**Lab 8 Pt2** (25[20+5] points)

There are several learning objectives to this assignment

* Creating Classes that use Inheritance
* Creating Interfaces and Abstract Classes
* Creating Classes that implement Interfaces and Abstract Classes
* Creating advanced algorithms

In this lab, you will draw arrow shapes with a tail and head that are potentially drawn several lines from the top line and offset from the left edge. This simulates a low end display that does not have any graphics capabilities and only allows text output. In order to be efficient and consistent from a CSCI 110 course perspective, this lab will incorporate interfaces, inheritance, and abstract classes.

Basic Interface

1. Create an interface called BasicInterface based on the following UML. Ensure that each method has an appropriate description (javadoc)

|  |
| --- |
| BasicInterface |
| + setXAdj (int newXAdj): void  + getXAdj (): int  + setYAdj (int newYAdj): void  + getYAdj (): int  + drawShapeOn (): void  + drawShape (): void |

ArrowInterface

1. Create an interface called ArrowInterface that extends BasicInterface and is based on the following UML. Ensure that each method has an appropriate description (javadoc)

|  |
| --- |
| ArrowInterface |
| + setArrowDim(int newTail, int newWidth): void |

BasicShape

1. Create an abstract class BasicShape that implements BasicInterface with two private instance integer variables called xAdj and yAdj. Note: After dropping down to the appropriate line based on yAdj, drawShapeOn () invokes drawShape() based on xAdj. drawShape is an abstract method. Ensure that each method has an appropriate description (javadocs)

RightArrow

1. Create a RightArrow class that extends BasicShape and implements ArrowInterface. RightArrow has two private instance integer variables tail and width. While not mandated, you should consider private methods such as drawArrow() that uses drawFiller() to draw the correct spaces in between the \*’s, drawTail(), and drawSpaces() that creates spaces per the xAdj. yAdj is accounted for in drawShapeOn().

LeftArrow

1. Create a LeftArrow class that extends BasicShape and implements ArrowInterface. LeftArrow has two private instance integer variables tail and width. While not mandated, you should consider private methods such as drawArrow() that uses drawFiller() to draw the correct spaces in between the \*’s, drawTail(), and drawSpaces() that creates spaces per the xAdj. yAdj is accounted for in drawShapeOn().

Driver

1. The driver is based on the following convention (xAdj, yAdj, tail, width). NOTE: RightArrow (and LeftArrow) class(es) should have a default constructor that passes: xAdj=0, yAdj=0, tail=5, width=5 and a second constructor that passes xAdj, yAdj, tail, and width per driver constructor calls.
   1. Create a default RightArrow object, right1
   2. Draw right1 object
   3. Adjust object in (a) to xAdj of 10
   4. Draw right1 object
   5. Adjust object in (a) to yAdj of 5
   6. Draw right1 object
   7. Simultaneously adjust object in (a) tail to 10 and width to 15
   8. Draw right1 object
   9. Create a RightArrow object with xAdj =5, yAdj =10, tail=15 and width=15, right2
   10. Draw right2 object
   11. Create a LeftArrow object with xAdj =13, yAdj =0, tail=15 and width=15, left1
   12. Draw left1 object
   13. Simultaneously adjust object in (k) tail to 10 and width to 11
   14. Draw left1 object
   15. Adjust object in (k) to xAdj of 5
   16. Draw left1 object
   17. Adjust object in (k) to yAdj of 5
   18. Draw left1 object
   19. Create a default LeftArrow object, left2
   20. Draw left2 object

**The driver is based on the following convention (xAdj, yAdj, tail, width)**

**The default constructor should be xAdj=0, yAdj=0, tail=5, width=5.**

You will need to do the following:

1. Create BasicInterface
2. Create ArrowInterface
3. Create BasicShape class
4. Create RightArrow class
5. Create appropriate constructors, getters and setters
6. Create Lab8Driver that incorporates the above reqts in the driver section

**Submitting your work**

For all labs you will need to provide a copy of all .java files. **DO NOT PROVIDE .class files. I cannot grade, what I cannot read.** In addition to your .java files, you will need to produce a pix of the screen output in .png or .jpg format for each project that covers all use cases in the lab. For persons using Windows 7 and above OS, you can use the built in snipping tool. Mac OS users, you can see how to take screenshots using the following url - <http://www.wikihow.com/Take-a-Screenshot-in-Mac-OS-X>

You will need to zip your files into a single container. **DO NOT USE .rar for Mac OS.** Submit appropriate .java files and screenshots to show input and output in either .png or .jpg format.