6.00 Quiz 1, 2011

Name _____

1. 15 /15
2. 7 /10
3. 10 /10
4. 12 /18
5. 8 /8
6. 10 /13
7. 7 /15
8. 0 /9
9. 1 /1
10. 1 /1

Total _____ /100

This quiz is open book and open notes, but do not use a computer.

Please **write your name on the top of each page**. Answer all questions in the boxes provided.

1) Are each of the following True or False (15 points)

F 1.1. In Python the **values** of a dict must be immutable.

T 1.2. There exist problems that **cannot** be solved in Python **without** using either iteration or recursion.

F 1.3. Floating point arithmetic behaves exactly like normal arithmetic on real numbers.

F 1.4. On all inputs, a bisection search will run faster than a linear search.

F 1.5. Let `L` be a list, each element of which is a list of ints. In Python, the assignment statement `L[0][0] = 3` mutates the list `L`.

Name _____

2) What does the following code print? (10 points)

```
T = (0.1, 0.1)
x = 0.0
for i in range(len(T)):
    for j in T:
        x += i + j
        print x
print i
```

0 → 1
0 → 1

0 + 0.1

0
1
1
2
1

0.1
0.2
1.3
2.4
1

x = 0.1
x = 0.2
x = 1.3

Name _____

3) What does the following code print? (10 points)

```
def f(s):
    if len(s) <= 1:
        return s
    return f(f(s[1:])) + s[0]   #Note double recursion

print f('mat')
print f('math')
```

f(f(at)) + m s[0] = m
f(ta) + m = atm
+ m

s[1:] = at
f(at) = f(f(t)) + a

f(f(t)) =
f(t) = t

f(ta) = f(f(a)) + t
> at

Name _____

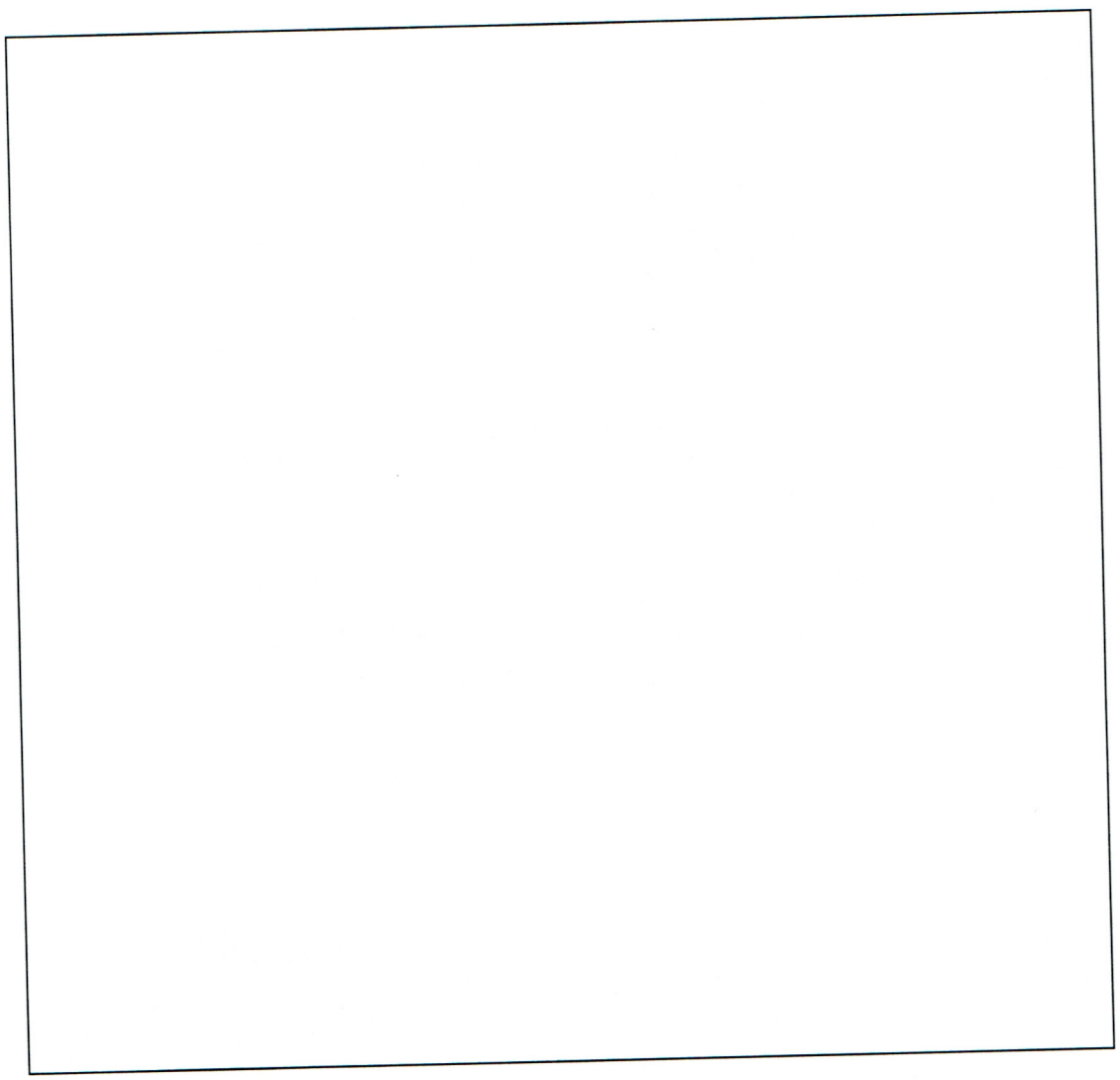
4) Implement the body of the function specified in the box. (18 points)

```
def findAll(wordList, lStr):
    """assumes: wordList is a list of words in lowercase.
                lStr is a str of lowercase letters.
                No letter occurs in lStr more than once
       returns: a list of all the words in wordList that contain
                each of the letters in lStr exactly once and no
                letters not in lStr."""
    goodwords = []
    for i in range(len(wordlist))
        limit = len(lStr)
        for j in range(len(wordlist[i]))
            for m in range(len(lStr))
                if wordlist[i][j] == lStr[m]
                    limit -= 1
        if limit == 0
            goodwords.append(wordlist[i])
```

Name _____

5) The following code does not meet its specification. Correct it. (8 points)

```
def addVectors(v1, v2):
    """assumes v1 and v2 are lists of ints.
       Returns a list containing the pointwise sum of
       the elements in v1 and v2.  For example,
       addVectors([4,5], [1,2,3]) returns [5,7,3],and
       addVectors([], []) returns []. Does not modify inputs."""
    if len(v1) > len(v2):
        result = v1.copy()
        other = v2.copy()
    else:
        result = v2.copy()
        other = v1.copy()
    for i in range(len(other)):
        result[i] += other[i]
    return result
```



6) Consider the following code:

```
def f(s, d):
    for k in d.keys():
        d[k] = 0
    for c in s:
        if c in d:
            d[c] += 1
        else: d[c] = 0
    return d

def addUp(d):
    result = 0
    for k in d:
        result += d[k]
    return result

d1 = {}
d2 = d1
d1 = f('abbc', d1)
print addUp(d1)
d2 = f('bbcaa', d2)
print addUp(d2)
print f('', {})
print result
```

{} = {0,0,0,0}
0 = {1,1,1,1,0}
4
{}
error

6.1) What does it print? (9 points)

6.2) Does it terminate normally? Why or why not? (4 points)

Name _____

7) Consider the following code:

```
def logBase2(n):
    """assumes that n is a positive int
       returns a float that approximates the log base 2 of n"""
    import math
    return math.log(n, 2)

def f(n):
    """assumes n is an int"""
    if n < 1:
        return
    curDigit = int(logBase2(n))
    ans = 'n = '
    while curDigit >= 0:
        if n%(2**curDigit) < n:
            ans = ans + '1'
            n = n - 2**curDigit
        else:
            ans = ans + '0'
        curDigit -= 1
    return ans

for i in range(3):
    print f(i)
```

7.1) What does it print? (10 points)

None
1
10

7.2) Under the assumption that `logBase2` is `O(n)`, what is the order (use big Oh notation) of `f`? (5 points)

8) Next to each item in the left column write the letter labeling the item in the right column that best matches the item in the left column. No item in the right column should be used more than once. (9 points)

[e]	Big O notation	a) induction
[a]	Newton's method	b) upper bound
[f]	recursion	c) lower bound
		d) approximation
		e) expected running time
		f) exponential

9. Do you think that the lectures are too slow paced, too fast paced, about right? (1 point)

10. Do you think that the problem sets are too easy, too hard, about right? (1 point)

6.00SC Introduction to Computer Science and Programming
Spring 2011

6.00 Quiz 2, 2011

Ryan Mooney
Name

1. 15 /15
2. 18 /18
3. 16 /16
4. 0 /10
5. 19 /19
6. 0 /10
7. 3 /12

Total 73 /100

This quiz is open book and open notes, but do not use a computer.

Please **write your name on the top of each page**. Answer all questions in the boxes provided.

1) Are each of the following True or False (15 points)

T 1.1. In Python, classes **do not** enforce information hiding.

F 1.2. In Python, classes **cannot** be used as a parameter of a function.

F 1.3. Increasing the number of buckets in a hash table typically **increases** the amount of time needed to locate a value in the table.

T 1.4. "Normal distribution" and "Gaussian distribution" are different names for the same thing.

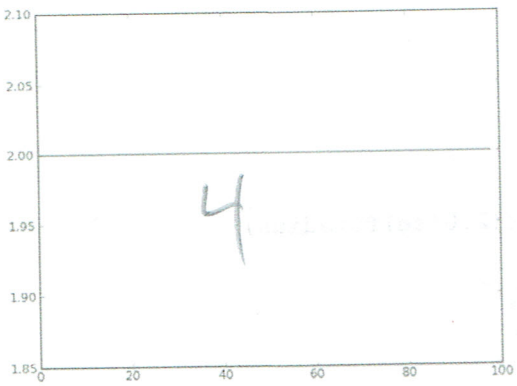
F 1.5. "Standard deviation" and "coefficient of variation" are different names for the same thing.

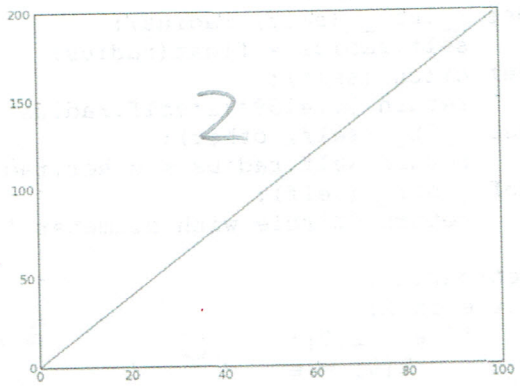
Carol Kelleher
113 Cherrington Dr.
Cranberry Twp., 16066

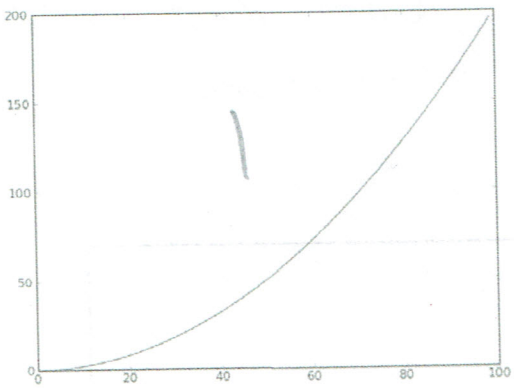
Name _____

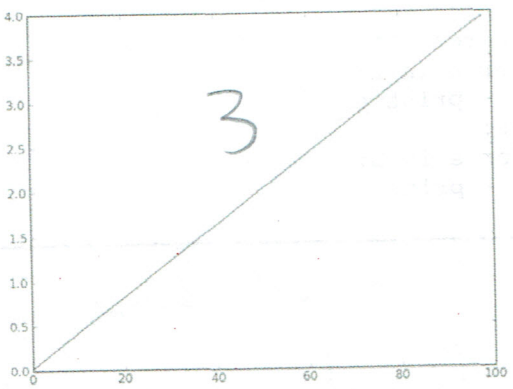
3) The code produces 4 plots. Match each plot below to a figure number by writing 1, 2, 3 or 4 **on the plots**. (16 points)

```
y1, y2, y3, y4 = [], [], [], []
for i in range(100):
    y1.append((i**2)/50.0)
    y2.append(2*i)
for i in range(99):
    y3.append(y1[i+1] - y1[i])
    y4.append(y2[i+1] - y2[i])
pylab.figure(1)
pylab.plot(y1)
pylab.figure(2)
pylab.plot(y2)
pylab.figure(3)
pylab.plot(y3)
pylab.figure(4)
pylab.plot(y4)
```









Name

5) Write a function that uses a Monte Carlo simulation to find the probability of a run of at least 4 consecutive heads out of ten flips of a fair coin, and then returns that probability. Assume that 10,000 trials are sufficient to provide an accurate answer. You may call the function :

```
def simThrows(numFlips):
    """Simulates a sequence of numFlips coin flips, and returns True if
       the sequence contains a run of at least four consecutive
       heads and False otherwise."""
```

(19 points)

def sim(): #write your code below

```
passes = 0
for i in range(10000)
    if simThrows(10)
        passes += 1
prob = passes / 10000
return prob
```

Name

6) If `pylab.polyfit` is used to fit an n^{th} degree and an $(n+1)^{th}$ degree polynomial to the same data, is one guaranteed to provide a least squares fit that is at least as good as the other? If so, which and why? If not, why not? (10 points)

No, if the degree is diff, it will get more off as it 1\

Name _____

7) Next to each item in the left column write the letter labeling the item in the right column that best matches the item in the left column. No item in the right column should be used more than once, and no box should contain more than one letter. (12 points)

[g, f] data abstraction	a) inheritance
[d, b] merge sort	b) divide and conquer
[a] polymorphism	c) O(log n)
[c, e] hashing - constant	d) O(n)
	e) O(1)
	f) specification
	~~g)~~ mutability

8) Do you think that the lectures are too slow paced, too fast paced, about right?

Too slow 1 2 3 4 5 Too fast

9) Do you think that the problem sets are too short, too long, about right?

Too short 1 2 3 4 5 Too long

6.00 Quiz 3, Spring 2011

Ryan Mooney
Name

1. 12 /15
2. 10 /10
3. 10 /15
4. 12 /18
5. 5 /5
6. 7 /17
7. 5 /5
8. 0 /5
9. 10 /10

Total 76 /100

This quiz is open book and open notes, but do not use a computer (or cell phone!). You have 120 minutes.

Please **write your name on the top of each page**. Answer all questions in the boxes provided.

1) Are each of the following True or False? (15 points)

F	1.1. Dynamic programming provides an `O(n)` solution to the 0/1 knapsack problem, where `n` is the number of items to be considered.
T	1.2. K-means clustering is more computationally efficient than hierarchical clustering.
T	1.3. The depth of a decision tree is related to the number of independent decisions it records.
T	1.4. Hierarchical clustering is deterministic, but k-means clustering is not.
T	1.5. When used to find a root of a polynomial, Newton's method converges in `O(log(n))` iterations, where n is the degree of the polynomial.

Name _____

3) Consider the following code.

```
def throwNeedles(fcn, numNeedles = 100000):
    inCircle = 0
    estimates = []
    for Needles in xrange(1, numNeedles + 1, 1):
        x = fcn(0, 1)
        y = fcn(0, 1)
        if (x*x + y*y) <= 1.0:
            inCircle += 1
    return 4*(inCircle/float(numNeedles))

print throwNeedles(random.uniform)
print throwNeedles(random.gauss)
print throwNeedles(random.random)
```

$x^2 + y^2 = 1$
$x = \sqrt{\frac{1}{2}}$
$x = 0.7$
1.4
2.8

[b] 3.1. With high probability, it will first print a value that is approximately equal to
- a. `0.5*pi`
- b. `pi`
- c. `2*pi`
- d. `4*pi`
- e. None of the above

[c] 3.2. With high probability, it will next print a value that is
- a. less than the first value printed
- b. about equal to the first value printed
- c. greater than the first value printed

[e] 3.3. With high probability, it will next print a value that is approximately equal to
- a. `0.5*pi`
- b. `pi`
- c. `2*pi`
- d. `4*pi`
- e. None of the above

(15 points)

Name _____

5) To get a sense of the quality of its dormitories, MIT surveyed all students living in a dorm. This is an example of (select one): (5 points)

[b]

a) The Texas sharpshooter fallacy

b) Sample bias

c) GIGO

d) A well-designed study

The following questions all refer to the code you were asked to study in preparation for this exam. A copy of the posted code is at the end of this quiz. Feel free to detach it.

Name

6.2) Do you believe Ryan's conjecture? Why or why not? (7 points)

Yes, because it would make a left shifted distribution of weight times favoring less waiting

Name _____

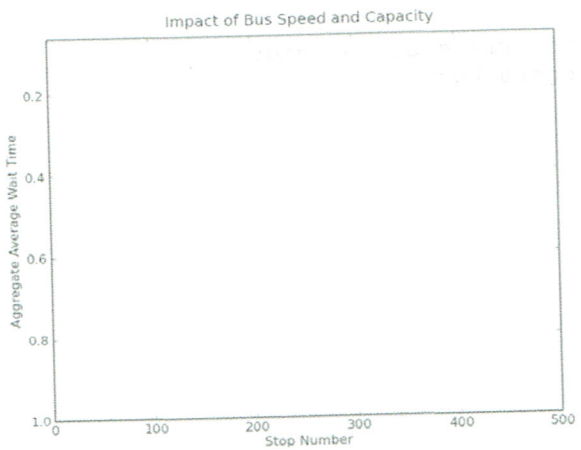
9) Match each of the plots below with a test that could have produced it, assuming that the statement `pylab.legend(loc = 'best')` were removed from `test`. You may not use the same plot more than once. (10 points)

`test([20], [0], 20)` [c]

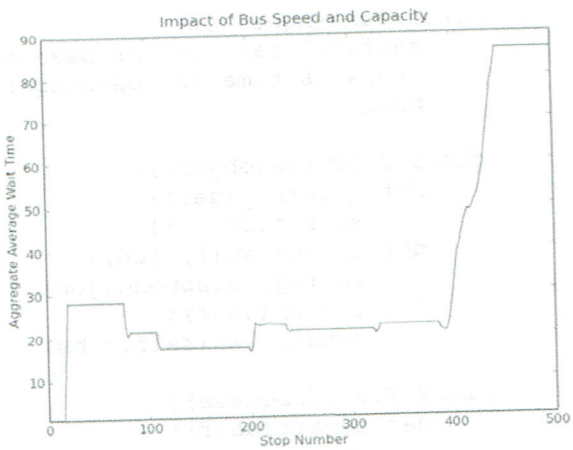
`test([0], [20], 20)` [d]

`test([1], [200], 1)` [b]

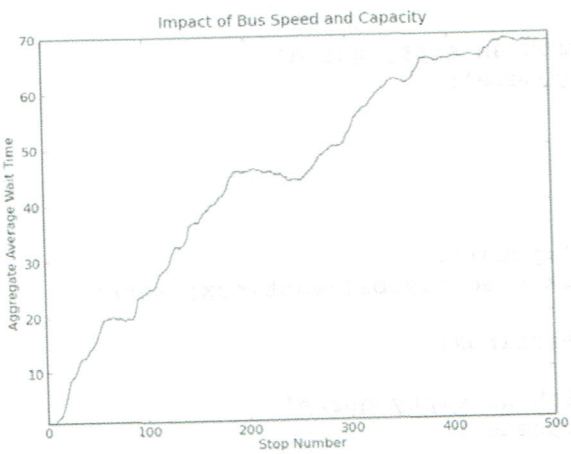
A



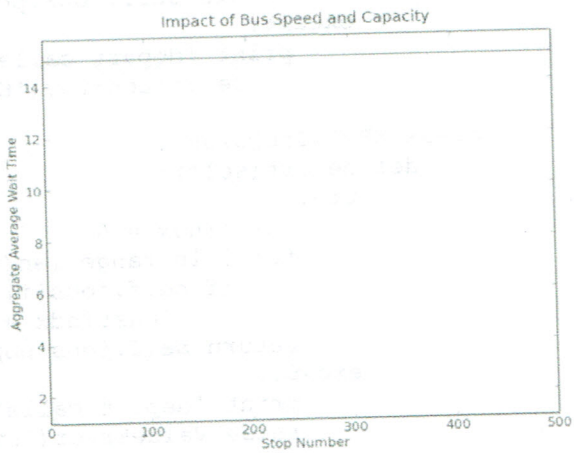
B



C



D



```
def test(capacities, speeds, numTrials):
    random.seed(0)
    for cap in capacities:
        for speed in speeds:
            totWaitTimes = pylab.array([0.0]*500) #keep track of 1st 500 stops
            totLeftWaiting = 0.0
            for t in range(numTrials):
                aveWaitTimes, leftWaiting = simBus(Bus(cap, speed))
                totWaitTimes += pylab.array(aveWaitTimes[:500])
                totLeftWaiting += leftWaiting
            aveWaitTimes = totWaitTimes/numTrials
            leftWaiting = int(totLeftWaiting/numTrials)
            lab = 'Spd = ' + str(speed) + ', Cap = ' + str(cap)\
                  + ', Left = ' + str(leftWaiting)
            pylab.plot(aveWaitTimes, label = lab)
    pylab.xlabel('Stop Number')
    pylab.ylabel('Aggregate Average Wait Time')
    pylab.title('Impact of Bus Speed and Capacity')
    ymin, ymax = pylab.ylim()
    if ymax - ymin > 200:
        pylab.semilogy()
    pylab.ylim(ymin = 1.0)
    pylab.legend(loc = 'best')
```

```
def simBus(bus, numStops = 6, loopLen = 1200, meanArrival = 90,
           meanWork = 10, simTime = 50000):
    assert loopLen%numStops == 0
    stops = []
    for n in range(numStops):
        stops.append(BusStop())
    time, totWait, totPassengers, lastArrival = [0.0]*4
    aveWaitTimes = []
    nextStop, busLoc, time = [0]*3
    nextJob = Passenger(meanArrival, meanWork)
    while time < simTime:
        #advance time and move bus
        time += 1
        for i in range(bus.getSpeed()):
            busLoc += 1
            if (busLoc)%(loopLen/numStops) == 0:
                break
        #see if there is a passenger waiting to enter queue
        if lastArrival + nextJob.interArrival() <= time:
            #passengers arrive simultaneously at each stop
            for stop in stops:
                stop.arrive(nextJob)
            nextJob.queue(time)
            lastArrival = time
            nextJob = Passenger(meanArrival, meanWork)
        #see if bus is at a stop
        if (busLoc)%(loopLen/numStops) == 0:
            #some passengers get off bus
            bus.unload(math.ceil(bus.getLoad()/float(numStops)))
            #all passengers who arrived prior to the bus's arrival
            #attempt to enter bus
            while stops[nextStop%numStops].length() > 0:
                try:
                    bus.enter()
                except:
                    break
                p = stops[nextStop%numStops].depart()
                totWait += time - p.queuedTime()
                totPassengers += 1
                time += p.work() #advance time, but not bus
            try:
                aveWaitTimes.append(totWait/totPassengers)
            except ZeroDivisionError:
                aveWaitTimes.append(0.0)
            #passengers might have arrived at stops while bus is loading
            while lastArrival + nextJob.interArrival() <= time:
                for stop in stops:
                    stop.arrive(nextJob)
                nextJob.queue(time)
                lastArrival += nextJob.interArrival()
                nextJob = Passenger(meanArrival, meanWork)
            nextStop += 1
    leftWaiting = 0
    for stop in stops:
        leftWaiting += stop.length()
    return aveWaitTimes, leftWaiting
```