SUMMARY

In this project you will create a new class for picture albums, using the Picture class. It will also require that you demonstrate applying to a bigger project the practice of keeping a VERSION HISTORY which was taught and required in the last few laboratories.

YOUR SUBMISSION MUST BE A VERSION HISTORY, with separate versions in which one of the two constructors was started and later completed first! Upload to Blackboard a .zip file containing ALL THE VERSIONS, as you did in the laboratories. Submissions that are NOT version histories will get 0 points!

DUE TIME

ON-TIME: Wednesday, May 2, 11:59PM. (-(100/7)% per day late after that.)

PREPARATION

Chapter 11 of the text, but instead of a Student with multiple constructors and instance methods you will make an Album which collects Pictures instead of grades.
DESCRIPTION
You must create one class named `Album`. So you must start editing a file named `Album.java` to contain the class definition:

```java
public class Album {

The `Album` class will have 7 fields:

1. An `int` named `width`
   Its value is the width of the completed result Album Picture when that width is known, either GIVEN or computed. You might use it to sum up the widths of the Pictures as they are added, or sum up the widths later from the array.

2. An `int` named `height`
   Its value is the height of the completed result Album Picture when the height is known, like `width`. You might use it to track the maximum height of Pictures as they are added, or use the array for this later.

3. A `Picture` named `result`
   Its value will be the reference to the `Picture` object into which the colors of the Pixels of the `Picture`s to be added will be copied.

4. An `int` named `nextX`
   Its value will start at 0 and be used to track the X-coordinate at which to begin copying the next `Picture`.

5. An `int` named `nPreGiven`
   Its value will be set by the constructor (see below) that enables us to specify exactly how many `Pictures` are going to be added to the `Album`. Its value will be 0 if the OTHER constructor was used instead.

6. An `int` named `nAdded`
   Its value will start at 0 and be used to track the number of `Pictures` that have been added so far. It will be used to tell where in the array below the next `Picture` should be put.

7. A `Picture` array reference named `pictArray`
   It will be initialized by `pictArray = new Picture[nPreGiven];` when `nPreGiven` is non-zero (see above). As `Pictures` are added one-by-one, references to them will be stored in this array so that they are available for copying LATER, at some time AFTER the correct values in `width` and `height` (see above) have been computed.

`width` and `height` (above) might also be computed using a loop that loops through the array `pictArray`, to sum the widths and find the maximum of the heights.
The Album class will have 2 constructors:

(Remember that a constructor is a special kind of method that is called automatically just after and only when the new operation is done to make a new Album.) The 2 constructors will be different because they have different numbers of parameters. (The technical term for such different constructors is "polymorphic"). They and their differences are explained below:

```
public Album( int theWidth, int theHeight )
```

This constructor enables the program using Album to set up an Album that will make a Picture with a pre-determined width and height. Pictures can then be added to the Album as long as there is room.

```
public Album( int theNumberOfPictures )
```

This constructor enables the program using Album to specify a pre-determined number of Pictures that will be added to the Album. Exactly that many Pictures must then be added and stored in the array. The Album code will calculate the width and height of the new Picture that the Album code will make, AFTER all that number of Pictures have been added.

The Album class will have 2 public methods:

```
public boolean addPicture(Picture thePicture)
```

This method tries to add the given Picture to the Album. It returns true if the Picture is successfully added and false otherwise.

```
public Picture getResult( )
```

This method returns the result Picture that is formed by copying the added Pictures into a new, white Picture. All the added Picture copies should start at the top of the new Picture and be laid out horizontally, with no overlaps or gaps. Return null if there's an error.

Tips: (1) The result Picture can be made immediately when the Album( int theWidth, int theHeight ) polymorphic form of constructor was used. But, when the other form was used, the result Picture CANNOT be made until AFTER ALL nPreGivenPictures have been added, because the width and height of the result cannot be computed until ALL the widths and heights of the added Pictures are available for the computer to use. So the array IS NEEDED! (2) Make a private method copy(Picture p, int xWhere, int yWhere) to copy into the result Picture!! (will be on final!)