CSCI 145 Quiz 7 practice questions on sets, references, and files

1. Suppose `a = {3, 6, 9, 12}` and `b = {4, 8, 12}`. Write the value of each expression. A set may list its values in any order.

`a & b`	_____	`a | b`	_____
`b - a`	_____	`len(a - b)`	_____
`len(set([7, 7, 7]))`	_____	`8 in b`	_____

2. Circle one answer for each part.

a) Which one makes an empty set?

A. `{}`
B. `[]`
C. `set()`
D. `{,}`

b) What happens when this runs?

```
tags = {"red", "blue"}
print(tags[0])
```

A. It prints `red`
B. It prints `blue`
C. A `KeyError`
D. A `TypeError`

c) What does this print?

```
pets = {"cat", "dog"}
pets.discard("fish")
print(len(pets))
```

A. 3
B. 2
C. 1
D. A `KeyError`

3. Consider this code:

```
week = [[2, 5], [8]]
saved = week.copy()
saved[0].append(3)
week[1] = [0]
print(week)
print(saved)
```

a) Write the exact output.

b) Suppose the code starts with `import copy`. Which replacement for line 2 makes a copy that shares no rows with `week`?

A. `saved = copy.deepcopy(week)`
B. `saved = week`
C. `saved = week[:]`
D. `saved = list(week)`

4. Suppose this code runs. Write the value of each expression.

```
nums = [7, 1]
same = nums
twin = nums.copy()
same.append(5)
```

`nums == same`	_____	`nums == twin`	_____
`nums is same`	_____	`twin is same`	_____
`len(twin)`	_____	`len(nums)`	_____

5. Consider this code:

```
def tag(seen, word):
    seen.add(word.upper())
    return len(seen)
words = {"HI"}
w = "yo"
count = tag(words, w)
print(count, w)
```

a) Write the exact output.

b) What is `words` at the end?

6. Write CSV or JSON: which file format fits each set of data better?

The date, high temperature, and low temperature for each day of a year _____

A recipe's name, its number of servings, and its list of ingredients _____

7. The file `fruit.txt` holds these three lines. Write the exact output of this code.

fruit.txt

```
fig
pear
mango
```

```
with open("fruit.txt") as f:
    lines = f.readlines()
print(len(lines))
for line in lines:
    print(line.strip().upper())
```

Output

8. Suppose `line = "Otter,7,12\n"` and `parts = line.strip().split(",")`. Write the value of each expression.

`parts` _____ `len(parts)` _____

`parts[1] + parts[2]` _____ `int(parts[2]) - 2` _____

9. Write the lines each file holds after the code runs, one per line.

a)

```
with open("a.txt", "w") as f:
    f.write("up\n")
    f.write(str(3) + "\n")
    f.write("down\n")
```

b) It runs twice; `b.txt` is new.

```
with open("b.txt", "a") as f:
    f.write("go\n")
```

c) The same, but with `"w"`.

```
with open("b.txt", "w") as f:
    f.write("go\n")
```

10. Write the exact output of each program.

a) `trees.csv`

```
tree,height_m
oak,21
birch,15
```

```
import csv
with open("trees.csv") as f:
    rows = list(csv.reader(f))
print(rows[2], len(rows))
print(rows[1])
```

b) `pet.json`

```
{"name": "Rex", "age": 4,
 "vet": true,
 "toys": ["ball", "rope"]}
```

```
import json
with open("pet.json") as f:
    pet = json.load(f)
print(pet["toys"][1], pet["age"] + 1)
print(pet["vet"])
```

11. Write a function `total_donations(filename)`. The file has a header line, then one `name,amount` line per donation (a whole number). A name can appear more than once. It returns a dictionary that maps each name to that person's total. If the file has no donations, it returns an empty dictionary. Do not include a docstring. For example:

`gifts.csv`

```
name,amount
Ava,20
Leo,5
Ava,15
```

```
>>> total_donations("gifts.csv")
{'Ava': 35, 'Leo': 5}
```