IT 211 – Intro to Applied Programming

June 4, 2018 Name _____

Part A. Multiple Choice. Circle the letter of the best correct answer. Don't circle more than one answer. Give an optional reason for each question for partial credit. If your question is correct, the reason will not be considered. If you think that none of the answers are correct, indicate your correct answer with a reason.
5 points each.

1. What is the Python concatenation operator?
   a. & b. || c. + d. #

2. Which expression converts the string value `s` into an integer?
   a. `n = int(s)` b. `n = s.int( )` c. `n = s.to_int( )` d. `b = s.to_integer( )`

3. Which Python method is used to read a line of text entered by the user at the keyboard?
   a. `input` b. `read` c. `readline` d. `scan`

4. What is the output?
```
a = ord("C")
b = 10
print(a % b)
```
   a. 2 b. 3 c. 5 d. 7

5. Which script prints each word in the list `a` defined by
```
a = ["broccoli", "carrot", "cabbage", "turnip", "zuccini"]
```
   a.
```
for each veg in a:
    print(veg)
```
   c.
```
for i in range(0, len(a)):
    print(a[i])
```
   c.
```
for i in range(1, 5):
    print(a[i])
```
   d.
```
if veg in a:
    print(veg)
```

6. What shape does this script draw?
```
for i in range(0, 20):
    for j in range(0, 10):
        print("*", end="")
```
   a. Horizontal line b. Rectangle c. Triangle d. Vertical line

7. If the input file `test.txt` contains these lines

```
This
is
a
test.
```

What is the output of this script?

```
f = open("test.txt", "r")
for i in range(0, 4):
    print(f.readline( ))
```

a. `Thisisatest.`

b. `This is a test`

c.
```
This
is
a
test.
```

d.
```
This

is

a

test.
```

8. What is the output?

```
import itertools
print(len(list(itertools.permutations(list("abc")))))
```

a. `3`

b. `[('a', 'b'), ('a', 'c'), ('b', 'c')]`

c. `6`

d. `['abc', 'acb', 'bac', 'bca', 'cab', 'cba']`

9. The standalone method `all_vowels` returns `true` if `input string` consists of all vowels and `false` otherwise. Which choice is the correct definition of the `all_vowels` method?

a.
```
def all_vowels(input_string):
    s = input_string.upper( )
    for c in s:
        if c != "A" and c != "E" and c != "I" and \
           c != "O" and c != "U":
            return False
        return True
```

b.
```
def all_vowels(input_string):
    s = input_string.upper( )
    for c in s:
        if c != "A" and c != "E" and c != "I" and \
           c != "O" and c != "U":
            return False
    return True
```

Choices c and d are on the next page.

c.
```
def all_vowels(input_string):
    s = input_string.upper( )
    for c in s:
        if c != "A" and c != "E" and c != "I" and \
           c != "O" and c != "U":
        return False
        return True
```

d.
```
def all_vowels(input_string):
    for c in s:
        s = input_string.upper( )
        if c != "A" and c != "E" and c != "I" and \
           c != "O" and c != "U":
            return False
        return True
```

10. This script draws a 100 by 100 image of gray pixels:

```
fout = open("image.ppm", "w")
fout.write("P3 100 100 255\n")
for x in range(0, 100):
    for y in range(0, 100):
        draw_gray_pixel(fout, 128)
    fout.write("\n")
fout.close( )
```

How should the `draw_gray_pixel` method be defined to do this?

a.
```
def draw_gray_pixel(fout, v):
    fout.write("g g g  ")
```

b.
```
def draw_ gray_pixel(fout, v):
    fout.write("gray ")
```

c.
```
def draw_ gray_pixel(fout, v):
    fout.write(f"{v}  ")
```

d.
```
def draw_ gray_pixel(fout, v):
    fout.write(f"{v} {v} {v}  ")
```

11. The lists `ids` and `names` are defined as

```
ids = [22222, 33333, 44444, 55555]
names = ["Alice", "Taylor", "Chloe", "Scott"]
```

The `make_dictionary` method returns a dictionary with keys in a list such as `ids` and values in a list such as `names`. Which of these choices has its statements in the correct order?

a.
```
def make_dictionary(keys, values):
    dictionary = { }
    for i in range(0, len(keys)):
        dictionary[keys[i]] = values[i]
    return dictionary
```

b.
```
def make_dictionary(keys, values):
    dictionary = { }
    for i in range(0, len(keys)):
        return dictionary
    dictionary[keys[i]] = values[i]
```

c.
```
def make_dictionary(keys, values):
    for i in range(0, len(keys)):
        dictionary = { }
        dictionary[keys[i]] = values[i]
    return dictionary
```

d.
```
def make_dictionary(keys, values):
    for i in range(0, len(keys)):
        dictionary[keys[i]] = values[i]
    return dictionary
    dictionary = { }
```

12. Which statement must be included before the statement

```
dictionary_array = loads(json_string)
```

can be executed?

a. `from json import loads`
b. `from loads import json`
c. `import json`
d. `import json.loads`

13. For the `Pair` class defined in Problem 2 of Part C on Page 7, which of these methods is the constructor?

a. `__init__`
b. `Pair`
c. `str`
d. `__str__`

14. For the `Pair` class defined in Problem 2 of Part C on Page 7, which of these methods is a dunder method?
    b. `__init__` b. `Pair` c. `multiply` d. `str`

15. For the `Pair` class defined in Problem 2 of Part C on Page 7, which of these variables is an instance variable?
    a. `output` b. `self` c. `multiply` d. `x`

Part B: Find the errors in Python Source Code. There are about 15 total errors in the Python source code files Script 1 and Script 2. Correct the errors directly on pages 5 and 6. Do not recopy. (8 points penalty for recopying.) Correcting a pair of `( )`, `[ ]`, `{ }`, `""`, or `''` only counts as one error. 15 points.

```
# Script 1
# Create JSON file from pets.txt
# The beginning of pets.txt is shown at the top of Page 6.

import json

lst = [ ]
fin = open(pets.txt, "w")

line = readline( )
while line = "":
    fields = line.split(",")
    name = fields[0].strip( )
    animal_type = fields[1].strip( )
    age = int(fields[2].split(","))
    d = {"name": name, "animal_type" : animal_type, "age"; age}
    lst.append(d)
    line = fin.readline( )

fin.close( )

fout = open("pets.json", "w")
json_string = json.dumps(lst)
fout.write(json_string)
```

```
# The beginning of pets.txt input file:
name/animal_type/age
Milo/dog/8
Coco/cat/3
Oscar/cat/2
Athena/dog/5
Izzi/dog/2
Shadow/cat/6
...

# Script 2
# For a desired name, print animal_type and age.
# Also print average age.

import json

fin = open("pets.json", "r")
json_string = fin.read( )
lst = json.loads(json_string)
fin.close( )

desired-name = readline("Enter pet name: ")

for d in lst:
    if desired_name == d["name"]:
        print("Animal Type: {d['animal_type']}; Age: {d["age"]}")



for d in lst:
    total = 0.0
    total += d["age"]
    count + = 1

if count > 0:
    print(f"Average age of pets is {round(count / total, 2}")
else:
    print("No pets in JSON file.")
```

Part C: Predict the Output. Draw the variable trace and predict the output for each problem.

Remember that the statement

`x, y = y, x`

swaps the values of `x` and `y`. 10 points each.

```
# Problem 1.
a = [23, 75, 19]
if a[0] > a[1]:
    a[0], a[1] = a[1], a[0]
if a[0] > a[2]:
    a[0], a[2] = a[2], a[0]
if a[1] > a[2]:
    a[1], a[2] = a[2], a[1]
print(a)
```

a[0]	a[1]	a[2]

Output:

```
# Problem 2.

# 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):
        output = f"({self.x},{self.y})"
        return output

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

Pair object p		Pair object q	
self.x	self.y	self.x	self.y

Output:

```
# Source code file test.py
from pair import Pair
p = Pair(3, 5)
q = Pair(4, 7)
q.multiply(3)
print(str(q))
print(str(p))
```