MIT AITI Mobile Application Development in Java

Nigeria, Summer 2012

Lab 07: Arrays, Methods, Classes and Objects

Read instructions carefully! Following instructions is part of the grade for this lab. Please do not hesitate to ask any of the team members for clarifications. Create a new project named "lab07".

Part I - Arrays and Methods

Part I of this lab will combine two concepts that we have learned so far, arrays and methods. You will create arrays to hold data entries of a contact list, and you will be creating methods to change the values of these entries. To begin, create a class called "ContactList" in your java project.

1. Create a main method and initialize three empty arrays of length 20, one to hold contact names, one to hold the contact phone numbers, and one to hold the contact email addresses (all elements being of type `String`s). An entry of the contact list is held in these three arrays. The i^{th} element of each array corresponds to the same contact. All of the information for one contact is held at the *same index* in the three arrays.

2. Also in the main method, prompt the user to input a series of contacts. For each contact the user will enter the contact's name, followed by the contact's phone number, and finally the contact's email address. Have the user input each of piece of information on a separate line. The contact list does not have to be full, so allow the user to terminate the prompting of new contacts by entering "--" for the contact name.

3. Write a method that prints the values of arrays by contact. Print out all the information for one contact on each line. For example, if your contact name array was {Mike, Zach, Michelle}, the corresponding phone number array was {0710392847, 0710482957, 0710376853}, and the corresponding email array was {mike.g@gmail.com, zach393@hotmail.com, mandm@aol.com}, then your method would print.

   ```
   Contact List:
   Mike, 0710392847, mike.g@gmail.com
   Zach, 0710482957, zach393@hotmail.com
   Michelle, 0710376853, mandm@aol.com
   ```

4. Create a method that adds a contact to the contact list and returns true if the contact list was not full and the contact was added, and returns false if the contact list was full and no new

contact was added. Then call this method in the main method to add a contact with the following information: Judy, 0710111321, judy.wabufo@gmail.com. Then print out the updated contact list.

(Hint: Find the index where to add the new contact information to the arrays by searching for the first element that equals (==) `null` in one array. `null` means that the element has no value set to it.)

5. Often people change their phone number or email address. Create a method that changes the value of the email address array or the phone number array to a new, updated value. You should create only one method, but have it be able to update either the phone number array or the email address array, depending on what arguments are passed. Then call this method in the main method to update a contact's information. Judy has changed her phone number to 0710539586. Now print out the updated list. The signature:

```
void changeValue(String name, String[] namesArray,
                 String newValue, String[] values)
```

(Hint: To do this, pass the name and the array of names as the first two arguments to the method. Use these arguments to find the index of the element to be replaced. Then use the values array to replace the element with the new value.)

6. Now run your program and add the following contacts into your contact list:

Mike, 0710392847, mike.g@gmail.com
Zach, 0710482957, zach393@hotmail.com
Michelle, 0710376853, mandm@aol.com
Cory, 07104343489, cory.smith@mit.edu
Julian, 0710492857, juice.man@gmail.com

When you run your program, it should print the following (without comments):

```
Contact List:
Mike, 0710392847, mike.g@gmail.com
Zach, 0710482957, zach393@hotmail.com
Michelle, 0710376853, mandm@aol.com
Cory, 07104343489, cory.smith@mit.edu
Julian, 0710492857, juice.man@gmail.com
Contact list:
Mike, 0710392847, mike.g@gmail.com
Zach, 0710482957, zach393@hotmail.com
Michelle, 0710376853, mandm@aol.com
Cory, 07104343489, cory.smith@mit.edu
Julian, 0710492857, juice.man@gmail.com
Judy, 0710111321, judy.wabufo@gmail.com     <=== Added contact
Contact list:
Mike, 0710392847, mike.g@gmail.com
Zach, 0710482957, zach393@hotmail.com
```

```
Michelle, 0710376853, mandm@aol.com
Cory, 07104343489, cory.smith@mit.edu
Julian, 0710492857, juice.man@gmail.com
Judy, 0710539586, judy.wabufo@gmail.com
```
<=== Changed Judy's phone number

Part II – Classes and Objects

In Part II, we are going to implement the contact list from Part I using a different technique. For Part II, create a class named `Contact` that encapsulates the information for each "contact". We will then create an array of `Contact`s in the `ContactList` class from Part I. This will allow us to use only one array: `Contact[]` (an array of `Contact`s) instead of the three arrays used in Part I. All data for a contact will be encapsulated inside of `Contact`.

1. Create a new class in your project called "Contact." Note that this class will NOT have a main method.

   `Contact` should have fields that store a name, e-mail address, and phone number (all `String`s). Its constructor should take arguments that initialize these fields:

   ```
   Contact(String name, String emailAddress, String phoneNum)
   ```

2. Implement the methods `getName()`, `getEmail()`, and `getNumber()` in `Contact`. These should return the value of the fields that store the name, e-mail address, and phone number respectively.

3. Implement the methods in `Contact`:

   - `changeEmail(String newEmail)`
   - `changeNumber(String newNumber)`

   These should change the value of the fields that store the e-mail address and phone number respectively.

4. Implement a method "`public String toString()`" that returns a `String` representation of a contact:

   name, phoneNumber, emailAddress

   (Remember `String` concatenation is achieved with "+". Do not add a new line at the end.)

5. Create a new method in your class from Part I (`ContactList`) to add a new `Contact` to your array of `Contact[]`. Use a similar technique as in Part I: search for the first empty element in the Contact[] array, then add your new contact to this index. Do not store your array in the `Contact` class. The `Contact` class models a single contact.

6. Create a new method in `ContactList` to re-create the implementation from Part I steps 2-5, but this time using the `Contact[]` array. Name this method `partII()` and call it from the `main` method in `ContactList`.

   Your new method (`partII`) should perform the following steps:

   - Ask the user to input a sequence of contacts (as in Part I step 2), then print the list.
   - Add Judy to the contact list, then print the list.
   - Change Judy's phone number, then print the list.

   Create helper methods as necessary (following the example in Part I). Reuse as much code as possible from Part I. Your method, `partII()`, should print the same results as Part I.

7. Think about the difference between the techniques used in Part I and Part II. Which technique is better and why? Explain your answer to a Teaching Assistant.