Programming Projects  Data Structures 
2, 3 AND 4  
CSI310  
Spring 2012

Due On Blackboard: [2 and 3 Saturday, Feb. 19 at 11:59PM.] [4 one week later: Saturday Feb. 26] Upload your .java files as attachments.

You will need to have installed a Java programming system like Dr. Java (and the JDK except on a MAC) and have downloaded and added the “java-source” directory to your CLASSPATH. See the textbook and recent Blackboard announcements for assistance with setting this up on your own computer. Your Lab01 work already put java-source in the “S: drive” visible on UA Library computers: On them, you start DrJava from “Java Development Tools” (in the Start menu) but you might have to set your CLASSPATH again.

Lab 02, and the basic Picture creation and composing technologies from Project 1 are the starting points. The main linked list idea was covered in Lct 05. More resources about (A) linked lists (and “social” networks) and (B) arrays will be announced. Create three complete Java applications each with the following structure and behaviors:

1. It must use Guzdial and Ericson’s Picture and FileChooser in their “Java Source” archive, so your code can use their compose() method. No more flipping! Also soon after execution begins, your programs must execute: (Of course, you must find out and observe what that does! We need that to help test your work.)

FileChooser.setMediaPath(FileChooser.getMediaDirectory());

2. Structure and behavior different for the three projects is described below.

• **Project 2** must use one array for holding Picture references.
  (a) It must ask the user how many images to import. It must then allocate an array of Pictures to refer to all of them. It must work for 1 image, 2 images, 100 images, indeed any number! If the user says zero, it must “behave reasonably” which could be print an explanation and exit.
  To ask for and get an int for the array length, and then allocate (that is, construct) an array having that length, use code like this:
  
  ```java
  import java.util.Scanner; //Put this above the class definition 
  class ... { }
  //...
  //within executable code, in suitable places:
  Scanner uiInput = new Scanner(System.in);
  //....
  System.out.println("How many images?");
  //....
  int nImages;
  nImages = uiInput.nextInt();
  //....
  Picture myArray[] = new Picture[nImages];
  //Java implementations must make each entry myArray[0], myArray[1], etc.
  //start out containing the null reference value for you.
  ```

  (b) Your program must, for as many images as the user asked for, bring up the FileChooser window that many times. Each time, it must make a Picture from the image and must store a reference to that Picture in the array.

• **Projects 3 and 4** must use one linked list for holding Picture references.
  (a) It must ask the user whether or not he or she wants to supply another (or the first) image. The user must type Y to supply each image.
  Your code should use a Scanner object as above. Code the user input and comparison against "Y" by:

  ```java
  String maybeY = uiInput.next(); if(maybeY.equals("Y")) { ... }
  ```
NOT maybeY == "Y".

If not, it stops inputting images and goes on to display the collage. It must behave reasonably if the user chooses not to import the first image.

(b) For each image, it must (1) bring up the FileChooser window, (2) it must make a Picture from the image and (3) must store a reference to that Picture in the linked list data structure at the end.

3. When the user finishes importing images, it must make and show a new Picture whose image is the result putting all the imported images in a row, each centered from top to bottom. They must be ordered left-to-right in the order they were inputted. For this project, a border is not required.

4. While the original collage is on display, it must print this text-based menu:

   CM - cut middle and move it to the clipboard
   PE - paste clipboard to end
   CE - cut end and move it to clipboard
   XX - stop running this program

   (Tip: Copy and paste the menu text from the .pdf directly into your Java code!)

Projects 2 and 4 require all four commands be implemented. Only the XX command is required in Project 3; add the other three within one week after Feb. 19. When a non-implemented command is given, the program should print a message like PE not implemented, specific each of the 3 non-implemented commands!

Here’s how to do the menu and command decision:

    import java.util.Scanner; //Put this above the class definition class ... {  
    //...
    //within executable code, in suitable places:
    System.out.println("CM - cut middle and move it to the clipboard");
    //...
    Scanner uiInput = new Scanner(System.in);
    //...
    if(uiInput.equals("CM") ) //***DON'T use uiInput=="CM" !
    {
        //code to run when CM is commanded...
    }

5. The clipboard shall be either empty or hold exactly one image, so “moving to the clipboard” discards the previous clipboard contents.

   People expect clipboards to retain their contents after pasting. So, make sure that if PE is commanded 2 or more times in a row, that number of duplicates of an image will appear at the end!

6. For Project 2 (the array version) only When more pastes than cuts have been commanded, the array can “run out of space” to hold more Pictures than the user specified when he or she was asked “How many images?” Your Project 2 MUST detect that, give an informative message, and continue to run. The user can then do some cuts and after that, paste successfully.

7. For Projects 2 and 4, skip for 3. When the number of displayed images (counting repetitions!) is odd, the middle one is well-defined. When that number is even, the one cut must be the one just left (or before) the place between the two subsequences with equal numbers of items. (Example: From sequence ABCD, we want the result of cutting the middle to be ACD, not ABD.)

   Notice: If a Picture is too big for your computer screen, the show method will cut it off. That is OK. Just use smaller or fewer imported images to test your program. Or try to save the results in files and view them with more functional viewer software.