🏠 / Dashboard / My courses / Programming in Python - 2. Python Programming / 2. Object Oriented Programming / 2.3 Quiz

Started on	Thursday, 4 April 2024, 12:59 PM
State	Finished
Completed on	Thursday, 4 April 2024, 1:07 PM
Time taken	8 mins 4 secs
Marks	4.00/5.00
Grade	**8.00** out of 10.00 (**80**%)

Question **1**

Complete

Mark 1.00 out of 1.00

Consider the following code and answer the given question.

```
class Person:
    def __init__(self, title="default_title"):
        self._name = None
        self.title = title

    def talk(self):
        print("I'm a person")

    def walk(self):
        print("I'm walking")

class Employee(Person):
    def __init__(self, id, title):
        super().__init__(title)
        self.id = id

    def talk(self):
        print("I'm an employee")

class SportsPerson(Person):
    def __init__(self, id, title):
        super().__init__(title)
        self.id = id

    def talk(self):
        print("I'm a sportsperson")

person1 = Person("p1")
person2 = Employee(1, "p2")
person3 = SportsPerson(2, "p3")

def check_talk(obj):
    obj.talk()
```

What will be the output of the following code?

check_talk(person3)

○ a. I'm a person

○ b. I'm an employee

● c. I'm a sportsperson

○ d. None of the above

Question 2

Complete

Mark 1.00 out of 1.00

What is the output of the following code segments?

```
len("Programiz")
len ([1,2,3,4])
```

● a. 9
4

○ b. 13

○ c. prog
4

○ d. None of the above

Question **3**

Complete

Mark 1.00 out of 1.00

Consider the following code and answer the given question.

```
class Person:
    def __init__(self, title="default_title"):
        self._name = None
        self.title = title

    def talk(self):
        print("I'm a person")

    def walk(self):
        print("I'm walking")

class Employee(Person):
    def __init__(self, id, title):
        super().__init__(title)
        self.id = id

    def talk(self):
        print("I'm an employee")

class SportsPerson(Person):
    def __init__(self, id, title):
        super().__init__(title)
        self.id = id

    def talk(self):
        print("I'm a sportsperson")

person1 = Person("p1")
person2 = Employee(1, "p2")
person3 = SportsPerson(2, "p3")

def check_talk(obj):
    obj.talk()
```

What will be the output of the following code?

person2.talk()

○ a. I'm a person

● b. I'm an employee

○ c. I'm a person
I'm an employee

○ d. None of the above

Question **4**

Complete

Mark 1.00 out of 1.00

Consider the following code and answer the given question.

```
class Person:
    def __init__(self, title="default_title"):
        self._name = None
        self.title = title

    def talk(self):
        print("I'm a person")

    def walk(self):
        print("I'm walking")

class Employee(Person):
    def __init__(self, id, title):
        super().__init__(title)
        self.id = id

    def talk(self):
        print("I'm an employee")

class SportsPerson(Person):
    def __init__(self, id, title):
        super().__init__(title)
        self.id = id

    def talk(self):
        print("I'm a sportsperson")

person1 = Person("p1")
person2 = Employee(1, "p2")
person3 = SportsPerson(2, "p3")

def check_talk(obj):
    obj.talk()
```

How to convert the talk() method in Person class to an abstract method?

○ a. Person class should extend ABC class.

○ b. talk() should have @abstractmethod decorator.

○ c. Either Person class should extend ABC class or talk() should have @abstractmethod decorator.

● d. None of the above

Question **5**

Complete

Mark 0.00 out of 1.00

Consider the following code and answer the given question.

```
class Person:
    def __init__(self, title="default_title"):
        self._name = None
        self.title = title

    def talk(self):
        print("I'm a person")

    def walk(self):
        print("I'm walking")

class Employee(Person):
    def __init__(self, id, title):
        super().__init__(title)
        self.id = id

    def talk(self):
        print("I'm an employee")

class SportsPerson(Person):
    def __init__(self, id, title):
        super().__init__(title)
        self.id = id

    def talk(self):
        print("I'm a sportsperson")

person1 = Person("p1")
person2 = Employee(1, "p2")
person3 = SportsPerson(2, "p3")

def check_talk(obj):
    obj.talk()
```

Which of these statements is true?

○ a. Person class is an abstract class.

○ b. The given code exhibits method overloading.

○ c. The constructor of the SportsPerson can be removed, and the code will work the same.

● d. None of the above

◄ 2.3 OOP Principles II

Jump to...

3.1 Database Basics ►

GET IN TOUCH

University of Moratuwa
Centre for Open & Distance Learning
CODL

☎ 011 308 2787/8

☎ 011 265 0301 ext. 3850,3851

✉ open@uom.lk

University Website

CODL Website

Data retention summary

mora_logo.png Sponsored By logo.png