CS1301 - Exam2 Name: _____ Section: _____

Instructions:

- Please write clearly. What I cannot read, I will not grade.
- Show all your work in detail. I give partial credit.
- This exam has 11 pages including the title page. Please check to make sure all pages are included.
- This exam is closed book, closed notes, no calculators.
- Don't get bogged down on any one question. 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.

Signature: _____

Question	Points	Score
1. Vocabulary	21	
2. Fill in the Blanks	4	
3. Python Expressions	22	
4. Robot Drawing	9	
5. Find the Error	3	
6. Leaky Pipes	6	
7. countUpBy	8	
8. Average a List	8	
9. Save Light Values	10	
Bonus Questions	0	
Total:	91	

Vocabulary Questions

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

   (a) (3 points) block

   Solution: One or more program statements that share the same level of indentation.

   (b) (3 points) dictionary

   Solution: A mutable compound data type that associates keys with values.

   (c) (3 points) flow of execution

   Solution: The order in which statements in a program are executed. Function calls, return statements, conditionals and loops all modify the standard top to bottom flow of execution.

   (d) (3 points) function

   Solution: A named sequence (block) of statements that performs some useful operation. Functions may or may not take parameters and may or may not produce a result.

   (e) (3 points) recursion

   Solution: recursion - The process of calling the function that is currently executing.

   (f) (3 points) slice

   Solution: A subsequence copied from a sequence specified by a range of indices. The slice operator is: sequence[start:stop].

   (g) (3 points) traverse

   Solution: To move through all elements of a set, performing a similar operation on each element.

2. (4 points) Fill in the blanks:

   Python has several compound data types that we have learned about. A _____ can be used to store a sequence of characters, while a _____ can store a sequence of any type of data (but is immutable). A _____ can also store any type of data, and allows you to change elements within it. A _____ associates keys to values.

Solution: Python has several compound data types that we have learned about. A **string** can be used to store a sequence of characters, while a **tuple** can store a sequence of any type of data (but is immutable). A **list** can also store any type of data, and allows you to change elements within it. A **dictionary** associates keys to values.

Code Understanding Questions

3. Python Expressions - For this question, assume the following statements have already been entered and interpreted:

```
a = [ 10, 32, 42, True, ["Ivy", "Oak", "Fern"], 3.14159, [ 10, 11, 12], 4]
b = a
c = a[0:4]
d = a[4]
d[2] = "Palm"
```

Act like the python interpreter and evaluate the following expressions, writing the value they evaluate to:

(a) (2 points) `a[0]`

Solution: 10

(b) (2 points) `3+2`

Solution: 5

(c) (2 points) `len(a)`

Solution: 8

(d) (2 points) `a[6][10]`

Solution: IndexError: list index out of range

(e) (2 points) `d`

Solution: ['Ivy', 'Oak', 'Palm']

(f) (2 points) `c`

Solution: [10, 32, 42, True]

(g) (2 points) `a[4][2]`

Solution: 'Palm'

(h) (2 points) `b[:2]`

Solution: [10, 32]

(i) (2 points) `b[-2]`

Solution: [10, 11, 12]

(j) (2 points) `c[-2]`

Solution: 42

(k) (2 points) `a[4] + [1,3,5]`

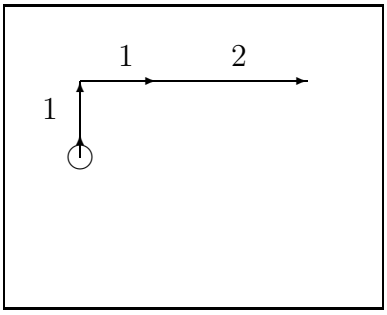
Solution: ['Ivy', 'Oak' , 'Palm' , 1 , 3, 5]

4. (9 points) Robot Drawing - Assume `turn90degrees()` has been defined as below so the robot turns right 90° and `nudge(x)` has been defined to move the robot forward x units.

```
def turn90degrees():
    turnRight(1, 1)

def nudge(x):
    forward(1, x)
```

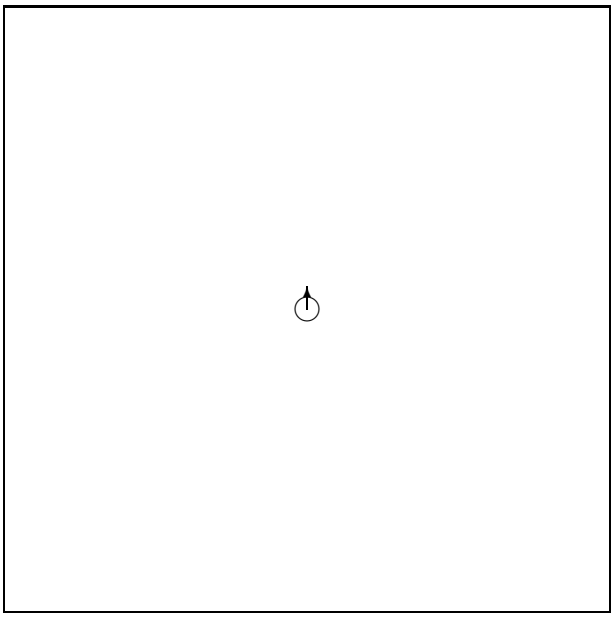
The following code makes the robot drive the trajectory drawn in the box to the right.

```
nudge(1)
turn90degrees()
nudge(1)
nudge(2)
```

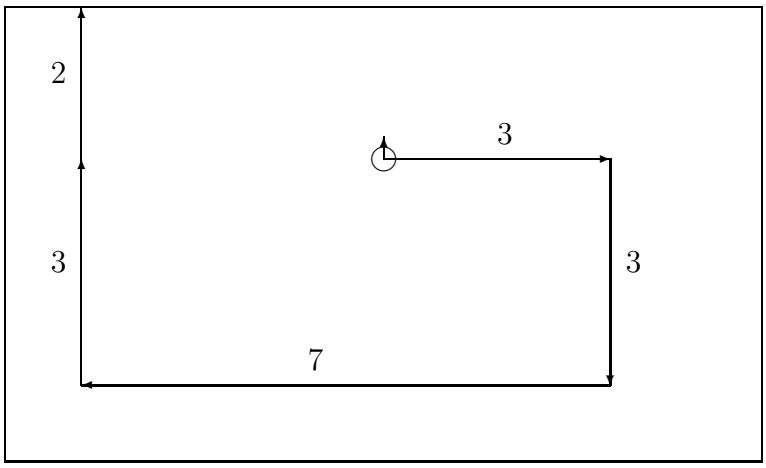


Draw the robot's trajectory when the following code is executed. Label the length of each move (nudge) using numbers as in the example above.

```
def turn90degrees():
    turnRight(1, 1)

def nudge(x):
    forward(1, x)

turns = [2, 6]

for idx in [2,2,6,2,1]:
    if idx in turns:
       turn90degrees()
    nudge(idx + 1)
```



Solution:



Grading: 1 point for each correct line segment and length. 1 point for each correct turn.

5. (3 points) Find the Error: The following code contains a statement that will cause a runtime error. Circle the line and explain what's wrong.

```
e = "2.718"
pi = 3.14
pie = str(pi) + e
print int(e)
print int(pi)
print pie
```

Solution: Line 4 (print int(e)) contains the error. You cannot convert a string into an integer if it's a floating point number. 1 point for identifying the line, 2 points for explaining what is wrong.

6. (6 points) Leaky Pipes - What is printed by the following function if it is called with an input of 12?

```
>>> leakyPipes( 12 )

def leakyPipes(n):
   if (n > 0):
      if (n % 4 == 0):
         print "drip %d" % n
         leakyPipes(n-3)
      if (n % 3 == 0):
         print "drop %d" % n
```

Solution:

```
drip 12
drop 9
drop 12
```

2 points for each line.

Code Writing Questions

7. (8 points) Write a function called `countUpBy` that accepts a single integer parameter and uses a while loop to print out a count from that number up to twenty (inclusive) by that number. You may assume that your input will be between one and 20 (inclusive).

Examples:

```
>>> countUpBy(5)
5
10
15
20
```

```
>>> countUpBy(7)
7
14
```

Solution:

```
def countUpBy(n):
   x = n
   while x <= 20:
      print x
      x = x + n
```

Grading: 1 point for correct def line, 2 points for starting at the number, 2 points for the while loop test, 2 points for incrementing correctly, 1 point for correct output (print not return)

8. (8 points) Average a List - Write a function called `average` that accepts a list of numbers (they may be ints or floats). It should **return** the average (mean) value of all the numbers in the list. If the list is empty, it must return `None`.

   For example:

```
>>> result = average( [10, 5, 5] )
>>> print result
 6.6666666
```

Solution:

```
def average( aList ):
    if ( len(aList) == 0 ):
       return None

    count = 0
    sum = 0
    for item in aList:
        sum = sum + item
        count = count + 1
    return sum / float(count)
```

Grading:
1pt - Correct def statement
2pt - Correctly sums list items
1pt - correctly counts list items
2 pts - divides sum by count (1 pt if they don't convert one of them to a float!)
2 pt - returns None for empty list

9. (10 points) Write a function named `saveLightValues` that accepts the name of a file to open as a string parameter. The function should open the file for writing, and save ten light values from calling the `getLight("center")` function, one per line. Between calls to the `getLight()` function to get light samples, the robot should turn left a small amount (you choose the speed and duration).

Solution:

```
def saveLightValues( aName):
   myFile = open(aName,"w")
   for i in range(10):
      value = getLight("center")
      turnLeft(1,1) # or any other speed/duration
      myFile.write( str(value) )
      myFile.write("\n")

   myFile.close()
```

Grading: 1 point for correct header, 1 point for opening file, 1 point for using "write" mode. 2 points for looping 10 times (or repeating the code 10 times!) 1 point for reading light value. 1 point for writing light value (as a string!). 1 point for putting in a newLine. 1 point for turningLeft every time. 1 point for closing the file handle when done.

10. (2 points (bonus)) Bonus Questions:
    (a) What did you name your robot?

    (b) What has been the most difficult topic or concept in this class for you to understand (what should we spend more time on)?