Name : _____
Section 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 4 questions on 8 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. Multiple Choice	12	
2. Sorting	12	
3. Read Coordinates	10	
4. Draw Points	8	
Total:	42	

1. *(12 points)*

For each of the following multiple choice questions, indicate the most correct answer by circling it!

(a) [1 pt] Convert 129_{10} to binary (base 2):

A. 01000011 B. 10000011 C. 01000111 D. 11001101 **E. 10000001** F. 01010101

(b) [1 pt] Convert 11000010_2 to decimal (base 10):

A. 194 B. 195 C. 196 D. 197 E. 42

(c) [1 pt] Which of the following will return [2,4,6,8,10]?

A. map(lambda x: x+1, range(0,10,2))
B. filter(lambda x: x%2 == 1, range(2,11))
C. reduce(lambda x,y: x**y, range(100))
D. filter(lambda x: x%2 == 0, range(1,11,2))
E. reduce (lambda x,y: x+y, [2,4,6,8,10])
F. map(lambda x: x+1, range(1,11,2))
G. map(lambda x: x%2 == 0, [2,4,6,8,10])

(d) [1 pt] An error in a program that makes it impossible to parse – and therefore impossible to interpret, is a:

A. Syntax Error B. Semantic Error C. Runtime Error D. Parse Error

(e) [1 pt] An error (in code) that leads to unexpected behavior. The program functions correctly (does what the code says) but the code does not actually perform the action that the programmer intended, is a:

A. Syntax Error **B. Semantic Error** C. Runtime Error D. Unexpected Error

(f) [1 pt] An error raised by the python interpreter while the program is executing if something goes wrong. For example, a divide by zero error, is a:

A. Syntax Error B. Semantic Error **C. Runtime Error** D. Interpreter Error

(g) [1 pt] Opening a file without specifying the second parameter opens it for:

A. reading B. writing C. overwriting D. appending E. None Of These

(h) [1 pt] What would the `a` variable point at after the following code is executed?

```
try:
  a = 2
  if a % 2 == 0:
     a = a+3
  a = a / 0
  a = a * 2
except:
  a = a + 3
  a = a / 2
```

A. 6.5 **B. 4.0** C. 3.5 D. 3.0 E. 2.5 F. 2.0 G. 1.5

(i) [1 pt] Which of the following is not a valid dictionary key?

A. 4, B. (4) C. 4 D. "4" **E. [4]**

(j) [1 pt] Which of the following is true about the keys in a dictionary?

A. An integer can be a key.
B. A list can be a key.
C. A dictionary can be a key.
D. All keys must be mutable.
E. All of the above are False.

(k) [1 pt] Which feature do Tuples and Strings have in common?

A. They are immutable
B. They are sequences
C. They can be used as dictionary keys
D. All of the above
E. None of the above

(l) [1 pt] What is the correct chronological order of the following inventions?

A. microprocessor, transistor, loom, mechanical duck, Harvard Mark 1 Computer
B. mechanical duck, loom, Harvard Mark 1 Computer, transistor, microprocessor
C. loom, mechanical duck, transistor, vacuum tube, Ethernet
D. vacuum tube, Ethernet, ARPA Net, PDP11 Computer, transistor

2. *(12 points)*
Here is a sequence of numbers: 1, 10, 17, 3, 5, 19, 17, 7, 2

(a) [4 pts] Illustrate how a bubble-sort would sort the above list of numbers. After each pass, underline the numbers that are guaranteed to be in sorted order. Do all passes, do not make short-cutting optimizations.

Solution: 1, 10, 3, 5, 17, 17, 7, 2, 19
1, 3, 5, 10, 17, 7, 2, 17, 19
1, 3, 5, 10, 7, 2, 17, 17, 19
1, 3, 5, 7, 2, 10, 17, 17, 19
and so on!

Grading:
+4 points for sorting the list with a recognizable bubblesort algorithm.
Misc things to take off for:
- Don't bubble the front half of the list correctly -1
- Don't skip the last 3 passes (which do not change the list) -1
- Didn't underline correct numbers: -1

(b) [4 pts] Illustrate how an insertion-sort would sort the above list of numbers. After each pass, underline the numbers that are guaranteed to be in sorted order.

Solution: 1, 10, 17, 3, 5, 19, 17, 7, 2
1, 10, 17, 3, 5, 19, 17, 7, 2
1, 3, 10, 17 , 5, 19, 17, 7, 2
1, 3, 5 , 10, 17 , 19, 17, 7, 2
1, 3, 5 , 10, 17,19 , 17, 7, 2
1, 3, 5 , 10, 17,17, 19 , 7, 2
1, 3, 5 , 7 , 10, 17,17, 19 2
1, 2, 3, 5 , 7 , 10, 17,17, 19

Grading:
4 points for doing insertion sort correctly.
2 points for doing selection sort instead of insertion sort.
0 points for not doing it right.

(c) [4 pts] Illustrate how a merge-sort would sort the above list of numbers.

Solution: Step 1 : break them up into 9 lists of one element each.
Step 2 : merge pairs together (leaving one out)
step 3 : merge 2's into 4's (one 2 merges with the single to make 3)

Step 4 : merge 4's into 8's, etc...

Grading:
4 points for doing it right. -1 point for starting with lists of size 2 instead of size 1 -1 point if they "forget" a number to get an even number of items.

3. *(10 points)*

Write a function called **readCoordinates** that will take in a string that holds the name of the file to open as its parameter.

Your function should open the named file, which will consist of pairs of numbers seperated by a comma. These pairs of numbers represent X and Y coordinates as integers. After reading the numbers, your function should return a list of tuples, where each tuple consists of the X,Y coordinate on a line, in the same order as they are found in the file, stored as integers.

If this was the comments of an example "points.txt" file:

```
0,0
1,1
3,1
4,0
```

Then calling your function like this would produce the following output:

```
>>> result = readCoordinates("points.txt")
>>> result
 [ (0,0), (1,1), (3,1), (4,0) ]
```

Solution:

```
def readCoordinates(filename):
    f = open(filename, 'r')

    pointList = []
    for line in f.readlines():
        item = line.split(",")
        x = int(item[0])
        y = int(item[1])
        point = (x,y)
        pointList.append( point )

    return pointList

Grading:
+1 point for correctly opening the file.
+1 point for reading all lines (readlines or readline)
+1 points for splitting the line on commas
+2 points for converting both numbers to ints (or floats)
```

```
+1 points for correctly creating the tuple with both x,y in it.
+1 point for appending tuple to the list
+1 point for returning a list.
+1 point for returning the correct result (in the list).
+1 point for closing the file
```

4. *(8 points)*

Write a function named **drawPoints** which takes a picture object and a list of points as parameters. The list will contain tuples with (x,y) values as integers. Your function should draw black pixels on the picture at each of the specified points. Your function should return None. For example, if your drawPoints function was called as in the following example, it would produce the image shown:

```
>>> p = makePicture(5,5)
>>> PointList = [ (0,0), (1,1), (3,1), (4,0), (0,3), (1,4), (2,4), (3,4), (4,3) ]
>>> drawPoints(p, PointList )
>>> show(p)
```

Solution:

```
def drawPoints(p, pointList):
  for point in pointList:
     x = point[0]
     y = point[1]
     pix = getPixel(p, x,y)
     setRGB(pix, (0,0,0) )  # Or 3 calls to setRed/setGreen/setBlue
```

Grading:
+1 for correct header.
+1 for iterating through the points in the list
+2 for correctly extracting the X and Y from each tuple.
+1 for getting/finding the correct pixel for each point.
+1 for coloring any pixel black.
+2 for coloring all of the CORRECT pixels.
(-1 if they return something other than none)
(-1 if they use makePicture or takePicture or copyPicture instead of working on the picture that was passed in.)