IT 212 – Midterm Exam

April 30, 2018 **Name _____**

The total of the raw points for this exam is 91. This total will be scaled to be out of 100.

Part A: Multiple Choice Questions. Circle the correct response for each question. Give an optional reason for each question. If you circle the correct response, the reason will not be considered. 5 points each.

1. Which of the following Python expressions converts the string value s to an integer value?
   a. `int(s)` b. `integer(s)` c. `s.int( )` d. `s.to_i( )`

2. What is output?
```
n = 124
print("{0:2x}".format(n))
```
   a. `01111110` b. `11001000` c. `7c` d. `c8`

3. What is the output?
```
s = 0x4e
t = 0x6a
print("{0:08b}".format(s & t))
```
   a. `01001010` b. `01101110` c. `4a` d.`410`

4. Given this hex dump output, what is the binary ASCII code for a lowercase `s`?
```
00000 54 68 69 73 0a 69 73 0a 61 0a 74 65 73 74 2e 0a
       T  h  i  s \n  i  s \n  a \n  t  e  s  t  . \n
```
   a. `00001010` b. `01001001` c. `01010100` d. `01110011`

5. If the method `f` is defined like this:

```
def f(the_x):
    return 2 * the_x + 3
```

   Which of these `assertEqual` statements succeeds?
   a. `self.assertEqual(f(1), 7)` b. `self.assertEqual(f(2), 8)`
   c. `self.assertEqual(f(3), 9)` d. `self.assertEqual(f(4), 13)`

6. What is the output?

```
r = [[3, 5, 1], [5, 8], [1]]
sum = 0
for row in r:
  sum += row[0]
print(sum)
```

a. 3 b. 9 c. 22 d. 341

7. How many percent signs does this script print?

```
for j in range(1, 5):
    for i in (1, 3):
        print("%")

for j in range(1, 6):
    for i in range(1, 3):
      print("%")

print("%")
```

a. 19 b. 34 c. 80 d. 370

8. If `num_quarters` is an instance variable, which of these is a proper entry in a UML diagram for this variable?
   a. – num_quarters(self) : int
   b. + num_quarters : int
   c. + num_quarters( )
   d. + num_quarters(the_num_quarters : int)

9. What is the output

```
arr = [4, 1, 3]
arr.pop( )
arr.sort( )
arr.append(2)
print(arr)
```

a. [1, 2, 3, 4] b. [1, 2, 4] c. [1, 4, 2] d. [4, 1, 3, 2]

10. The file values.txt contains Fixnum values, one per line. The following scripts are supposed to read these values into an array, sort the array, and print the sorted values. In which choice are the lines of the script in the correct order?

a.
```
array = [ ]
fin = open("values.txt", "r")
line = fin.readline( )
while line != "":
    value = int(line.strip( ))
    array.append(value)
    line = fin.readline( )
for value in array:
         print(value, end=" ")
array.sort( )
```

b.
```
fin = open("values.txt", "r")
line = fin.readline( )
while line != "":
    array = [ ]
    value = int(line.strip( ))
    array.append(value)
    line = fin.readline( )
array.sort( )
for value in array:
    print(value, end=" ")
```

c.
```
array = [ ]
array.sort( )
fin = open("values.txt", "r")
line = fin.readline( )
while line != "":
    value = int(line.strip( ))
    array.append(value)
    line = fin.readline( )
for value in array:
         print(value, end=" ")
```

Choice d is on the next page.

d.
```
array = [ ]
fin = open("values.txt", "r")
line = fin.readline( )
while line != "":
    value = int(line.strip( ))
    array.append(value)
    line = fin.readline( )
array.sort( )
for value in array:
    print(value, end=" ")
```

Part B. Predict the Output. What is the output of this script? Justify your answer. 8 points.

```
# --- Source code file: pair.py
class Pair:

    def __init__(self, the_x, the_y):
        self.x = the_x
        self.y = the_y

    def __str__(self):
        return f"({self.x},{self.y})"

    def multiply(self, the_factor):
        self.x *= the_factor
            self.y *= the_factor
```

```
# --- Source code file: test1.py
from pair import Pair

p = Pair(3, 5)
p.multiply(2)
print(p)
```

Variable trace table:
for `Pair` object `p`

`self.x`	`self.y`

Output:

Part B. Correct Errors in Python Class and Test File. Correct the errors in source code for the MovieReview class on this page. There are about 15 total errors on this page. Mark your corrections directly on the source code; do not recopy. Adding, deleting, or correcting a pair of parentheses (( )), brackets ([ ]), braces ({ }), single quotes (' ') , double quotes (" "), or triple quotes (''' ''') only counts as one error. 15 points.

```
# --- Source code file: moviereview.py

Class movie_review:

    def __init_(self, the_title, director):
        self.title = the_title
        self.director = the_director
        self.comment = ""
        # The rating can be between 0.0 and 5.0, inclusive.
        # A rating of -1 means uninitialized.
        self.rating = -1

    def set_rating(the_rating):
        if 0.0 < = self.rating and self.rating <= 5.0
            self.rating = the_rating
        else:
            self.rating == -1

    def __str__(self)
        return '''Title: {0:s}
Director: [1:s]
Comment: {2:s}
Rating: {4:.1f}".format(self.title, self.director, \
                        self.comment, self.rating)

# Source code file: test1.rb
# Script to test MovieReview class.

imports moviereview
review = MovieReview("La La Land", Damien Chazelle)
review.comment = "An effervescent sample of pure entertainment.
review.set_rating(4.5)
print(review)
print(review.title  review.director)
print(review.comment)
print(review.rating)
review.set_rating 6.0
print(review.rating)
```

Part C. Predict the Output. Predict the output of this class and test file. Show the output on this page. 10 points.

```
# === Source code file Television.py ===========

class Television:

    def __init__(self, the_brand):
        self.brand = the_brand
        self.channel = 2
        self.volume = 0

    def __str__(self):
        return f"{self.brand}, {self.channel}, {self.volume}"

    def increase_volume(self):
        self.volume += 10
        if self.volume >= 100:
            self.volume = 100

    def decrease_volume(self):
        self.volume -= 10
        if self.volume <= 0:
            self.volume = 0
```

```
# === Traditional Test File test1.py ===========

from television import Television

tv = Television("Soni")
print(tv)
print(tv.brand, tv.channel)
tv.channel = 11
print(tv.channel, tv.volume)
tv.decrease_volume( )
print(tv.volume)
tv.increase_volume( )
tv.increase_volume( )
print(tv.volume)
```

Variable trace table
for Television object tv:

self.brand	self.channel	self.volume

Part D. Write Unit Test File. Convert the traditional test file `test1.py` in Part C to the unit test file `test2.py`. 8 points.

```
from television import Television
import unittest

class MyUnitTestClass(unittest.TestCase):

    def test_1(self):

        # To write the unit test, convert the print statements to assert
        # equal statements of this form:
        # self.assertEqual(computed, expected)
        # Leave the other statements as is.

        tv = Television("Soni")

if __name__ == '__main__':
    unittest.main( )
```