

MAT 2170: Laboratory 1

Key Concepts

The purpose of this lab is to give you a chance to become acquainted with the equipment and software in Old Main 3041 that we will be using this semester.

1. Mac OS-x: file structure and commands
2. Netbeans: creating projects, programs, and applets
3. Web page: creation and updating
4. Syntax errors: recognition and correction

Exercises

Check off each box as you complete the following exercises:

1. Complete the Prelab exercises and be ready to present it to your professor at the beginning of lab.
2. Using your EIU email user id and password, login to your mathcs account.
3. Obtain software to facilitate electronic submission. (One time only.)
 - Open a new Finder window (Command-n).
 - In the sidebar on its left, locate **Applications** and click on it. You should see multiple files and folders, beginning with **Address Book**, **App Store**, and so on.
 - Scroll down until you find the application **submit**, with the EIU logo.
 - Drag the **submit** app down to the dock. The name will not show on the dock, but the EIU logo remains visible.
4. Read and follow the instructions in the *Creating Java Programs and Maintaining a Web Site* handout. This 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.
5. Complete the Postlab worksheet and programming exercise.
6. As directed in the Postlab Programming Exercise, submit printouts and an electronic copy of your work when it is completed.

Postlab Programming Exercise

With few exceptions, there will be weekly programming exercises in addition to lab. In this first homework, you are to create another project based on the java program given below.

Getting Started

- To begin, login to your mathcs account and start **netbeans** (by clicking on the bluish-silver cube on the dock), if necessary. Close any open projects.
- Select **File**→**New Project...**
Within the *Categories* panel, select **Java**.
Within *Projects*, select **Java Class Library**.
Click **Next** to proceed.
- You are now able to provide a project name and assign it a location. For this exercise, we want to put the project in the **lab1** directory we created this week, and the name of the project is to be **TemperatureConvert**.
 - For the project name, use **TemperatureConvert**
 - In the project location, modify the path name if needed, to give a result similar to:
`/Users/jsbach/mat2170/lab1/TemperatureConvert`

Click **Finish** to advance.

Include acmLibrary

- Right-click on **Libraries** under **TemperatureConvert** in the left-side frame, select **Add Project**.
Choose **acmLibrary**, then click on **Add Project JAR files**.

Create a new java program file

- Right-click on **Source Packages**, then select **New**→**Empty Java File**
Click **Next**, then replace the suggested class name with **TemperatureConvert**, since the class name must match the program class name. Check that the location is correct.
Click the **Finish** button.
- Enter the code shown below into the new program window. Add header comments using your name, class section, and the date, and describing the purpose of the program.

```
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 + ".");
    }
}
```

Build and Run

- Be sure your cursor is in the program window, then click on the hammer (BUILD) on the tool bar to build `TemperatureConvert`.

If all goes well, this program will compile with no errors. (You will see *Build Successful* in the **Output** window.) Otherwise, the computer will complain about *syntax errors*. If your program has errors, carefully compare what you typed with the code shown above. Make corrections to the program, then click the tool bar hammer again. Repeat this process until the program has no more errors.

- Once the program is error-free, select **Run**→**Run File** — you can also right-click in the editor and choose **Run File**. This causes the Java applet viewer to start, and the results of your program appear in that window. After you enter a temperature and have seen the results, close the applet viewer window (click on the red dot in the top, left-hand corner). 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.

Submissions

- From `netbeans`, print a copy of this program to hand in at the beginning of Lab 2. Select **File** → **print**. The printer is located in the hallway outside our lab. You will need your Panther card and about 7 cents for each page printed. Staple all printouts to the `Postlab` sheet, and hand all of it in at the beginning of Lab 2.

- Also in `netbeans`, create the file `TemperatureConvert.html` in your `www` directory, copy the contents of `HelloProgram.html` into it, then replace all occurrences of `HelloProgram` with `TemperatureConvert`. Save this file.

- Using the **Favorites** panel in `netbeans`, create a copy of `TemperatureConvert.jar` and paste it into your `www` directory.

- Update your `index.html` file (in `netbeans`) with a new link to the `TemperatureConvert.html` file. Add the following line just after the other links:

```
<a href="TemperatureConvert.html" target="_blank">Temperature Conversion</a> <br>
```

- Save `index.html`, then open it in `firefox` (right-click in the editor window and select **View**). Check that the links work. If not, carefully inspect the files and attempt to debug the problem.

- Next week, we hope to have an app to make uploading your web page to your personal EIU web site very easy. For now, we'll skip this step.

- After you have completed both the **Postlab** (`RosePoem`) and **Postlab Programming Exercise** (`TemperatureConvert`), submit an electronic copy of your work. To do this, in a Finder window, locate and drag your `lab1` folder to the `submit` icon you placed on the dock earlier, then choose `mat2170`.

- Lastly, for this lab, you are to also submit your `www` folder as well. In a Finder window, locate and drag your `www` folder to the `submit` icon, and choose `mat2170`.

1. Create the `RosePoem` project (from your lab preview) using `netbeans`, placing it in your `lab1` directory with the other week 1 projects. Recall that the class and file names must match, and that all programs in this course must begin with header comments. After you have it running correctly, modify it as indicated below and answer the following questions:
 - (a) Add delimiter lines of `*`'s on the first and last lines of the header comments:

```
/****** and *****/
```

The exact number of stars isn't important. What happens if you put a blank line before each comment in the header?
 - (b) Remove one of the slashes from one of the comment lines and build the program. What error do you get? (Put the slash back before proceeding.)
 - (c) Try putting a comment within a comment, so that a `/* — */` pair appears within the header comments. Does this cause problems? (Take out the extra comment.)
 - (d) Take out a semicolon (`;`) from the end of the first `import` statement. Try to build the program. What happens? (Put the semicolon back.)
 - (e) Try taking out a semicolon from a different statement and rebuilding the program. What happens? (Put the semicolon back.)
 - (f) Change the line `public class RosePoem extends GraphicsProgram` to:

```
public class Poetry extends GraphicsProgram
```

and rebuild. What happens? (Put the name back to `RosePoem` and rebuild.)
 - (g) Create `RosePoem.html` and update `index.html` as you did in lab for your first two programs. 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 about 7 cents for each page printed. Staple all printouts to the `Postlab` sheet, and hand all of it in at the beginning of Lab 2.

Exercises continue on back.

2. Number the following actions in the order in which they can be used to create a new Java project in **netbeans**. There is more than one correct answer.

- _____ Add **acmLibrary** to the available libraries
- _____ Run the program
- _____ Provide a project name and location
- _____ Select **File** → **New Project** from the **netbeans** menu
- _____ Build the program
- _____ Start **netbeans**
- _____ Select **New** → **Empty Java File** after right-clicking the Source Packages.
- _____ In the *Categories* panel, select **Java**, in *Projects*, select **Java Class Library**

3. Number the following actions in the order in which they should be used to place a new applet on your web page. There is more than one correct answer.

- _____ Copy the applet (**jar** file in the **dist** folder) to your **www** folder
- _____ In the editor, type in the html code to create a web page for the applet
- _____ In the **Favorites** panel in **netbeans**, locate **www** and create a new, empty file
- _____ Edit the **index.html** file to add a link to the applet's **html** file
- _____ Check that the **index.html** file works correctly (in *mathcs lab*)