CIS 210 Fall 2017
Midterm Two - KEY

[20 questions at 2 pts. each, plus 1coding question worth 10 pts.]
1. Given the following Python functions:

```
1 def even(n):
2     return n % 2 == 0

3 def mymedian(li):
4     '''(list of numbers) --> number

5     Returns median value of li.
6     '''
7     copyli = li[:]
8     copyli.sort()
9     if even(len(copyli)):   #li is even length
10        rmid = len(copyli) // 2
11        lmid = rmid - 1
12        median_val = (copyli[lmid] + copyli[rmid]) / 2
13    else:  #li is odd length
14        mid = len(copyli) // 2
15        median_val = copyli[mid]

16    return median_val
```

Your job is to revise function `mymedian` so that it implements the behavior of function `median_low` from the Python statistics module, described here:

```
>>> help(statistics.median_low)
Help on function median_low in module statistics:

median_low(data)
    Return the low median of numeric data.

    When the number of data points is odd, the middle value is
returned.
    When it is even, the smaller of the two middle values is
returned.

    >>> median_low([1, 3, 5])
    3
    >>> median_low([1, 3, 5, 7])
    3
```

Which lines of code will need to be changed?

a) 12, 16 **b) 5, 12** c) 5, 12, 16 d) 15, 16 e) 5, 15

2-3. Given the following Python functions to return the maximum integer from a list of integers:

```
def getmax1(li):
    '''(list of ints) -> int

    UNTESTED - Returns max int from li.
    '''
    max_so_far = 0
    for item in li[1:]:
        if item > max_so_far:
            max_so_far = item

    return max_so_far

def getmax2(li):
    '''(list of ints) -> int

    UNTESTED - Returns max int from li.
    '''
    max_so_far = li[0]
    idx = 0
    while idx < len(li):
        if li[idx] > max_so_far:
            max_so_far = idx
        idx += 1

    return max_so_far
```

2. Which statement is true?

a) `getmax1` always returns the maximum value, but `getmax2` does not.

b) `getmax2` always returns the maximum value, but `getmax1` does not.

c) Both `getmax1` and `getmax2` always return the maximum value.

d) Neither `getmax1` nor `getmax2` always returns the maximum value.

3. Which set of test cases for the get max functions is better?

a)	**b)**
`[2, 0, 1, 3]`	**`[1, 2, 3, 4]`**
`[77, 88, 99]`	**`[4, 3, 2, 1]`**
`[21, 22]`	**`[-1, 4, 1, 0]`**
`[]`	**`[]`**
`[1]`	**`[1]`**
`[10, 20, 30, 40, 50]`	**`[-6, -4, -2, -5]`**

4-5. Given the following Python code:

```
import math
import random

def isInCircle(x, y, r):
    '''(number, number, number) -> Boolean

    Returns True if point (x, y) is in
    the circle centered at (0,0) with radius r.

    >>> isInCircle(0, 0, 1)
    True
    >>> isInCircle(1, 2, 1)
    False
    '''
    d = math.sqrt(x**2 + y**2)
    isIn = d <= r

    return isIn
```

```
1 def montePi(numDarts):
2     '''
3     (integer) -> float
4
5     Uses a Monte Carlo algorithm (looping
6     numdarts times) to generate an
7     approximate value for pi, which is returned.
8
9     For example,
10    >>> montePi(100000)
11    3.13572
12    '''
13    inCircle = 0
14
15    for i in range(numDarts):
16        x = random.random()
17        y = random.random()
18
19        d = math.sqrt(x**2 + y**2)
20
21        if d <= 1:
22            inCircle += 1
23
24    approxPi = inCircle/numDarts * 4
25
26    return approxPi
```

4. Which lines of function `montePi` would need to be changed to use function `isInCircle`?

a. 13, 22 b. 3, 19, 21, 22 **c. 19, 21** d. 3, 10, 13, 22

5. Which call to `isInCircle` from `montePi` is correct?

a. `isInCircle(x**2, y**2, 1)` **b. `isInCircle(x, y, 1)`**

c. `isInCircle(x**2, y**2, d)` d. `isInCircle(x, y, d)`

6-7. Given the following Python code:

```
>>> x = 999
>>> x
999
>>> id(x)
4381366640

>>> y = x
>>> y
999
>>> id(y)
?? – checkpoint 1

>>> x = 1000
>>> x
1000

>>> id(x)
?? – checkpoint 2

>>> y
?? – checkpoint 3

>>> id(y)
?? – checkpoint 4
```

6. Replace the ??s with the correct result at checkpoint 1:

a) `4381366640` b) `4381366384` c) `10` d) `20`

7. The values at checkpoints 2 and 4 will be

a) the same – the same value as at checkpoint 1

b) the same – but not the same value as at checkpoint 1

c) different – `id(x)` will be the same as at checkpoint 1, but `id(y)` will not

d) different – `id(y)` will be the same as at checkpoint 1, but `id(x)` will not

8. Given the following Python code:

```
def dtobr(n):
    '''(int) -> str

    Convert n >= 0 to binary string.

    >>> dtob(44)
    '101100'
    '''
    print(n)                #checkpoint 1
    if n < 2:
        return str(n)
    else:
        return dtobr(n // 2) + str(n % 2)

>>> dtobr(27)
```

What value will be printed at checkpoint 1 the last time `dtobr` is executed?

a) `27` b) `13` **c) 1** d) `0` e) `'11011'`

9-11. Given the following Python code:

```
numbers = ['1', '2', '3', '4', '5', '6', '7', '8', '9', '0']
special = ['!', '@', '#', '$', '%', '&', '*']

def q9(psw):
      ''' exam function '''

      score = 0
      for ch in psw:
            if ch in numbers:
                  score += 2
            elif ch in special:
                  score += 3
            else:
                  score += 1

      if score > 8:
            return True
      else:
            return False
```

9. An appropriate type contract for function `q9` would be

a) `(str) -> Boolean` b) `(str) -> int`

c) `(int) -> Boolean` d) `(int) -> str`

10. Function `q9`

a) returns a non-`None` value; no side effect
b) returns `None` value; causes a side effect
c) returns a non-`None` value; causes a side effect
d) returns `None` value; no side effect

11. What will be the result of executing the following code:

```
>>> q9('hello!')
```

a) `6` b) `8` c) `True` **d) `False`**

12-13. What will be the result of executing the following code:

```
>>> parks = ['Crater Lake','Rainier','Olympia']
>>> park = ['Glacier']
>>> parks.append(park)
>>> parks[0]
```

12. **a) `'Crater Lake'`** b)`'Rainier'` c)`'Olympia'` d)`'Glacier'` e) error

```
>>> parks = parks.remove('Olympia')
>>> parks[0]
```

13. a) `'Crater Lake'` b)`'Rainier'` c)`'Olympia'` d)`'Glacier'` **e) error**

14. Given the following Python code:

```
def strReverseR(s):
    '''(str) -> str

    UNTESTED function to reverse s;
    return the reversed string.
    '''
    if (len(s) == 1) or (len(s) == 0):
        return s
    else:
        return s[0] + strReverseR(s[1:])
```

What is the result of executing `>>> strReverseR('hello')`

a) `'h'` **b) `'hello'`** c) `'olleh'` d) `None`

15-17. Given the following Python code:

```
def q15(s):
    '''(str) -> int

    UNTESTED midterm function.
    Returns length of longest
    consecutive string of duplicate
    Characters in s.
    '''
    if len(s) != 0:
        prev_char = s[0]
        dup_ct = 1
        high_ct = 1
    else:
        high_ct = 0

    for i in range(1, len(s)):
        if s[i] == prev_char:
            dup_ct += 1

        else:
            prev_char = s[i]

            if dup_ct > high_ct:
                high_ct = dup_ct
            dup_ct = 1

    return high_ct
```

15. What is the result of executing `>>> q15('abbbc')`?

a) `0` b) `1` **c) 3** d) `None` e) no value is returned

16. What is the result of executing `>>> q15('abccc')`?

a) `0` **b) 1** c) `3` d) `None` e) no value is returned

17. The error that results when `>>> q15('abccc')` is executed is a

a) logic error b) run time error–TypeError c) run time error-NameError d) syntax error

18-19. Given the following Python code:

```
def q18(x):
    ''' '''
    y = 2
    result = y * x
    return result

>>> y = 5
>>> q18(y)
10
>>> y
??-18
>>> x
??-19
```

Replace the ??s with the correct results:

18. a) `2` **b) `5`** c) `10` d) `NameError`

19. a) `2` b) `5` c) `10` **d) `NameError`**

20. Given the following Python code:

```
def q20(x, y):
    '''(int, int) -> None

    Midterm question.
    '''
    x = f(x, y)
    y = f(x, y)
    print(x, y)
    return None

def f(x, y):
    '''(int, int) -> int

    Midterm question.
    '''
    x = 2 * x
    y = 2 * y
    if y > x:
        return y
    else:
        return x
```

What is the result of executing `>>> q20(5, 2)`?

a) `5 2` b) `2 5` c) `4 10` d) `20 10` **e) `10 20`**

21. [10 pts.] Write function, `rainfall`, with one parameter, `rainli`, a list of numbers that record the daily rainfall in Eugene over a number of days. The data has not been cleaned, so the list may contain some negative numbers, which should be ignored.

Function `rainfall` should return the average (mean) daily rainfall for Eugene in this time period. If `rainli` is empty, or does not contain usable (non-negative) data, `rainfall` should return `-999`. For example,

```
>>> rainfall([-4, 5.0, 6, -2, -3])
5.5
```

Code should be written according to CIS 210 style guidelines, including a docstring with a type contract and at least two examples of use - one basic example and one edge/boundary. YOU MAY OMIT THE BRIEF DESCRIPTION OF THE FUNCTION.

```
def rainfall(rainli):
    '''(list of numbers) -> number

    Return average rainfall, i.e., mean
    of the non-negative numbers in rainli.

    >>> rainfall([-1, 1, 3.0, -5, 2])
    2.0
    >>> rainfall([-1])
    -999
    >>> rainfall([])
    -999
    '''
    rainsum = 0
    datactr = 0
    for rain in rainli:
        if rain >= 0:
            rainsum += rain
            datactr += 1

    if datactr > 0:
        avg_rain = rainsum / datactr
    else:
        avg_rain = -999

    return avg_rain
```

NAME____

STUDENT ID____