PAL Worksheet Arrays Advanced CSC20

Topics: Arrays of Objects - Parallel Arrays - Two dimensional Arrays

Ref: CSC 15 PAL WS Array Basics, Call by Reference, Public Fields, For Loops

Suppose you have two arrays: One of them called *friendsNames,* holds the names of ten of your friends. The other called *phoneNumbers*, holds their phone numbers. Given a name, to be able to *lookup* a phone number, it would be a good idea to make sure that these two arrays have information that corresponds at each index much like columns of an excel spreadsheet. So, if "Diego" is at index 0 of the array *friendsNames,* it would make sense to store his phone number at *phoneNumbers[0].*

Ex.1. Write a small program that reads in names and phone numbers from a text file and stores them into parallel arrays of size 10. Your file may contain either less or more than 10 pieces of data. Each pair of data is on a new line in the file and the name and phone number are separated by a space e.g. Diego 919-777-3455. *Hint: use a counter to count the actual number of data pairs that were read into the arrays. What would work better a for loop or a condition based while loop?*

Ex.2. Since an array index can be an expression, lets use a random number to give you a random friends name and phone number. Update Ex.1. with a method that does that.

Ex.3. Suppose we would like to store friend's birthdays along with their names and phone numbers. Adding another array seems cumbersome. Create a *ContactInfo* class that contains name, phone number and birthday (all string type). Create a constructor for this class that takes in name, phone number and birthday as parameter. Modify your programs in Ex1 and 2 to create an array of *ContactInfo* objects. Hint:

1. Declare ContactInfo [] friendInfo = new ContactInfo[10];
2. Read in the data for one friend from a file
3. Create an object that stores a friends information using the constructor
4. ContactInfo friend = new ContactInfo(name, phoneNumber,birthday);
5. Assign this friend to an element of the array starting with element 0. Repeat as required.

Display the entire friend's directory. What do you think would happen if you tried to print the contents of the *ContactInfo* array before doing step 1 above? Try and see what happens.

Sometimes we need to work with a table of information again like an excel spreadsheet with many rows and many columns. Java allows us to create multidimensional arrays. The number of rows in a table need not be the same as the number of colums (so we can have a square or rectangular table. We will not work with "jagged arrays" where the number of columns can be different for each row). Suppose we have a 2x 3 matrix called *simple* that looks like this

[1 2 3]

[4 5 6]

If *r* stands for number of rows and *c* for number of columns, then r==2 and c==3. In Ex.3 we were able to use a single for loop to traverse the single dimension array. With 2 dimensions, we need two loops where the outer loop traverses the rows and the inner loops traverses the columns for a particular row. Read the PAL worksheet on For loops if you need to review nested loops.

Ex 4. Write a method that takes in a two-dimensional array (like *simple*) and returns the sum of all the numbers in that array.

Hint: Often we need to determine the values of *r* and *c* to be able to traverse them. Try *simple.length* to get the number of rows and *simple[0].length* to get the number of columns. Explain how this works.

Copying complete or partial arrays, sorting and displaying can be tedious since it involves Iterating over each element. The Arrays utility offers methods to create, compare, sort, search, display, and transform one dimensional arrays. E.g. for *Arrays.sort(friendsNames).* You can read more about this in Chapter 7.

Ex5. Practice using *Arrays.toString*, *Arrays.copyOf*, *Arrays.sort* etc on a one dimensional array

Ex6. Copy *simple* into an array called *anotherSimple using Arrays.copyOf*. Display the arrays using *Arrays.toString*. Did it work? Why or why not? Look up how to *"deepCopy"* a two dimensional array. Why would regular copy not work?