ArrayList Practice Lab

In this lab, you will practice using the ArrayList class. ArrayLists can be used to store and manipulate objects just like arrays. Like arrays, you can get the size of the list (in arrays, this is arr.length, in array lists, you use al.size()) and access individual elements of the list (in arrays, you use arr[i] to get the value of one element, in array lists, you use al.get(i)). Unlike arrays, array lists can be resized. That is, new elements can be added to the array list at any time, and old items can be removed.

Exercise 1 – Infinite Family Info 1

Just like the last iteration of our family info table program, this program will read in names, ages and favorite colors of various people. However, this time, rather than prompting the user to tell us how many people there are at the beginning of the program, after adding each new person, our new program will prompt the user to see if they are done adding new people. Additionally, rather than using parallel arrays to store information (that is, we had one array to store names, one for ages, one for favorite colors), we’ll use an object to store the information for each person and then have one list of Person objects.

1a – Class Person

The Person class needs to be written. A Person has a name, an age and a favorite color. Use Strings to represent the name and color and an int to represent the age. Provide a constructor for the Person object which consumes one value for each field. Then, provide getters for each field. Finally, add a method called haveBirthday which increments the age of the Person.

1b – Read in Persons and their information.

In the function infiniteFamilyInfo, declare and create an array list of Person objects. Then, in the while loop, prompt the user to give you the next person’s name, age and favorite color. Then, add a new person to your list. Then, prompt the user to type Y or N if they are done or not done. Read in their response and set the variable done to true if they user is done or false if they aren’t.

1c – printInfoTable

This function consumes a list of Person objects and prints out the table of their information.

At this point, test your program. Your output should look like this:

Enter person 1's name:

adam

Enter adam's age:

19

Enter adam's favorite color:

green

Are you done? Y/N

N

Enter person 2's name:

betty

Enter betty's age:

13

Enter betty's favorite color:

red

Are you done? Y/N

n

. . .

Enter person 8's name:

heidi

Enter heidi's age:

78

Enter heidi's favorite color:

white

Are you done? Y/N

y

Name Age Favorite Color

==========================================================

adam 19 green

betty 13 red

carol 45 pink

dave 38 blue

edgar 34 purple

frank 40 chartreuse

gretchen 28 yellow

heidi 78 white

Exercise 2 – Infinite Family Info 2 –

Once you have finished exercise 1, you can switch over to running the line

infiniteFamilyInfo( "FamilyInfoInputData.txt" );

This will read information from a file instead of from the keyboard. This will save you time typing in all the information for a bunch of people each time you want to run. Run this now and make sure that you get the table below:

Name Age Favorite Color

==========================================================

Adam 32 Green

Betty 45 red

Carol 23 green

Dave 38 GREEN

Edgar 30 Pink

Fred 32 pink

Greg 22 green

Helen 34 purple

Ingrid 30 gren

Julie 29 blue

In this exercise, you will implement several functions which consume Person lists and produce various bits of information or perform various tasks.

2a - getAverageAge

getAverageAge consumes a list of Person objects and returns the average of the ages of the people in the list. You should get 31.5 for the data file above.

2b – getNames

getNames consumes a list of Person objects and returns a new list which contains the names (Strings) of the Persons in the input list. For the data file above, you should get the list

[Adam, Betty, Carol, Dave, Edgar, Fred, Greg, Helen, Ingrid, Julie]

2c – getFavoriteColors

Just like getNames, but gets a list of the favorite colors of the people in the list. For the data file above, you should get the list

[Green, red, green, GREEN, Pink, pink, green, purple, gren, blue]

2d – haveBirthdays

haveBirthdays consumes a list of Person objects and gives every person a birthday. After the birthdays, you should get the table:

Happy Birthday, everybody!!

Name Age Favorite Color

==========================================================

Adam 33 Green

Betty 46 red

Carol 24 green

Dave 39 GREEN

Edgar 31 Pink

Fred 33 pink

Greg 23 green

Helen 35 purple

Ingrid 31 gren

Julie 30 blue

2e – filterFavoriteColorGreen

filterFavoriteColorGreen consumes a list of Person objects and produces a new list of Person objects which contains all the people whose favorite colors are green (ignore case). Note that the input list should NOT BE MODIFIED in this process. Printing the table for this new list of people should produce:

Here are the people whose favorite color is green:

Name Age Favorite Color

==========================================================

Adam 33 Green

Carol 24 green

Dave 39 GREEN

Greg 23 green

Here are all the people again:

Name Age Favorite Color

==========================================================

Adam 33 Green

Betty 46 red

Carol 24 green

Dave 39 GREEN

Edgar 31 Pink

Fred 33 pink

Greg 23 green

Helen 35 purple

Ingrid 31 gren

Julie 30 blue

2f – killOverThirties

killOverThirties consumes a list of Person objects and removes from that list all the people over thirty. Note that this WILL change the input list. After calling this function, you should get this table:

Let's kill everybody who is over thirty!

Name Age Favorite Color

==========================================================

Carol 24 green

Greg 23 green

Julie 30 blue

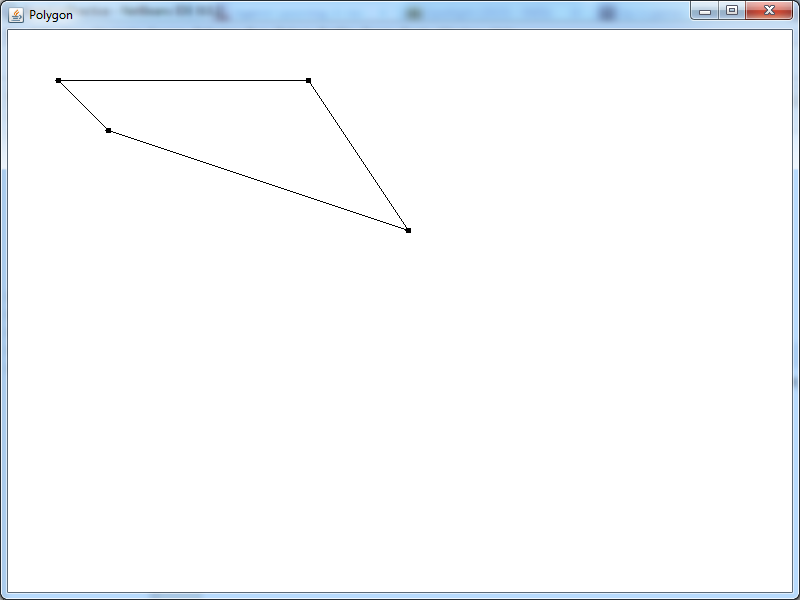
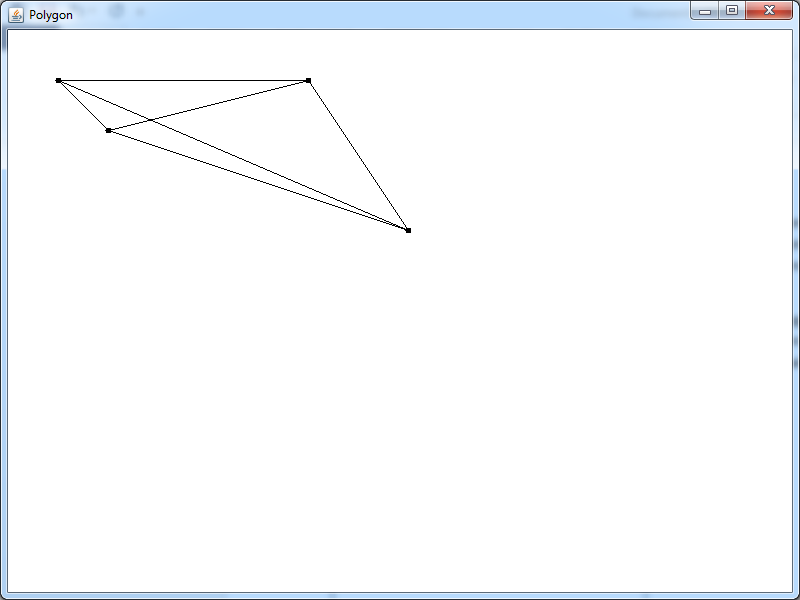
Average age is 25.666666666666668

2g – Compare and contrast killOverThirties and filterFavoriteColorGreen. Think about the purpose of these two functions (ignore the implementation details). How are they similar? How are they different? Could you write a function like killOverThirties to do what filterFavoriteColorGreen does? Could you write a function like ffcg to do what kot does? What are some advantages to each approach? For example, if someone had written the function killOverThirties, and you just wanted to get a list of the people under 30. Could you use their function to accomplish your task? What about if someone had written the function filterFavoriteColorGreen and you wanted to kill everybody whose favorite color is not green. Could you use the existing function to accomplish your task?

Exercise 3 – PolygonView

This view object will allow you to create and draw a polygon. A polygon is a list of Point objects which represent the vertices of the polygon. The Point class is provided for you. It has two fields (which are public) that keep track of the x and y coordinates of the point. The constructor consumes two values – one for x and one for y.

The paint method of PolygonView should paint each vertex of the polygon and the lines which connect adjacent vertices. If the drawDiagonals field is true, then the paint function should, additionally, paint each of the diagonals of the polygon (that is, one line between each pair of vertices). Test by running launchPolygonViewer1. Press ‘d’ to reveal the diagonals:

The handleMouseClick function should add a new vertex to the polygon. Use the mouse coordinates in the MouseEvent to figure out where the point should be, then add a new Point to the list.

When you are done with this task, test by running function launchPolygonViewer. You will initially see nothing, but by clicking around the window, you can create a polygon with ever increasing numbers of vertices (without and with diagonals):

