toolbar along the top edge of the Finder window, so they appear as shown with the search bar adjacent to the right:

WEB SUB MIT

☐ 6. The ACM Java Libraries — One time only.
The Java programs described in our textbook use the ACM Java libraries to simplify the task of writing programs with Java. We have extended this library with some locally written programs and other files needed to work with the Finch, and are making this modified file available on our web site. To use these libraries, some initial steps are needed.

☐ (a) Create a directory named `www` inside a `mat2170` directory within your home directory.

- Go to your home directory in the Finder window.
- Create a new `mat2170` directory by right-clicking an empty space in the window and choosing **New Folder**, and type `mat2170`. Press return to make the new name permanent.
- To give you quick access in the future, drag the `mat2170` folder onto the Finder window's sidebar under `Favorites`.
- Click on this new link, going to your **mat2170** folder. Since it's brand new, it should be currently empty.
- Create a new folder inside the `mat2170` folder and name it `www` (all lower–case).

☐ (b) Download the `acmLibrary.jar` file from the course web site and place it in your `www` directory.

- Locate the `Safari` icon (a blue compass) near the center of your dock and click on it to start the browser. If you are given the option to enable a browser extension, do so. In `Safari`, go to our course website: `www.eiu.edu/~mathcs` and click on the `mat2170` link.
- Find the link to the `acmLibrary.jar` file in the `Labs` section under **Laboratory 1**.
- Download it by right-clicking the link and choosing DOWNLOAD LINKED FILE AS.... Click the ▽ to expand your options, and navigate to your `www` directory. Clicking SAVE will place a copy of the `acmLibrary.jar` file in the `mat2170/www` folder of your home directory.
- Confirm you have obtained the file by clicking on the `www` folder.

☐ 7. Turn off a distracting feature in `NetBeans`, and open a display panel — One time only.

- Start `netbeans`: locate the bluish–silver cube on the dock at the bottom of the screen (toward the right end). When you mouse over it, `NetBeans 7.3.1` should display. Click on the cube.

  You may be asked if you wish to participate in automatic user feedback — simply answer NO. If you are asked to register NetBeans, click on NEVER REGISTER. The first time you run NetBeans, a number of files will be created, so this may take a few moments.

- Autocompletion can be a very useful feature, but at this juncture it is too distracting. Within the NetBeans IDE, select **NetBeans→Preferences→Editor**, then the **Code Completion** tab. Uncheck the "Auto Popup Completion Window" and "Auto Popup Documentation Window." Click **OK** to continue.
- From the `NetBeans` menu bar, select **Window** → **Output** → **Output** to open a display panel below the editor pane.

Programming Exercises

Read and follow the instructions in the *Creating Java Programs and Maintaining a Web Site* handout for each of the exercises below. This writeup will take you through the steps of how to create simple Java programs and put their corresponding applets on your personal website. Keep it as a handy reference for future labs.

1. **HelloProgram**. Traditionally, the first program one writes in a new language should cause "Hello, World!" to appear when the program is executed. The name for this project and program is `HelloProgram`. Following the instructions from the **Creating Java Programs** Handout to:

   - ☐ Create a `HelloProgram` project located in `mat2170/lab1`
   - ☐ Add the `acmLibrary.jar` to the project
   - ☐ Create an empty Java file, using `HelloProgram` for the class name.
   - ☐ In **Safari**, on our course webpage, locate the `HelloProgram` link and click on it. The Java code should now appear in the browser window, as shown below in Figure 1. Select the entire text and copy it to the editor panel in `NetBeans`. Modify the program by replacing the author and date, then use RIGHT-CLICK and FORMAT in the `NetBeans` editor window to auto-format the code to improve readability. Always format your programs in this way before submitting them electronically.

     Very important: All programs you write for this class should begin with a header comment section which includes: your name, class section, lab and exercise number, file name, and the purpose of the program (a brief summary of what the program does).

   - ☐ Execute your program by right-clicking in the editor window and choosing `Run File`.
   - ☐ *Modify* the "hello world" Java program so it displays your name instead of `hello, world`. Run the project again to check that it is behaving properly.
   - ☐ *Modify* the program again to move your name closer to the top of the applet viewer. Check that your change worked.
   - ☐ *Insert* a second `add(new GLabel())` statement after the one already in the program. Have the displayed text show your hometown and major at a new location beneath (and not overwriting) your name. Run the program to ensure it is correct.
   - ☐ *Modify* the header comments to reflect the changes in your program. Make sure it still executes correctly, then re–format one last time.
   - ☐ Publish this program to the web. Check that this was done correctly by visiting your web site and following the `HelloProgram` link.
   - ☐ Close this project.

2. **Add2Integers**. In the first Programming Exercise, you went through the mechanical aspects of creating a Java program and publishing the result to your web site. To reinforce these steps, you will now go through this process again. For this second exercise, create a program which adds two integers, as shown below in Figure 2.

   - ☐ Create an `Add2Integers` project located in `mat2170/lab1`
   - ☐ Add the `acmLibrary.jar` to the project
   - ☐ Create an empty Java file using the name `Add2Integers`
   - ☐ As before, copy the program code (as it appears in Figure 2) from the `Add2Integers` link on the course web site. Be sure to modify it to include your name and course information within the comment section, as previously discussed.

```
/*
 * Author: Johann Bach
 *        MAT 2170, Section 2
 * Exercise: Lab 1, #1
 * Date: January 6, 2012
 * File: HelloProgram.java
 * Purpose: This program displays "hello, world"
 * on the screen.  It is inspired by the first
 * program in Brian Kernighan and Dennis Ritchie's
 * classic book, The C Programming Language.
 *
 */

import acm.graphics.*;
import acm.program.*;

public class HelloProgram extends GraphicsProgram
{
    public void run()
    {   // Create a phrase and display to the user
        add(new GLabel("hello, world", 100, 75));
    } // end of run()

    public static void main(String[] args){
        new HelloProgram().start(args);
    } // end of main()

} // end of class HelloProgram
```

Figure 1: A first Java program, stored in the file `HelloProgram.java` by *NetBeans*.

☐ • Execute your program — you will need to enter two integers, then the final window displays their sum. Run it several times with various inputs. Will it accept negative numbers? Alphabetic characters? Close the applet viewer window when finished with each execution.

☐ • *Replace* `ConsoleProgram` (near the top of the program) with `DialogProgram`, then rebuild and run the program again. What changed? Are the same results generated? Repair any errors in the program and execute again to make sure all is well. Check that your name and other information are included in the file. When finished, **re–format** the code and `Save` it.

☐ • Publish this program to the web, and check that this was done correctly.

☐ • Close this project.

☐ 3. **TemperatureConvert**. Write a program to convert temperature measurements from Fahrenheit to Celsius. Name this project and program `TemperatureConvert`.

☐ (a) After creating the new project and java file, enter the code as shown in Figure 3 in the editor window. Add header comments using your name, class section, Lab 1, Exercise 3, the date, filename, and purpose of the program.

☐ (b) Test the program several times to ensure it is producing the correct result. What are some good values to use when testing this program? Don't forget to format the file when you are satisfied that it is working correctly.

☐ (c) Publish this program to the web and check it.

☐ (d) Close this project.

```
/*
 * Author: Johann Bach
 *         MAT 2170, Section 1
 * Exercise: Lab 1, #2
 * Date: August 28, 2007
 * File: Add2Integers.java
 * Purpose: This program requests two integers
 * from the user, adds them together, and displays
 * the sum.
 */

import acm.program.*;

public class Add2Integers extends ConsoleProgram
{
    public void run()
    {  // prompt for and obtain two integers from the user,
       // then find and display their sum
       print("This program adds two integers.\n");
       int n1 = readInt("Enter first integer  : ");
       int n2 = readInt("Enter second integer : ");
       int total = n1 + n2;
       println("The total of " + n1 + " plus " + n2 + " is " + total + ".");
    } // end of run()

    public static void main(String[] args){
        new Add2Integers().start(args);
    } // end of main()

} // end of class Add2Integers
```

Figure 2: The second Java program, stored in the file `Add2Integers.java` by *NetBeans.*

```
import acm.program.*;

public class TemperatureConvert extends DialogProgram
{
    public void run()
    {  // Prompt for and obtain temperature in Fahrenheit from user,
       // convert it to Celsius using the formula C = 5/9(F-32),
       // and display the result as a Celsius equivalent
        print("This program converts Fahrenheit to Celsius.\n");
        double Fahrenheit = readDouble("Enter temperature in Fahrenheit  : ");
        double Celsius = 5.0 * (Fahrenheit - 32.0) / 9.0;
        println("The same temperature in degrees Celsius is " + Celsius + ".");
    } // end of run()

    public static void main(String[] args){
        new TemperatureConvert().start(args);
    } // end of main()

} // end of class TemperatureConvert
```

Figure 3: The third Java program, stored in the file `TemperatureConvert.java` by *NetBeans.*

☐ 4. `RosePoem`. Create a `RosePoem` project (refer to Worksheet #1) as you did for your first three programs. It will be quite similar to `HelloProgram`. Test the program, and when you are satisfied it is correct, publish it. Print this program: in `NetBeans`, select **File** → **print**. The printer is located in the hallway outside our lab. You will need your Panther card and 7 cents for each page printed. Hand in your printed output at the beginning of Lab 2.

☐ It is very likely that you will have sufficient time to complete Lab 1 today. This will not always be the case, and you are expected to complete labs on your own time when this happens. When all exercises are completed, drag the `lab1` folder to the SUB-MIT icon at the top of a Finder window, as described in the *Creating Java Programs* handout.

☐ Hand in Worksheet #1 before you leave lab.
Hand in any printouts from this lab at the beginning of the next lab, Lab 2. Worksheet 2 will be due at the end of that lab.

☐ **Always log out of your account** before you leave lab. Failure to do so may result in catastrophic loss of files, or worse.