

IST 256
Lab Week 11, Wednesday, November 10, 2010

1. Practice Using Arrays and Classes.

Suppose that we have a class that represents people with two fields for name and age.

```
public class Person
{
    // fields for the name and age
    private String personname;
    private int age;

    // constructor initializes all two fields
    public Person ( String startname, int startage)
    {
        personname = startname;
        age = startage;
    }

    // accessor functions for the name and age
    public String getName()
    {
        return name;
    }
    public int getAge()
    {
        return age;
    }
    ....
}
```

Suppose that we have a main method with an array declared of type Person, and that data is read into all 50 of the Persons in the array.

```
Person [ ] personarray = new Person [ 50 ];
```

a. Write the first line of an If statement that will test if the name of the first Person in the array is “Alan”.

b. Write code that will compute the average of the ages of all 50 Persons in the array.

2. Writing a program that uses an array of classes to store file data

In this example using classes, we will define a class to represent information about Students and write a Java GUI application that can read student data from a file. Each student's information will be stored in an instance of a class called Student. This example will be used for the next two weeks as we add functionality to the form.

a. Start by creating a Java GUI application and name it something like Students. Then for that project,

- create a new JFrameForm to get the GUI window and
- put it into **package** students, and
- set the GUI to be the main class of the project.

b. Create a Student class: In the left pane of NetBeans, find the Students project and right click on the top line that says Students. In the menu, select New -> Java class. In the new class window:

- give the **class name** as Student
- select the **package** of the class to be students

c. Copy the data file

Make a Resources package, by again selecting the project Students and creating a new package that goes under the src folder. From the web page, copy the file students.txt to the Resources folder. (Or create the file in NetBeans as a new – Other – empty file and then copy and paste some student data into the file.)

d. Write the Student class.

Use the following code for the Student class (be sure and get the code from the top of the next page):

```
// fields for Student
// name, gender (either M or F), age in years, height in inches
private String studentname;
private String gender;
private int age;
private int height;

// Constructor gives an initial value to all fields
public Student(String startname, String startgender,
               int startage, int startheight)
{
    studentname = startname;
    gender = startgender;
    age = startage;
    height = startheight;
}
```

```

// no accessor methods so far

// display all field values in a String
public String toString()
{
    String result = "Student: " + studentname + ", gender: " + gender
        + ", age: " + age + ", height(inches): " + height;
    return result;
}

```

As usual, correct any issues with spacing and curly brackets.

e. Create the form interface.

On the form, we will have a button to read the file and save the data. Then add a button and label to display all the students. The form can look something like this:

__ Read Student File__	File Status	(button and label)
__Display Students__		(button)
Results:		(label)

You may find it helpful to change the variable names of the buttons and labels. For each button, select Event -> action -> actionPerformed.

f. Write the code for the File Read button

Most of the code for this button can be adapted from the code in the FileCandyClassGUI program. So it will make sense to keep that code open in your browser and to copy and paste parts of it and modify it for the Students example.

1. Declare variables at the class (global) level for an array of Students, allowing up to 50, and for the number of students read from the file. Call the array students and the integer numstudents.
2. Declare variables for a BufferedReader and Scanner. Do not create a FileChooser in order to save time in lab.

```

BufferedReader in = null;
Scanner sc = null;

```

And add the import statements at the top of the class, right after the package statement.

```

import java.io.*;
import java.util.Scanner;

```

3. In the button code, add the variable declaration for a count and add variable declarations to read each item from the file: studentname, gender, age and height.

4. Add a try/catch block and add the code to create a `BufferedReader` and `Scanner` for a CSV file:

```
// open the file and set up the scanner
in = new BufferedReader(new FileReader("src/Resources/students.txt"));
sc = new Scanner(in);
// use comma as a delimiter
sc.useDelimiter(",!(\\n\\r)+");
```

5. Copy the while loop that reads the file from the `FileCandyClassGUI` code and modify it:

- Change the variable names of items read from the file
- Add another if statement to read 4 items from each line, instead of 3
- Create a `Student` object instead of a `CandySales` object
- Change the array and `numstudent` variables names to save each `Student`.

6. After the while loop, add the code to save the count in `numstudents` and to close the scanner.

g. Write the code for the Display button.

In the code for the second button, copy the code from the second button of the `FileCandyClassGUI` and modify it for the `Students` program.

Change all the variable names: `CandySales` becomes `Student`, `oneSale` becomes `oneStudent`, and the `sales` array becomes the `students` array.

Change the title string in the message to say "`<html>Students: <p>`" and change the name of the label where you are displaying the results.

h. Test your program.