Learning Objectives:

1. Understand and use for image analysis algorithms and programs some basics:
   - Recognize that digital images are rectangular arrays of pixels.
   - Each pixel has a red (R), a green (G) and a blue (B) intensity each represented by an integer.
   - Each pixel has x and y position coordinates.
   - The gray scale value for a pixel is the average of its R, G and B intensities.
   - The three color separation images and the calculated gray scale image. Observe the 3 separations and the gray scale image that come from one color image.
   - A histogram is an array and the method for computing the histogram values.

2. Recognize, and use and program in new situations:
   - A doubly-nested loop (like the one used in the ASCII plotting exercise and homework).
   - The concept of an array and Python lists used as arrays.
   - Integers calculated or inputted and then used for array subscripts.
   - Incrementing many different array elements in order to obtain counts of many things at once.

3. Learn the definition (i.e., formula) for the centroid (also known an mean in statistics) of a set of weighted points. Program calculations of centroids by coordinates. Understand the geometric concept of centroid by locating the centroid of the different colors and their gray scale value in an image. Be guided to think, hypothesize and express one's own conclusions about centroids, light and color; and their possible application to autonomous robots with cameras.

4. Practice and skill development:
   - Creating, structuring, expressing, testing and debugging programs in the first of the course's teaching languages. Programming the selection of alternatives according to a string value. Learning new features of an Application Programming Interface, in this case, Myro.
   - On a personal computer, locating and manipulating files produced by a program (IDLE), and importing them into a word processed document when they are images.

Pitfalls:

Don't confuse function calling (round () like fun(…)) with list or array element access ( square [ ] like myList[ n ]). Don't spend more than 20 minutes at a time on a stubborn syntax, semantic or runtime bug. Take a break, try again briefly and then get help (sdc@cs.albany.edu).

Reading:

Begin by reading Chapter 9 of Learning Computing with Robotics, but as usual, TYPE IN and RUN all the examples of code and programs, and experiment, i.e., play with variations. When you are still puzzled, make a note to ask about it in class, send me an email, ask a friend, classmate,
etc. A small amount of reading in ThinkCSPy indicated below is also assigned to learn more about using lists. Some reading notes:

1. We already practiced using lists in the way they are used to store pictures, so that will not be part of the homework. However, Myro provides a convenient operation that constructs an animated gif from a list of pictures. You can put the resulting animated .gifs on web sites, Facebook, email and other messaging systems that transmit images, etc. (That is not part of this homework.)

2. The code due for this assignment follows the pattern of the program on page 228. However, multiple functions must be written as specified below. For this time, a "main" with text input, a call to workers, and output printing is not expected. We add the requirement that the program save the resulting pictures as .jpg's so that you can include them in the homework report. However, the image processing operations given in the textbook are a bit more complicated and subtle than the ones assigned now.

3. Read the material about image understanding and "blobs" in order to crystallize (i.e. learn better or learn more solidly) what you were introduced to in Lab 5. This material will be used in following projects.

**Program and Report Specifications**

1. All your Python/Myro functions must be in one .py file. You must also create and submit a digitally written report that includes 4 images. Use MS Word or any other word processor that supports including images. Then, use Adobe Acrobat (or other software) to "export" your document in .pdf, with the images included. This is very conveniently done on the UAlbany Information Commons and other computers which have the Acrobat processor connected to MS Word. (Email me if there is a problem.) Submit two files: the .py program and the .pdf report.

2. **Create Color Separations, a Negative and a Gray Scale Reduction:** Write a function that accepts a Myro picture as its one argument, returns nothing, and saves three images, Red, Green and Blue using savePicture( .. ) as three .jpg files: Red.jpg, Green.jpg, and Blue.jpg. Each of these three color separations has the same image height and width as the argument picture. Each pixel of say the Red image has its red value gotten from the corresponding pixel in the argument picture, but its green and blue values are zero instead. The Green and Blue image pixels are set in the analogous way.

The above paragraph defines what a "color separation" is. Chapter 9 covers all the other details one needs to do this, but questions will be entertained by email, in class or at tutorial/office hour meetings.

Repeat the same thing in a different function that saves two images: Negative.jpg and Gray.jpg. For each pixel, each color's intensity in Negative.jpg is 255-x where x is that color's intensity in the original image. For Gray.jpg, the R, G and B intensities are equal to (r + g + b)/3 (rounded) where r, g and b are the RGB intensities in the corresponding pixel of the original image. Write in your report a short explanation (written with care for grammar, spelling, punctuation and precise expression) of why my formulas are correct, or reasonable. A full explanation would include each new intensity is in the correct range (1 ... 255) and how each new intensity varies (increases, decreases, etc.) when each original intensity increases.
Include in your report an original .jpg image you took with the Myro camera (get in lab if necessary) plus the three color separation .jpg images produced by using your first function plus the negative and gray scale image from your second function. Of course, first look at the images, evaluate whether they seem correct based on the original, and debug and consult if necessary. Uninformative demonstrations on say gray, monochromatic, all-black or all-white, etc. original images will earn zero points!

3. **Create Histograms:** Read or review pages 89-92 in ThinkCSPy. (In basic Python, lists serve the purpose of arrays.) Write a function that accepts a Myro picture as its first argument and any one of the 4 strings 'red', 'green', 'blue' and 'value' as its second argument. The function will return the histogram of the corresponding pixel values in the form of a list of length 256. Here is a detailed explanation: For each integer V among 0, 1, 2, ..., 255, the integer in the list at position V will be the number (a count) of pixels whose red, green, blue or average value equals (or approximately equals) V, depending on which of the four histograms is specified by the string argument. Note that V denotes an integer that is used BOTH as a list position number AND a value representing a color or brightness intensity. The average of the 3 RGB values for one pixel will have to be rounded and converted to an integer in the case when the 'value' histogram is specified by the string argument. (Search ThinkCSPy or LCR for "types" to help find out about how to do this. Hint: Add 0.5 and convert to an integer.) You must also write a DIFFERENT function that will print the histogram returned by the first function in the form of a 256 line table:

```
0    # of pixels with V == 0, make sure there is a space between the value and the count
1    # of pixels with V==1

....
255  # of pixels with V==255
```

Programming details: The histogram will be represented by a Python list of 256 integers. Write the following code to create the list so (a) its length is 256 and (b) each element in it is initially the integer 0:

```
histogram = [0] * 256  #See ThinkCSPy page 91 for an explanation.
```

Next, code a loop to process every pixel. For each pixel, determine the value V described above, remembering to convert it to an integer in the case of the 'value' histogram. (Hint: Code a conditional using the string parameter to distinguish the different calculations of V for the four different kinds of histograms.) Then, increment the list entry at position V using the code:

```
histogram[ V ] = histogram[ V ] + 1
```
4. **Compute centroids of red, green, blue and value:** Each pixel has a location in the image given by its x and y "locations." (Those x and y values are called "coordinates.") Suppose we consider each pixel $P_{x,y}$ to have a quantity of mass $W_{x,y}$. (In physics, mass is related to weight by a constant multiple depending on your planet. We will therefore call the $W_{x,y}$ values "weights.") The "centroid" or center of gravity has x-coordinate given by

$$\text{Sum of } W_{x,y} \cdot x \text{ over all pixels} \over \text{Sum of } W_{x,y} \text{ over all pixels}$$

and y-coordinate given, similarly, by

$$\text{Sum of } W_{x,y} \cdot y \text{ over all pixels} \over \text{Sum of } W_{x,y} \text{ over all pixels}$$

In words, the x-coordinate is the weighted average of the x-coordinates of the pixels with the weights given by the masses. Program the division so it is done with floating point numbers. Create, include and test a function `printCentroids( pic )` that computes from Picture `pic` and then prints the coordinates of four centroids: Three with the R, G and B values as weights and the fourth with the average RGB value as weight. Your function must print the coordinates in a format like "R-centroid=(..., ...) G-centroid=(..., ...) B-centroid=(..., ...) V-centroid=(..., ...)" so the user can understand the data. Run the function on your image.

5. Copy the data output from 4. into your report. Write into the report how useful you think such centroid data is for making the robot steer toward a goal. Base, explain and clearly justify your opinion based on what you see in your particular image, its color separations and gray scale reduction (all included in the report). Begin by explaining approximately where each of the four centroids is located with respect to visible image features. Outside reading or research is **not** expected, but care and thoroughness in thinking and then expression (including spelling, grammar and punctuation) will be graded.

**Grading:** INDIVIDUAL work! Parts 2, 3, 4 and 5 count for equal weight. Due 10:00PM Friday.