15-112 Midterm #1a – Spring 2016
80 minutes

Name: _____

Andrew ID: _____@andrew.cmu.edu

Section: _____

- **You may not use any books, notes, or electronic devices during this exam.**
- **You may not ask questions about the exam except for language clarifications.**
- **Show your work on the exam (not scratch paper) to receive credit.**
- **If you use scratch paper, you must submit it with your andrew id on it, and we will ignore it.**
- **All code samples run without crashing. Assume any imports are already included as required.**

DO NOT WRITE IN THIS AREA		
Part 1 (CT)	10 points	
Part 2 (RC)	10 points	
Part 3 (Big-Oh)	10 points	
Part 4 (SA)	10 points	
Part 5 (FR)	20 points	
Part 6 (FR)	20 points	
Part 7 (FR)	20 points	
Part 8/bonus	5 points bonus	
Total	100 points	

1. **[10 pts] Code Tracing**

Indicate what each will print:

Statement(s):	**Prints:**

`def ct1(s, n):`
`  L = [ ]`
`  while (s != ""):`
`  for i in range(1, len(s)):`
`  if (int(s[:i]) > n):`
`  break`
`  L.append("%d:%s" % (i, s[:i]))`
`  s = s[1:-1]`
`  return [[val, len(val)] for val in L]`
`print(ct1("544367", 45))`
`# Hint: prints a 3x2 list`
	_____

`def ct2(a):`
`  a = a + [0]`
`  b = a[0:2]`
`  c = copy.deepcopy(b)`
`  a[1][0] = 4`
`  b[0] = 5`
`  b[1][1] = 6`
`  b += [7]`
`  print("a", a, "\nb", b, "\nc", c)`
`a = [[1],[2, 3]]`
`ct2(a)`
`print(a)`
	_____
	_____
	_____
	_____

2. **[10 pts] Reasoning about code**
For each function, find values of the parameters so that the function will return True. Place your answers in the box on the right of each function.

```
def rc1(n):
    i = p = 0
    while (n > 0):
        m = n%100
        assert(m > p)
        (i, n, p) = (i+1, n//100, m)
    return ((i == 4) and (i+p == 100))
```

n = _____

```
def rc2(L):
    # L is a non-rectangular 2d list
    assert ([len(r) for r in L] ==
            list(range(4,-1,-2)))
    for i in range(len(L)):
        assert(len(set(L[i])) == 2-i)
    return True
```

L = _____

3. **[10 pts] Big-Oh:**
   State the Big-Oh for each of the following functions:

Function:	**Big-Oh:**

`def bigOh1(L):`
`  # assume L is a 1d list`
`  N = len(L)`
`  M = [0] * 100`
`  for val in L:`
`  M[val % 100] += 1`
`  return set(L + sorted(M))`
	_____

`def bigOh2(L):`
`  # assume L is an NxN (square) 2d list`
`  # assume selectionSort is written as usual`
`  N = len(L)`
`  M = [ ]`
`  for i in range(len(L)):`
`  for j in range(i, len(L)):`
`  M.append(L[i][j])`
`  selectionSort(M)`
`  return 42 if (42 not in M) else M.index(42)`
	_____

`def bigOh3(L):`
`  # assume L is a pre-sorted 1d list (do not`
`  # count the cost of sorting L in your`
`  # answer)`
`  # assume binarySearch is written as usual`
`  N = len(L)`
`  M = [ ]`
`  for val in L:`
`  M.append(binarySearch(L, val))`
`  return max(M) * min(M)`
	_____

`def bigOh4(n):`
`  # assume n is an integer`
`  N = math.log(n, 2)`
`  d = dict()`
`  for c in str(n):`
`  d[c] = d.get(c, 0) + 1`
`  return d`
	_____

4. **[10 pts] Short Answer**
   Answer each of the following very briefly.

   A. State and prove the worst-case big-oh runtime of selectionSort.

   B. What specifically would go wrong if set elements were allowed to be mutable?

   C. Why is the following function not the preferred way of checking if a string is a palindrome?

```
def isPalindrome(s):
    while(len(s) > 1):
        if (s[0] != s[-1]): return False
        s = s[1:-1]
    return True
```

   D. Give an example of a common MVC violation -- that is, something that is explicitly disallowed by the rules for Model-View-Controller applications.

   E. Give an example of Python code that demonstrates short-circuit evaluation.

5. **[20 pts] Free Response: isPowerish(n) and nthPowerish(n)**

Background: we will say that a non-negative integer is "powerish" (a coined term) if it is within 2, inclusive, of some positive power of 10 (that is, 10**x for a positive integer x). The first several powerish numbers are: 8, 9, 10, 11, 12, 98, 99, 100, 101, 102, 998...

With this in mind, write the function isPowerish(n) that takes a possibly-negative int n, and returns True if n is powerish and False otherwise. Also, write the function nthPowerish(n) that takes a non-negative int n and returns the nth powerish number, where the 0^{th} powerish number is 8.

For full credit, your answers must run in a faster function family than an approach based on counting up to n, but you can do the latter for partial credit.

[this page is blank (for isPowerish and nthPowerish]

6. **[20 pts] Free Response: mostCommonCities(cityMap)**

Background: for this problem, we will say that a cityMap is a dictionary mapping a state (a lowercase string) to some cities in that state (a set of lowercase strings). For example:

```
cityMap = {
            "maine": { "madison" },
            "ohio" : { "akron", "toledo", "dayton", "madison"},
            "iowa" : { "ames", "dayton", "akron" },
            "indiana": { "akron", "madison" }
          }
```

With this in mind, write the function mostCommonCities that takes a cityMap as just described and returns a sorted list of all the cities that occur the most in that map. If we use the cityMap in the example above, this would return `["akron", "madison"]`.

[this page is blank (for mostCommonCities))

7. **[20 pts] Free Response: circleAroundCircle**

Assuming the run() function is already written for you, write init, keyPressed, mousePressed, redrawAll, and timerFired so that when the animation is first run:

A. a blue circle with a diameter of half the window's height is centered in the window; and

B. a red circle of radius 10 moves clockwise around the perimeter of the blue circle at a rate of one full revolution every 3 seconds (note that data.timerDelay may be handy in computing this); and

C. a score of 0 is shown in the bottom-left corner.

Game play proceeds as such:

D. each time the user clicks inside the red circle, its motion changes from clockwise to counter-clockwise or vice-versa, and the score increases by 1; and

E. each time the user presses the Up arrow, the radius of the red circle grows by 1; and

F. if the red circle radius gets to 20, all motion and event handling stops, and "Game Over" is displayed.

Make reasonable assumptions for anything not specified here, and in any case avoid hardcoding values (such as data.width, data.height, or data.timerDelay).

[this page is blank (for circleAroundCircle)]

[the top of this page is blank (for circleAroundCircle)]

Bonus/Optional: **[2.5 pts] What will this print?**

```
def bonusCT1(L):
    def f(L):return L[:]*len(L[:]) + [val*len(L) for val in L]
    def g(L, n): return len(L) if (len(L) >= n) else g(f(L), n)
    return g(L, 42)
print(bonusCT1(list(range(5))))
```

Bonus/Optional: **[2.5 pts] What will this print?**

```
def bonusCT2():
    a = [int(math.log(x,3)) for x in range(1,3)]
    for k in range(1234): a = [a[0] or k, a[1] and k]
    return int("".join([str(x)*x**y for x in a for y in a]))
print(bonusCT2())
```