Weeks 9-10 (Chapter 7) Lab 1

Do a couple of array initialization exercises plus Chapter 7 Exercise **2** (write and test a ***CountFamilies*** class), and, for extra credit, an exercise where *main* uses the *args* array.

A. Create a program called **Initializers.java** to practice creating and initializing or setting the values in an *int* array, and accessing elements in that array in *main*:
   1. Declare and instantiate an *int* array named ***array1*** having size **10** using the Java *array initializer* syntax with **{ }** to fill *array1* with the even integers 2 through 20. ***Hint*: use just one statement:**

      int[] array1 = { … };
   2. Using a *for* loop, print out the elements in *array1* all across one line, separated by single spaces, then print a blank line.
   3. Now ask the user how long to make the array using the *Scanner* or *Keyboard nextInt()* method, and assign a new *int* array of that length to reference variable *array1*, then fill it with even integers starting with 2, using a *for* loop; finally, print *array1*'s values, using a *for* loop as above.

      ***Hints*: you will have to compute each value that you put into the array in the loop, and you will need to use *array1.length* or the size the user types in as the limit in the two *for* loop conditions.**

Show me your program source code and how it works, or submit on Blackboard.

B. Do Chapter 7 Exercise **2**, create and test a ***CountFamilies*** class:

- Write a program in a class *CountFamiles* that counts the number of families whose income is below a certain value.
- Read an integer *k* from the keyboard and then create a *double* array of size *k* called *incomes*.
- Read *k* values representing family incomes from the keyboard using *Scanner* or *Keyboard nextDouble()* and place them into the array.
  ***Hint*: use a *for* loop to fill the array with these *double* values.**
- Find the maximum *double* income among these values.
  ***Hint*: you can determine the maximum as you're filling the array.**
- Then count the families that make less than 10 percent of this maximum income. Display this count and the incomes of these families.
  ***Hints*: once you know the maximum, you can calculate 10% of it to use in a comparison; counting and displaying incomes of families that make less than that amount requires two additional *for* loops.**

Show me your program source code and how it works or submit on Blackboard.

C. For practice in writing and using methods that take an array parameter, copy **Initializers.java** to **Initializers2.java** and modify it as follows – first write two sets of new, overloaded *display()* and *fill()* methods ahead of *main* that take array parameters:

1. Define two overloaded versions of a *static void display()* method:
   - **static void display(String header, int[] a)** – if the *header* is not the empty *String*, "", print that String followed by a space, using *System.out.print()*; in any case, print all the elements of array *a*, separated by single spaces, across one line, then print a blank line. ***Hint*: copy the second version of the printing *for* loop from Initializers.**
   - **static void display(int[] a)** – call **display("", a);** // no header
2. Define three overloaded versions of a *static void fill()* method:
   - **static void fill(int[] a, int start, int diff)** – fill up array *a* with values starting with *start* and adding *diff* for each element **(use a *for* loop)**
   - **static void fill(int[] a, int start)** – **call fill(a, start, 2);** // diff 2
   - **static void fill(int[] a)** – call **fill(a, 2, 2);** // start at 2, diff is 2

Now modify *main* (parts carried over from **Initializers** are in blue):

3. Declare and instantiate an *int* array named ***array1*** having size **10** using the Java *array initializer* syntax with **{ }** to fill *array1* with the even integers 2 through 20, then use the **first** version of the *display()* method, defined above, to print out the values of *array1*, preceded by the header *String* **"array1:"**.
4. Now ask the user how long to make the array, and assign a new array of that length to reference variable *array1*, then fill it with even integers starting with 2; use the **third** version of the *fill()* method, defined above; finally, print *array1* using *display()* as in #3 above. **Using *fill()* and *display()* makes *main* much simpler.**

D. Extra Credit: Write and test a ***MainArgs*** class that takes input from *args* in *main*:

- If *args.length* > 2 *main* should print this usage message and exit:

  `usage: MainArgs [name [salary]]`

- Write a program in a class *MainArgs* that uses information from the *args* array if information was passed in that way, or else reads information from the user at the keyboard.
- *MainArgs* expects two values, a *name* (*String*) and a *salary* (*double*).
- If *args.length* == 0 *main* should ask the user for those two values using *Scanner*'s or *Keyboard*'s *nextLine()* and *nextDouble()* methods, and assign them to *String* variable *name* and *double* variable *salary.*
- If *args.length* == 1 *main* should assign *args[0]* to *name* and prompt the user to enter the *salary* using *Scanner*'s or *Keyboard*'s *nextDouble()* method.
- If *args.length* == 2 *main* should assign *args[0]* to *name* and *args[1]* to *salary*: note that **`Double.parseDouble(args[1])`** will convert *args[1]* to a *double.*
- If *args.length* <= 2 *main* should then print the *name* and *salary*.

Show me your program source code and how it works, or submit on Blackboard.