Started on	Tuesday, 2 April 2024, 10:30 PM
State	Finished
Completed on	Tuesday, 2 April 2024, 10:38 PM
Time taken	7 mins 43 secs
Marks	2.00/5.00
Grade	**4.00** out of 10.00 (**40**%)

Question **1**

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, words):
        print("talk 1")
    def talk(self, words):
        print("talk 2")

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

Which of the following statements is false?

a. The '_name' attribute will have different values in different Person objects.

b. The 'title' attribute will always have the same value in different Person objects.

c. Accessing the '_name' attribute from outside the class is not a good programming practice in Python.

d. A getter and setter have to be defined for the '_name' attribute.

Question 2

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, words):
        print("talk 1")
    def talk(self, words):
        print("talk 2")

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

What will be the output of the following code?

person1 = Person ()

person1.talk("Hello")

a. talk 1

b. talk 2

c. talk 1
talk 2

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, words):
        print("talk 1")
    def talk(self, words):
        print("talk 2")

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

What will be the output of the following code?

person2 = Person("Professor")

print (person2.title)

- a. default_title
- b. Professor
- c. Any of the above
- d. None of the above

Question 4

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, words):
        print("talk 1")
    def talk(self, words):
        print("talk 2")

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

To achieve method overloading in the given code

a. Add a talk() method to the Employee class with the exact same parameters as the talk() method in the Person class.

b. Add a talk() method to the Employee class with parameters different to the talk() method in the Person class.

c. Have more than one talk() methods in Person class, each taking a different set of parameters.

d. Have a single talk() method in Person class, with parameters having default values.

Question 5

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, words):
        print("talk 1")
    def talk(self, words):
        print("talk 2")

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

What should be given to *xx* in *Employee* class constructor?

Answer: title

◀ 2.2 OOP Principles I

Jump to...

2.3 OOP Principles II ►

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