CS 1301

Exam 3 A Spring 2015

Name : ____

Grading TA: _________________________________

- Integrity: By taking this exam, you pledge that this is your work and you have neither given nor received inappropriate help during the taking of this exam in compliance with the Academic Honor Code of Georgia Tech. Do NOT sign nor take this exam if you do not agree with the honor code.

- Devices: If your cell phone, pager, PDA, beeper, iPod, or similar item goes off during the exam, you will lose 10 points on this exam. Turn all such devices off and put them away now. You cannot have them on your desk.

- Academic Misconduct: Academic misconduct will not be tolerated. You are to uphold the honor and integrity bestowed upon you by the Georgia Institute of Technology.
    - Keep your eyes on your own paper.
    - Do your best to prevent anyone else from seeing your work.
    - Do NOT communicate with anyone other than a proctor for ANY reason in ANY language in ANY manner.
    - Do NOT share ANYTHING during the exam. (This includes no sharing of pencils, paper, erasers).
    - Follow directions given by the proctor(s).
    - Stop all writing when told to stop. Failure to stop writing on this exam when told to do so is academic misconduct.
    - Do not use notes, books, calculators, etc during the exam.

- Time: Don't get bogged down by any one question. If you get stuck, move on to the next problem and come back once you have completed all of the other problems. This exam has 6 questions on 10 pages including the title page. Please check to make sure all pages are included. You will have 50 minutes to complete this exam.

I commit to uphold the ideals of honor and integrity by refusing to betray the trust bestowed upon me as a member of the Georgia Tech community. I have also read and understand the requirements outlined above.

Signature: ____

Question	Points	Score
1. Vocabulary	9	
2. Robot Drawing	8	
3. Functional Practice	7	
4. Algorithm Recognition	7	
5. grayLines	10	
6. Summer Interns	10	
Total:	51	

1. *(9 points)*
For each of the following vocabulary terms, write a concise 1-2 sentence definition. Be brief, and to the point.

(a) [3 pts] operator

> **Solution:** Operators are special tokens that represent computations like addition, multiplication and division.

(b) [3 pts] pixel

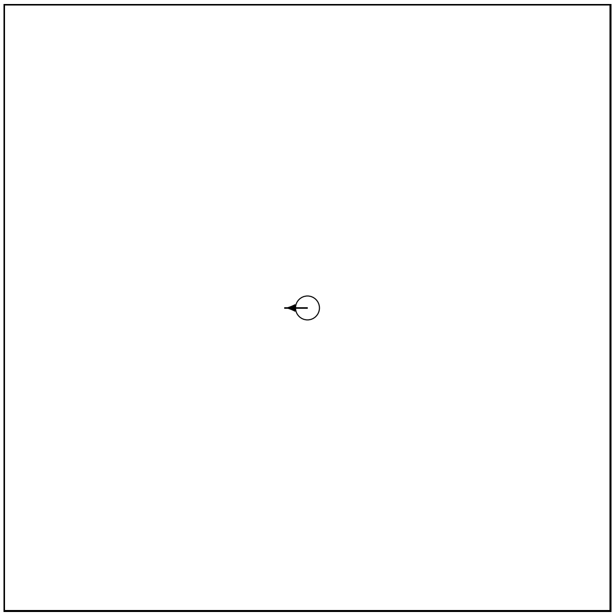
> **Solution:** pixel - An image/picture (when stored in a computer) is made up of tiny elements called Pixels. Each pixel contains color information (Red, Green and Blue) values for a small point in the image, and is located at a specific X/Y location within the overall image.
>
> Looking for: Small elements that make up an image. Contain color information.

(c) [3 pts] robot

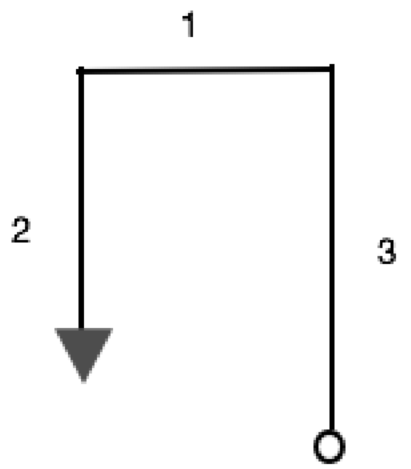
> **Solution:** Any mechanism that is guided by automated control.

2. *(8 points)*
Draw the robot's trajectory when the following code is executed. Use a triangular arrow to indicate the robots final "forward" direction. Assume that the robot starts in the middle facing towards the left of the paper ("west"). Assume that if you turn for one second the robot turns exactly 90 degrees. Assume that if you drive forward or backwards for one second the robot travels exactly 1 cm. Label each line segment drawn with the number of cm it is meant to represent. (Try to draw lines to scale.)

```
from Myro import *
init()
x=1
y = 1
aList = [(1,2),3,4,(5,6),(7,8)]
for i in range(len(aList)):
     if (i%x==0):
         turnRight(1,1)
     else:
         turnLeft(1,2)
         backward(1,1)
     if (i in [1,4]):
         y= y * -1
     if (y==1):
         forward(1,2)
         x = x + 1
```



Solution:



Grading:
+1 for having the first line go up +1 First line is 3 cm (2cm + 1cm) +1 Second line goes left +1 Second line is 1 cm +1 3rd line goes down +1 3rd line is 2 cm +1 Final arrow faces DOWN +1 Getting it all correct -1 points for extra line segments or corners.

3. *(7 points)*
Examine the following python code that uses functional programming.

```
a = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
b = filter(lambda x: x % 2 == 0, a)
c = map(lambda x: x**3, b)
d = reduce(lambda x,y: x+y, c)
```

Tell us what each of the above variables refers to after the code is executed:

A: ______________________________

B: ______________________________

C: ______________________________

D: ______________________________

Solution: Grading: +1 for A, +2 points for B/C/D. A: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9] B: [0,2,4,6,8] C: [0, 8, 64, 216, 512] D: 800

4. *(7 points)*
Pretend you are the python interpreter. Write down what is printed when the following code is executed beside the code. Draw a box around your final answer.

```
def mysterySortA(aList):
   for index in range(1,len(aList)):
      currentvalue = aList[index]
      position = index

      while position>0 and aList[position-1] > currentvalue:
          aList[position] = aList[position-1]
          position = position-1

      aList[position]=currentvalue
      print(aList)

aList = [93,54,26]
mysterySortA(aList)
```

What sorting algorithm is mysterySortA using?: ____________________

What Big O complexity class is this code?: ____________________

Solution: mysterySortA: InsertionSort (+1), BigO of N squared (+1) Printouts: +1 each correct line. [54, 93, 26] [26, 54, 93]

Pretend you are the python interpreter. Write down what is printed when the following code is executed beside the code. Draw a box around your final answer.

```
def mysterySearch(aList, aNum):
   position = 0
   for item in aList:
      if item == aNum:
         return position
      position = position + 1
   return -1

index = mysterySearch([9,3,2,6,1], 6)
print("index is:", index)
```

What is the name of this search? ____________________

What is the Big-O complexity class of this search? ____________________

Solution: +1 point for the printout (”index is: 3”) +1 point for ”Sequential” or ”linear” search +1 point for O(N)

5. *(10 points)*

Write a function named `grayLines` that accepts one parameter representing the name of a picture file on your disk as a string.

The function should load the picture, and convert every other column (i.e. columns 1,3,5,7, and so on) to grayscale. To do this, you should take the red, green, and blue values for each pixel, find their average, and place that average value back into all three color channels (r,g,b) of the pixel.

Then, it should save the picture with the name "grayLines.jpg". You may assume that the Myro picture functions are already imported.

`Example call: >>> grayLines("myPicture.jpg")`

Solution:

```
def grayLines(p):
    pic = makePicture(pic)
    for x in range(1, getWidth(pic), 2):
        for y in range(getHeight(pic)):
            pixel = getPixel(pic, x, y)
            red = getRed(pixel)
            green = getGreen(pixel)
            blue = getBlue(pixel)
            avg = (red + green + blue) / 3
            setRed(pixel, avg)
            setGreen(pixel, avg)
            setBlue(pixel, avg)
    savePicture(pic, "greyLines.jpg")
```

```
Grading:
+1 correct function header
+2 converting the parameter to a picture object
+1 for generating every other X value (1,3,5...)
+1 for generating every Y value (0....picWidth)
+1 for getting the pixels from the picture
+2 calculating average color value
+1 setting R,G,B values of each pixel to the average color value
+1 correctly saving picture
```

6. *(10 points)*
Write a function named **summerInterns** that accepts the name of a file to open as a string parameter. The function should open the file, which will be formatted as follows (one Employee, ID per line). You may assume that names all start with capital letters. Note that the lines may or may not have spaces after the comma:

```
Rachel,2300
Richard, 1020
Daniel,2301
Jorge, 231
David,2310
David,2315
```

Your function must read in the file and return a list of ONLY the names in Alphabetical order of the Summer Interns. You know that in order to be considered a Summer Intern your ID has to start with the number 23 (Example: 2310). Remember to close your file after you are finished reading it. If you find duplicate Employees (Same Name), DO NOT include duplicates in the list! For example, when using the above input, the function would work as follows:

```
>>> interns = summerInterns("Employees.txt))
>>> interns
['Daniel', 'David', 'Jorge',  'Rachel]
```

Solution:

```
def summerInternsOnly(filename):
    f = open(filename, "r")
    lines = f.readlines()
    f.close()

    myList = []

    for line in lines:
        newLine = line.split(",")
        ID = newLine[1].strip()
        name =  newLine[0].strip()
        if(ID[0:2] == "23" and (name not in myList)):
            myList.append(name)
    myList.sort()
    return(myList)
```

```
Grading:

+1 point for correct header (correct name of function and one parameter)
+1 point for opening the file in read mode
+2 points for going through all the lines in file
+1 point split each line
+1 checking that the ID starts with 23
+1 for checking the name is not a duplicate.
+1 sorting the list
+1 points for returning the list
+1 point for closing the file
 Misc minus: -1 for not taking into account a possible space before the ID
  -1 for syntax errors.

TOTAL: 10 points
```