(/)

Evaluation quiz correction

Evaluation Quiz: Evaluation #4

Date: 2023-04-28

Status: Done

Duration: 21 minutes

Score: 94.44%

 # "I don't know": 0

 # Success: 17

 # Fail: 1

Responses

0. What is the `unistd` symbolic constant for the standard error?

Score: 1.0

- [ ] STDIN_FILENO
- [ ] STDOUT_FILENO
- [x] **STDERR_FILENO**
- [ ] I don't know

1. What do these lines print?

```
class User:
    id = 1

u = User()
User.id = 98
print(u.id)
```

Score: 1.0

- [ ] None
- [ ] 1
- [ ] 89
- [x] **98**
- [ ] I don't know

2. In this following code, what is `__password` ?

```
class User:
    id = 89
    name = "no name"
    __password = None

    def __init__(self, new_name=None):
        self.is_new = True
        if new_name is not None:
            self.name = new_name
```

Score: 1.0

- [ ] A public class attribute
- [ ] A public instance attribute
- [ ] A protected class attribute
- [ ] A protected instance attribute
- [x] **A private class attribute**
- [ ] A private instance attribute
- [ ] I don't know

3. What is `__repr__` ?

Score: 1.0

- [ ] Instance method that prints an "official" string representation of an instance
- [x] **Instance method that returns an "official" string representation of an instance**
- [ ] Instance method that returns the dictionary representation of an instance
- [ ] I don't know

4. Which of the following sorting algorithms has best case time complexity of `O(nlog(n))`?

Score: 1.0

- [x] **Quick Sort**
- [ ] Bubble Sort
- [ ] Insertion Sort
- [ ] Selection Sort
- [ ] I don't know

5. In a singly linked list, what are possible directions to traverse it?

(select all possible answers)

Score: 1.0

- [x] **Forward**
- [ ] Backward
- [ ] I don't know

6. Is this a standardized way to comment a function in Python?

```
/* Addition function */
def add(a, b):
    return a + b
```

Score: 1.0

- [x] **No**
- [ ] Yes
- [ ] I don't know

(/)

7. What data structure is the foundation of a Python dictionary or set?

Score: 1.0

- [x] **Hash Table**
- [ ] Stack
- [ ] Queue
- [ ] Binary Tree
- [ ] I don't know

8. Based on this code, what should all the test cases be?

(select all possible answers)

```
def uniq(list):
   """ Returns unique values of a list """
   u_list = []
   for item in list:
       if item not in u_list:
           u_list.append(item)
   return u_list
```

Score: 0.0

- [x] **empty list**
- [x] **list with one element (any type)**
- [x] **list with 2 different elements (same type)**
- [x] **list with the same element twice (same type)**
- [x] **list with more than 2 times the same element (same type)**
- [ ] **list with multiple types (integer, string, etc...)**
- [x] **not a list argument (ex: passing a dictionary to the method)**
- [ ] I don't know

9. What do these lines print?

```
>>> class User:
>>>     id = 89
>>>     name = "no name"
>>>     __password = None
>>>
>>>     def __init__(self, new_name=None):
>>>         self.is_new = True
>>>         if new_name is not None:
>>>             self.name = new_name
>>>
>>> u = User()
>>> u.name
```

Score: 1.0

- [ ] name
- [ ] None
- [ ] 'John'
- [x] **'no name'**
- [ ] I don't know

10. What do these lines print?

```
class Base():
    """ My base class """

    __nb_instances = 0

    def __init__(self):
        Base.__nb_instances += 1
        self.id = Base.__nb_instances

class User(Base):
    """ My User class """

    def __init__(self):
        super().__init__()
        self.id += 99

u = User()
print(u.id)
```

Score: 1.0

- [ ] 99

- [x] **100**
- [ ] 1
- [ ] I don't know

11. Is this module correctly documented?

```
#!/usr/bin/python3
"""
    My calculation module
"""
import sys
...
```

Score: 1.0

- [x] **Yes**
- [ ] No
- [ ] I don't know

12. Given this code:

```
struct point {
   int x;
   int y;
};
struct point my_point = { 3, 7 };
struct point *p = &my_point;
```

To set the member y of my variable my_point to 98, I can do (select all valid answers):

Score: 1.0

- [x] **`my_point.y = 98`**
- [ ] `my_point->y = 98`
- [ ] `p.y = 98`
- [x] **`(*p).y = 98`**
- [x] **`p->y = 98`**
- [ ] I don't know

13. What is the size of the `int` data type on a 64-bit machine?

Score: 1.0

- [ ] 1 byte
- [ ] 2 bytes
- [x] **4 bytes**
- [ ] 8 bytes
- [ ] I don't know

14. What do these lines print?

```
class Base():
    """ My base class """

    __nb_instances = 0

    def __init__(self):
        Base.__nb_instances += 1
        self.id = Base.__nb_instances

class User(Base):
    """ My User class """

    def __init__(self):
        super().__init__()
        self.id = 89

u = User()
print(u.id)
```

Score: 1.0

- [x] **89**
- [ ] 90
- [ ] 1
- [ ] I don't know

15. What is `__doc__` ?

Score: 1.0

- [x] **The string documentation of an object (based on docstring) (/)**
- [ ] Prints the documentation of an object
- [ ] Creates man file
- [ ] I don't know

16. What does the following Bash script do?

```
#!/usr/bin/env bash

var="Tech"
if [ -e "$var" ]
then
    if [ -f "$var" ]
    then
        echo "Betty"
    elif [ -d "$var" ]
    then
        echo "School"
    fi
else
    echo "$var doesn't exist"
fi
```

Score: 1.0

- [x] **Checks if `Tech` exists, otherwise prints "Tech doesn't exist". If it exists and it's a file, print "Betty", otherwise if it's a directory, print "School".**
- [ ] Checks if a file (inputted by the user) exists, otherwise prints "File doesn't exist". If it exists and it's a file, print "Betty", otherwise if it's a directory, print "School".
- [ ] Checks if `Tech` exists and prints "Tech exists"
- [ ] I don't know

17. Bubble Sort is a _____.

Score: 1.0

- [x] **simple comparison sorting algorithm**
- [ ] complex comparison sorting algorithm
- [ ] simple non-comparison searching algorithm
- [ ] simple non-comparison sorting algorithm
- [ ] I don't know

(/)