

1. Which of the following is considered by the text to be the most important consideration when designing a class?

1. Each class should represent an appropriate mathematical concept.
2. Each class should represent a single concept or object from the problem domain.
3. Each class should represent no more than three specific concepts.
4. Each class should represent multiple concepts only if they are closely related.

Title

Which of the following is considered by the text to be the most important consideration when designing a class?

type

mc

Section

9.1 Object-Oriented Programming

id

testbank-py-1-ch09-01

2. Which of the following questions should you ask yourself in order to determine if you have named your class properly?

1. Does the class name contain 8 or fewer characters?
2. Is the class name a verb?
3. Can I visualize an object of the class?
4. Does the class name describe the tasks that this class will accomplish?

Title

How do I ensure I have named a class properly?

type

mc

Section

9.1 Object-Oriented Programming

id

testbank-py-1-ch09-02

3. Which of the following is NOT a true statement regarding object-oriented programming?

1. Object oriented programming views the program as a list of tasks to perform.
2. Object oriented programs usually contain different types of objects, each corresponding to a particular kind of complex data, real-world object or concept.
3. Object oriented programming is a style where tasks are solved by collaborating objects.
4. Object oriented programs are organized around "objects" rather than "actions" and "data" rather than "logic".

Title

Which of the following is NOT a true statement regarding object-oriented programming?

type

mc

Section

9.1 Object-Oriented Programming

id

testbank-py-2-ch09-03

from

testbank-py-1-ch09-03

4. Which statement about classes is true?

1. A class contains data and methods that act upon that data.
2. A class decomposes tasks into functions.
3. A class describes a set of objects with different behaviors.
4. A class describes a set of objects that all have the same behavior.

Title

Which statement about classes is true?

type

mc

Section

9.1 Object-Oriented Programming

id

testbank-py-2-ch09-04

from

testbank-py-1-ch09-04

5. Which statement is correct about the public interface for a class?

1. The public interface for a class is a description of its class variables, and the types of information they are supposed to store.
2. The public interface for a class is the set of all methods provided by a class, together with a description of their behavior.
3. The public interface for a class is the set of all its methods, along with the code in the method bodies.
4. The public interface for a class is the name of the class without any additional information.

Title

Which statement is correct about the public interface for a class?

type

mc

Section

9.1 Object-Oriented Programming

id

testbank-py-2-ch09-05

from

testbank-py-1-ch09-05

6. Which statement is NOT a description of encapsulation?

1. The act of hiding the implementation details
2. A collection of methods through which the objects of the class can be manipulated
3. Enables changes in the implementation without affecting users of a class
4. Mechanism for restricting access to some of the object's components

Title

Which statement best describes the public interface for a class?

type

mc

Section

9.1 Object-Oriented Programming

id

testbank-py-1-ch09-06

7. Which of the following statements is **not** legal?

1. ["Hello", "World"].lower()
2. "Hello, World".lower()
3. ["Hello", "World"].pop()
4. ["Hello, World"].pop()

Title

Which statement is not legal?

type

mc

Section

9.1 Object Oriented Programming

id

testbank-py-1-ch09-07

8. What is the purpose of an object's instance variables?

1. Store the data required for executing its methods
2. Store the data for all objects created by the same class
3. To provide access to the data of an object
4. To create variables with public visibility

Title

What is the purpose of an object's instance variables?

type

mc

Section

9.2 Implementing a Simple Class

id

testbank-py-1-ch09-08

9. Naming conventions for Python dictate that instance variables should start with an underscore to represent:

1. public visibility
2. private visibility
3. encapsulation
4. public interface

Title

Why do instance variables start with an underscore?

type

mc

Section

9.2 Implementing a Simple Class

id

testbank-py-1-ch09-09

10. The following two objects are created from the `Counter` class: `studentCounter`, and `teacherCounter` to represent the total number of students and the total number of teachers respectively. If the `Counter` class contains an instance variable, `_num`, that is initialized to zero and increases every time the user executes the `add` method, what is stored in each object's instance variable after the following code snippet executes?

```
studentCounter.add()
teacherCounter.add()
studentCounter.add()
```

1. `studentCounter : 3, teacherCounter : 3`
2. `studentCounter : 2, teacherCounter : 1`
3. `studentCounter : 1, teacherCounter : 2`
4. `studentCounter : 1, teacherCounter : 1`

Title

What are the values of two instance variables after the given code snippet is executed?

type

mc

Section

9.2 Implementing a Simple Class

id

testbank-py-2-ch09-10

from

testbank-py-1-ch09-10

11. What is the purpose of a method?

1. Enables changes in the implementation without affecting users of a class
2. To access the instance variables of the object on which it acts
3. Store the data for all objects created by the same class
4. Provide comments about the program implementation

Title

What is the purpose of a method?

type

mc

Section

9.2 Implementing a Simple Class

id

testbank-py-1-ch09-11

12. Which of the following is NOT a difference between methods and functions?

1. A method is defined within a class. A function is defined outside of a class.
2. A function is defined within a class. A method is defined outside of a class.
3. The first parameter variable for a method is `self`. Functions do not have a `self` parameter variable.
4. A method can access the instance variables of an object. A function cannot.

Title

Which of the following is NOT a difference between methods and functions?

type

mc

Section

9.2 Implementing a Simple Class

id

testbank-py-2-ch09-12

from

testbank-py-1-ch09-12

13. What is the purpose of the `self` parameter?
1. Enables changes in the implementation without affecting users of a class
 2. To create variables with public visibility
 3. Store the data for all objects created by the same class
 4. Refers to the object on which the method was invoked

Title

What is the purpose of the `self` parameter?

type

mc

Section

9.2 Implementing a Simple Class

id

testbank-py-1-ch09-13

14. According to the textbook, what is the best practice for updating instance variables?
1. Directly access the variables
 2. Restrict access to instance variables, only allow updates through methods
 3. Restrict access to instance variables, only allow updates through functions
 4. Provide both direct and indirect access to instance variables

Title

According to the textbook, what is the best practice for updating instance variables?

type

mc

Section

9.2 Implementing a Simple Class

id

testbank-py-1-ch09-14

15. Suppose you have a class `ShoppingList`, with instance variables `_quantity`, `_cost`, and `_itemName`. How should you access these variables when writing code that is not contained within the `ShoppingList` class?
1. Directly access the instance variables by using the object and the instance variables' names.
 2. Use methods provided by the `ShoppingList` class.
 3. It is not possible to access the instance variables outside of the `ShoppingList` class.
 4. Use the `self` parameter followed by the instance variables' names.

Title

How should you access instance variables in your program?

type

mc

Section

9.2 Implementing a Simple Class

id

testbank-py-1-ch09-15

16. Assume a class exists named `Fruit`. Which of the follow statements constructs an object of the `Fruit` class?

```

1. def Fruit() :
2. class Fruit() :
3. x = Fruit()
4. x = Fruit.create()

```

Title

Which statement creates an object?

type

mc

Section

9.2 Implementing a Class

id

testbank-py-1-ch09-16

17. Which of the following statements is used to begin the implementation of a new class named `Fruit`?

```

1. class Fruit :
2. def Fruit :
3. object Fruit :
4. x = Fruit()

```

Title

Which statement is used to begin the implementation of a new class?

type

mc

Section

9.2 Implementing a Simple Class

id

testbank-py-1-ch09-17

18. What is the name of the instance variable in the following code segment?

```

class Fruit :
    def getColor(self) :
        return self._color

1. _color
2. Fruit
3. getColor
4. self

```

Title

What is the name of the instance variable in the following code segment?

type

mc

Section

9.2 Implementing a Simple Class

id

testbank-py-1-ch09-18

19. What is the name of the method in the following code segment?

```

class Fruit :
    def getColor(self) :
        return self._color

1. _color
2. Fruit
3. getColor
4. self

```

Title

What is the name of the method in the following code segment?

type

mc

Section

9.2 Implementing a Simple Class

id

testbank-py-1-ch09-19

20. How many methods are there in the following class?

```
class Person :
    def getName(self) :
        return self._name

    def getSalary(self) :
        return self._salary

    def giveRaise(self, howMuch) :
        self._salary = self._salary + howMuch
```

1. 0
2. 1
3. 2
4. 3

Title

How many methods are there in a class?

type

mc

Section

9.2 Implementing a Simple Class

id

testbank-py-1-ch09-20

21. Consider the following class:

```
class Counter :
    def getValue(self) :
        return self._value

    def click(self) :
        self._value = self._value + 1

    def unClick(self) :
        self._value = self._value - 1

    def reset(self) :
        self._value = 0
```

Which method creates a new instance variable?

1. click
2. getValue
3. reset
4. unClick

Title

Which method creates a new instance variable?

type

mc

Section

9.2 Implementing a Simple Class

id

testbank-py-1-ch09-21

22. Assume a class exists named `Person`. Which of the follow statements constructs an object of the `Person` class?

1. `bob = __init__(Person)`
2. `bob = Person.__init__()`
3. `bob = Person`
4. `bob = Person()`

Title

Which statement creates an object?

type

mc

Section

9.2 Implementing a Class

id

testbank-py-2-ch09-22

from

testbank-py-1-ch09-22

23. Objects with the same behaviour are grouped into:

1. **Classes**
2. Functions
3. Groups
4. Interfaces

Title

How are objects with the same behaviour grouped?

type

mc

Section

9.1 Object-Oriented Programming

id

testbank-py-2-ch09-23

from

testbank-py-1-ch09-23

24. Where is an object's data stored?

1. Global variables
2. **Instance variables**
3. Local variables
4. Parameter variables

Title

Where is an object's data stored?

type

mc

Section

9.2 Implementing a Simple Class

id

testbank-py-2-ch09-24

from

testbank-py-1-ch09-24

25. A method that changes the object on which it operates is known as a(n):

1. Accessor
2. Interface
3. **Mutator**
4. Function

Title

What term is used to describe a method that changes the object on which it operates?

type

mc

Section

9.3 Specifying the Public Interface of a Class

id

testbank-py-2-ch09-25

from

testbank-py-1-ch09-25

26. A method that queries an object for some information without changing it is known as a(n):

1. **Accessor**

2. Function
3. Interface
4. Mutator

Title

What term is used to describe a method that queries an object for some information without changing it?

type

mc

Section

9.3 Specifying the Public Interface of a Class

id

testbank-py-2-ch09-26

from

testbank-py-1-ch09-26

27. Consider the following class:

```
class Pet:
    def __init__(self, name):
        self._name = name

    def getName(self):
        _____
```

What line of code should be placed in the blank to complete the `getName` accessor method that is supposed to return the pet's name?

1. `return`
2. `return _name`
3. `return self`
4. `return self._name`

Title

What code is needed to complete an accessor method?

type

mc

Section

9.6 Implementing Methods

id

testbank-py-2-ch09-27

from

testbank-py-1-ch09-27

28. When designing a class, what is one of the first tasks that need to be done?

1. Defining the instance variables
2. Writing the method headers
3. Specifying the public interface
4. Adding comments to your program

Title

When designing a class, what is one of the first tasks that need to be done?

type

mc

Section

9.3 Specifying the Public Interface of a Class

id

testbank-py-1-ch09-28

29. What items should be considered when creating the public interface?

1. Method headers and method comments
2. Instance variables
3. Data and method bodies
4. Method implementations

Title

What items should be considered when creating the public interface?

type

mc

Section

9.3 Specifying the Public Interface of a Class

id

testbank-py-1-ch09-29

30. Before invoking a method on an object, what must be done first?

1. Initialize all instance variables
2. Invoke the accessor and mutator methods
3. Create a public interface
4. Construct the object

Title

Before invoking a method on an object, what must be done first?

type

mc

Section

9.3 Specifying the Public Interface of a Class

id

testbank-py-1-ch09-30

31. Which method below would be considered an accessor method?

1. getCount()
2. addItem()
3. clearCount()
4. updateItem()

Title

Which method below would be considered an accessor method?

type

mc

Section

9.3 Specifying the Public Interface of a Class

id

testbank-py-1-ch09-31

32. Which method below would be considered a mutator method?

1. getCount()
2. addItem()
3. getTotal()
4. printItem()

Title

Which method below would be considered an mutator method?

type

mc

Section

9.3 Specifying the Public Interface of a Class

id

testbank-py-1-ch09-32

33. Assume that the class `ShoppingList` has already been defined with the following public interface:

```
## A simulated grocery list
#
class ShoppingList :
```

```

## Adds an item to this shopping list.
# @param cost the cost of this item
# @param quantity the number of items
# @param itemName the name of the item
#
def addItem(self, cost, quantity, itemName) :
    # implementation hidden

## Gets the number of items in the list.
# @return the item count
#
def getCount(self) :
    # implementation hidden

## Clears the list.
#
def clear(self) :
    # implementation hidden

```

What is wrong with the following code segment?

```

list.clear()
list = ShoppingList()
list.addItem(2.95, 1, "milk")
list.addItem(.50, 3, "cucumber")
print(list.getCount())

```

1. Nothing, the program prints 2 for the number of items in the list.
2. The `list` object has not been created before invoking the `clear` method.
3. The `addItem` method does not take three arguments.
4. There is no `print` method defined for the `ShoppingList` class.

Title

What is wrong with the following code segment that invokes methods on an object?

type

mc

Section

9.3 Specifying the Public Interface of a Class

id

testbank-py-1-ch09-33

34. Assume that the class `ShoppingList` has already been defined with the following public interface:

```

## A simulated grocery list
#
class ShoppingList :

## Adds an item to this shopping list.
# @param cost the cost of this item
# @param quantity the number of items
# @param itemName the name of the item
#
def addItem(self, cost, quantity, itemName) :
    # implementation hidden

## Gets the number of items in the list.
# @return the item count
#
def getCount(self) :
    # implementation hidden

## Clears the list.
#
def clear(self) :
    # implementation hidden

```

What is wrong with the following code segment?

```

list = ShoppingList()
list.clear()
list.addItem(2.95, 1, "milk")

```

```
list.addItem(.50, 3, "cucumber")
print(list.getItems())
```

1. Nothing, the program prints 2 for the number of items in the list.
2. The `list` object has not been created before invoking the `clear` method.
3. The `addItem` method does not take three values.
4. There is no `getItems` method defined for the `ShoppingList` class.

Title

What is wrong with the following code segment that invokes methods on an object?

type

mc

Section

9.3 Specifying the Public Interface of a Class

id

testbank-py-1-ch09-34

35. Consider the following class:

```
class Fruit :           # Line 1
    def getColor(self) : # Line 2 ✓
        retval = self._color # Line 3
        return retval      # Line 4
```

Which line(s) are part of the public interface for the class?

1. Only line 1
2. Only line 2
3. Only lines 2 and 3
4. Only lines 3 and 4

Title

Which line(s) are part of the public interface of a class?

type

mc

Section

9.3 Specifying the Public Interface of a Class

id

testbank-py-1-ch09-35

36. Consider the following class:

```
class Counter :
    def getValue(self) :
        return self._value

    def click(self) :
        self._value = self._value + 1

    def unClick(self) :
        self._value = self._value - 1

    def reset(self) :
        self._value = 0
```

Which method is an accessor?

1. `getValue`
2. `click`
3. `unClick`
4. `reset`

Title

Which method is an accessor?

type

mc

Section

9.3 Specifying the Public Interface of a Class

id

testbank-py-1-ch09-36

37. Consider the following class:

```
class Counter :
    def getValue(self) :
        return self._value

    def click(self) :
        self._value = self._value + 1

    def unClick(self) :
        self._value = self._value - 1

    def reset(self) :
        self._value = 0
```

Which method(s) are mutators?

1. Only reset
2. Only click and unClick
3. Only click, unClick and reset
4. All four methods are mutators

Title

Which method(s) are mutators?

type

mc

Section

9.3 Specifying the Public Interface of a Class

id

testbank-py-1-ch09-37

38. In the following example, which data is considered instance data?

You are assigned the task of writing a program that calculates payroll for a small company. To get started the program should do the following:

- Add new employees including their first and last name and hourly wage
- Ask for the number of hours worked
- Calculate payroll (applying 1.5 for any hours greater than 40)
- Print a report of all employees' salary for the week, total of all hours and total of all salaries

1. firstName, lastName, hoursWorked, hourlyWage
2. firstName, lastName, hoursWorked, hourlyWage, payrollAmount
3. firstName, lastName, hoursWorked, hourlyWage, totalHours, totalSalary
4. firstName, lastName, hoursWorked, hourlyWage, payrollAmount, totalHours, totalSalary

Title

In the following example, which data should be considered instance data?

type

mc

Section

9.4 Designing the Data Representation

id

testbank-py-1-ch09-38

39. Which name would be best for a private instance variable?

1. confidential
2. HIDDEN
3. private
4. Secret

Title

Which name would be best for a private instance variable?

type
mc
Section
9.4 Designing the Data Representation
id
testbank-py-1-ch09-39

40. What is the purpose of a constructor?
1. To specify the public interface
 2. To identify the accessor and mutator methods
 3. To define and initialize the instance variables of an object
 4. To print the contents of an object

Title
What is the purpose of a constructor?
type
mc
Section
9.5 Constructors
id
testbank-py-1-ch09-40

41. When is a constructor invoked?
1. When an object is created
 2. When instance data is modified
 3. When a method is invoked
 4. When multiple arguments are required to create an object

Title
When is a constructor invoked?
type
mc
Section
9.5 Constructors
id
testbank-py-1-ch09-41

42. Which of the following method headers represent a constructor?

1. `def init(self) :`
2. `def __init__(self) :`
3. `def _init(self) :`
4. `def init() :`

Title
Which of the following method headers represent a constructor?
type
mc
Section
9.5 Constructors
id
testbank-py-1-ch09-42

43. Consider the class `Employee`:

```
class Employee :
    def __init__(self, firstName, lastName, employeeId) :
        self._name = lastName + "," + firstName
        self._employeeId = employeeId
        . . .
```

If an object is constructed as:

```
sam = Employee("Sam", "Fisher", 54321)
```

What is the contents of the instance variable `_name`?

1. Sam
2. Sam Fisher
3. Fisher, Sam
4. Fisher, Sam, 54321

Title

Given the constructor for the class `Employee`, what is the contents of one of the instance variables?

type

mc

Section

9.5 Constructors

id

testbank-py-1-ch09-43

44. Which of the following method headers represent a constructor for an `Employee` class?

1. `def Employee(self) :`
2. `def __init()__ :`
3. `def __Employee__(self) :`
4. `def __init__(self) :`

Title

Which of the following method headers represent a constructor?

type

mc

Section

9.5 Constructors

id

testbank-py-1-ch09-44

45. How many constructors can be defined for each class?

1. At least one must be defined
2. Only one may be defined
3. At least one, but as many as needed
4. At least one, but no more than five

Title

How many constructors can be defined for each class?

type

mc

Section

9.5 Constructors

id

testbank-py-1-ch09-45

46. Consider the following code segment:

```
class PhoneNumber :

    def __init__(self, lName, phone = "215-555-1212") :
        self._name = lName
        self._phone = phone

    def getName(self):
        return self._name

    def getPhone(self):
        return self._phone
```

```
Jones = PhoneNumber("Jones")
print(Jones.getName(), Jones.getPhone())
```

What will be printed when it executes?

1. Jones 215-555-1212
2. AttributeError: 'PhoneNumber' object has no attribute
3. Jones
4. Jones 000-000-0000

Title

What is printed by a code segment that creates an object?

type

mc

Section

9.5 Constructors

id

testbank-py-1-ch09-46

47. What change needs to be made in the following code segment so that `lName` will have a default value of "unknown"?

```
class PhoneNumber :
    def __init__(self, lName, phone = "215-555-1212") :
        self._name = lName
        self._phone = phone
```

1. Replace `self._name = lName` with `self._name = "unknown"`
2. Replace the `lName` parameter with `lName = "unknown"`
3. Add `name = "unknown"` to the end of the constructor
4. Add `self._lName = "unknown"` to the end of the constructor

Title

What change should be made to a code segment so that the constructor includes a default value for a parameter?

type

mc

Section

9.5 Constructors

id

testbank-py-1-ch09-47

48. What method name is used for a constructor?

1. `__begin__`
2. `__construct__`
3. `__create__`
4. `__init__`

Title

What method name is used for a constructor?

type

mc

Section

9.5 Constructors

id

testbank-py-1-ch09-48

49. What is wrong with the following constructor?

```
def __init__() :
    self._weight = 0.0
    self._color = "None"
```

1. The constructor must have a different name
2. The constructor must take the `self` parameter

3. The constructor must not initialize private instance variables
4. The constructor must initialize `_color` before `_weight`

Title

What is wrong with this constructor?

type

mc

Section

9.5 Constructors

id

testbank-py-1-ch09-49

50. Consider the following code segment which constructs two objects of type `Fruit`:

```
x = Fruit()
y = Fruit("Banana", "Yellow")
```

Which constructor header will construct both objects successfully?

1. `def __init__(name, color) :`
2. `def __init__(name="", color="") :`
3. `def __init__(self, name, color) :`
4. `def __init__(self, name="", color="") :`

Title

Which constructor head will allow two objects to be constructed successfully?

type

mc

Section

9.5 Constructors

id

testbank-py-1-ch09-50

51. How do you access instance variables in a method?

1. Using the constructor
2. Using the public interface
3. Using a default value
4. Using the `self` reference

Title

How do you access instance variables in a method?

type

mc

Section

9.6 Implementing Methods

id

testbank-py-1-ch09-51

52. Given the following code snippet, what statement completes the code to add several items to one grocery list:

```
def addItems(self, price, quantity, itemName) :
    for i in range(quantity) :
```

```
def addItem(self, price, itemName) :
    self._itemCount = self._itemCount + 1
    self._totalPrice = self._totalPrice + price
    . . .
```

1. `self.addItem(price, itemName)`
2. `addItem(price, itemName)`
3. `self._addItem(price, itemName)`
4. `_addItem(price, itemName)`

Title

Given the following code snippet, what statement completes the code to add several items to one grocery list:

type

mc

Section

9.6 Implementing Methods

id

testbank-py-1-ch09-52

53. Consider the following code segment:

```
def mutate(self, newType) :
    self._type = newType
    self._mutations = self._mutations + 1
```

What is the name of the local variable in it:

1. mutate
2. _mutations
3. newType
4. _type

Title

Identify the local variable in a method

type

mc

Section

9.6 Implementing Methods

id

testbank-py-1-ch09-53

54. Consider the following code segment:

```
class Fruit :
    _type = "Fruit"

    def __init__(self, color) :
        self._color = color
```

What is the name of the class variable?

1. color
2. _color
3. self
4. _type

Title

What is the name of the class variable?

type

mc

Section

9.6 Implementing Methods

id

testbank-py-1-ch09-54

55. What is the purpose of unit testing?

1. To verify that the constructor works correctly.
2. To verify that the entire program works correctly.
3. To verify that a class works correctly in isolation, outside a complete program.
4. To verify that one method works correctly.

Title

What is the purpose of unit testing?

type

mc

Section

9.7 Testing a Class

id
testbank-py-1-ch09-55

56. Which of the following is NOT considered part of unit testing for a class?
1. Construct an object
 2. Identify syntax errors
 3. Call methods
 4. Verify the program returns the expected values

Title
Which of the following is NOT considered part of unit testing for a class?
type
mc
Section
9.7 Testing a Class
id
testbank-py-1-ch09-56

57. Which of the following is NOT a step carried out by a tester program?
1. Construct one or more objects of the class
 2. Invoke one or more methods
 3. Print out one or more results
 4. Identify syntax errors

Title
Which of the following is NOT a step carried out by a tester program?
type
mc
Section
9.7 Testing a Class
id
testbank-py-1-ch09-57

58. When using the process of tracing objects for problem solving, which types of methods update the values of the instance variables?
1. Accessor methods
 2. Mutator methods
 3. Constructors
 4. Both B and C

Title
When using the process of tracing objects for problem solving, which types of methods update the values of the instance variables?
type
mc
Section
9.8 Problem Solving: Tracing Objects
id
testbank-py-1-ch09-58

59. Which of the follow code segments could be used to help test the `getColor` method from the `Fruit` class?

1. `print(f.getColor())`
`print("Expected: Yellow")`
2. `print("Yellow")`
`print("Expected: Yellow")`
3. `print(f.getColor())`
`print("Expected:", f.getColor())`
4. `print("Yellow")`
`print("Expected:", f.getColor())`

Title

Which code segment helps test a class?

type

mc

Section

9.7 Testing a Class

id

testbank-py-1-ch09-59

60. Consider the following class:

```
class Contact :
    def __init__(self, name, phone="")
        self._name = name
        self._phone = phone

    def getName(self) :
        return self._name

    def setName(self, new_name) :
        self._name = new_name

    def getPhone(self) :
        return self._phone

    def setPhone(self, new_phone) :
        self._phone = new_phone
```

What is output by the following code segment?

```
p1 = Contact("Bob", "555-123-4567")
p2 = Contact("Joe", "555-000-0000")
p1.setName(p2.getName())

print(p1.getPhone())
```

1. Bob
2. Joe
3. 555-000-0000
4. 555-123-4567

Title

Trace code involving objects and classes

type

mc

Section

9.8 Problem Solving: Tracing Objects

id

testbank-py-1-ch09-60

61. Which of the following patterns can be used for designing your class to help keep track of the number of fees charged by a bank?

1. Keeping a total
2. Counting events
3. Collecting values
4. Managing properties of an object

Title

Which of the following patterns can be used for designing your class to help track of the number of fees charged by a bank?

type

mc

Section

9.9 Problem Solving: Patterns for Object Data

id

testbank-py-1-ch09-61

62. Which of the following patterns can be used for designing your class to update the balance of a bank account?

1. Keeping a total

2. Counting events
3. Collecting values
4. Managing properties of an object

Title

Which of the following patterns can be used for designing your class to update the balance of a bank account?

type

mc

Section

9.9 Problem Solving: Patterns for Object Data

id

testbank-py-1-ch09-62

63. Which of the following patterns can be used for designing your class to add error checking such as not allowing a blank name for a bank account?
1. Describing the Position of an Object
 2. Modeling Objects with Distinct States
 3. Managing Properties of an Object
 4. Collecting Values

Title

Which of the following patterns can be used for designing your class to add error checking such as not allowing a blank name for a bank account?

type

mc

Section

9.9 Problem Solving: Patterns for Object Data

id

testbank-py-1-ch09-63

64. Which of the following patterns can be used for designing your class for a train object that drives along a track and keeps track of the distance from the terminus?
1. Describing the Position of an Object
 2. Modeling Objects with Distinct States
 3. Managing Properties of an Object
 4. Collecting Values

Title

Which of the following patterns can be used for designing your class for a train object that drives along a track and keeps track of the distance from the terminus?

type

mc

Section

9.9 Problem Solving: Patterns for Object Data

id

testbank-py-1-ch09-64

65. Consider the following class which is used to represent a polygon consisting of an arbitrary number of (x, y) points:

```
class Polygon :
    def __init__(self) :
        self._x_points = []
        self._y_points = []
```

Which of the following code segments is the correct implementation for the `addPoint` method that adds another point to the polygon?

1.

```
def addPoint(self, x, y) :
    self._points.append(x, y)
```
2.

```
def addPoint(self, x, y) :
    self._x_points.append(x)
    self._y_points.append(y)
```

3.

```
def addPoint(self, x, y) :
    self._x_points = x
    self._y_points = y
```
4.

```
def addPoint(self, x, y) :
    self._x_points = [x]
    self._y_points = [y]
```

Title

Implement a new method in a class

type

mc

Section

9.9 Problem Solving: Patterns for Object Data

id

testbank-py-1-ch09-65

66. Consider a class that represents a hardware device. The device can be in one of two states: Plugged in, or unplugged. Which of the following class definitions is best for this situation?

1.

```
PLUGGED_IN = 0
UNPLUGGED = 1

class Device :
    def __init__(self) :
        . . .
```
2.

```
class Device :
    PLUGGED_IN = 0
    UNPLUGGED = 1

    def __init__(self) :
        . . .
```
3.

```
class Device :
    def __init__(self) :
        PLUGGED_IN = 0
        UNPLUGGED = 1
        . . .
```
4.

```
class Device :
    def __init__(self) :
        self.PLUGGED_IN = 0
        self.UNPLUGGED = 1
        . . .
```

Title

Which class best represents a device with 2 possible states?

type

mc

Section

9.9 Problem Solving: Patterns for Object Data

id

testbank-py-1-ch09-66

67. What is the value of the variable `bankAcct` after the following code snippet is executed?

```
bankAcct = BankAccount("Fisher", 1000.00)
```

1. "Fisher", 1000.00
2. A memory location
3. "Fisher"
4. 1000.00

Title

What is the value of the variable `bankAcct` after the following code snippet is executed?

type

mc
 Section
 9.10 Object References
 id
 testbank-py-1-ch09-67

68. Consider the following code segment:

```
bankAcct = BankAccount("Fisher", 1000.00)
bankAcct2 = bankAcct
```

What is the relationship between bankAcct and bankAcct2?

1. bankAcct and bankAcct2 are self references
2. bankAcct and bankAcct2 are aliases for the same object reference
3. bankAcct and bankAcct2 are NONE references
4. bankAcct and bankAcct2 are separate references

Title
 What is the relationship between two object references in a code segment?
 type
 mc
 Section
 9.10 Object References
 id
 testbank-py-2-ch09-68
 from
 testbank-py-1-ch09-68

69. Given the following code snippet, how can you test if they reference the same object (or does not refer to any object)?

```
bankAcct2 = bankAcct

1. if bankAcct == bankAcct2 :
    print("The variables are aliases")

2. if bankAcct is bankAcct2 :
    print("The variables are aliases")

3. if bankAcct is not bankAcct2 :
    print("The variables are aliases")

4. if bankAcct.equals(bankAcct2) :
    print("The variables are aliases")
```

Title
 Given the following pre snippet, how can you test if they reference the same object?
 type
 mc
 Section
 9.10 Object References
 id
 testbank-py-1-ch09-69

70. Which statement determines if the middleInit variable is empty or does not refer to any object?

1. if middleInit is None :
 print("No middle initial")
2. if middleInit is not None :
 print("No middle initial")
3. if middleInit == None :
 print("No middle initial")
4. if middleInit != None :
 print("No middle initial")

Title
Which statement determines if the `__middleInit` variable is empty or does not refer to any object?

type
mc

Section
9.10 Object References

id
testbank-py-1-ch09-70

71. What happens when an object is no longer referenced?
1. Nothing until the problem terminates
 2. The garbage collector removes the object
 3. The address is saved for future instances of the object
 4. Creates a `None` reference

Title
What happens when an object is no longer referenced?

type
mc

Section
9.10 Object References

id
testbank-py-1-ch09-71

72. Which of the following is NOT true about objects?
1. An object reference specifies the location of an object
 2. Multiple object variables can contain references to the same object
 3. The `==` and `!=` operators test whether two variables are aliases
 4. The `None` reference refers to no object

Title
Which of the following is NOT true about objects?

type
mc

Section
9.10 Object References

id
testbank-py-1-ch09-72

73. Which of the following is NOT true about constructors?
1. A constructor initializes the instance variables of an object
 2. The constructor is automatically called when an object is created
 3. The constructor is defined using the special method name `__default__`
 4. Default arguments can be used with a constructor to provide different ways of creating an object

Title
Which of the following is NOT true about constructors?

type
mc

Section
9.5 Constructors

id
testbank-py-1-ch09-73

74. Which of the following is NOT true about instance methods for a class?
1. The object on which a method is applied is automatically passed to the `self` parameter variable of the method
 2. In a method, you access instance variables through the `self` parameter variable
 3. A class variable belongs to the class, not to any instance of the class
 4. The accessor and mutator methods are automatically called when an object is created

Title
Which of the following is NOT true about instance methods for a class?

type

mc

Section

9.6 Implementing Methods

id

testbank-py-1-ch09-74

75. Which of the following is NOT a pattern used to help design the data representation of a class?
1. An instance variable for the total is updated in methods that increase or decrease the total amount
 2. An object reference specifies the location of an object
 3. An object can collect other objects in a list
 4. To model a moving object, you need to store and update its position

Title

Which of the following is NOT true about instance methods for a class?

type

mc

Section

9.9 Problem Solving: Patterns for Object Data

id

testbank-py-1-ch09-75

76. Given the variable assignment `sentence = ""`, what is the value of `len(sentence)`?
1. 1
 2. -1
 3. an exception is raised
 4. 0

Title

Given the variable assignment `sentence = ""`, what is the value of `len(sentence)`?

type

mc

Section

9.10 Objects and Classes

id

testbank-py-1-ch09-76

77. Which of the following statements sets `x` so that it refers to no object at all?
1. `x =`
 2. `x = ()`
 3. `x = ""`
 4. `x = None`

Title

Which statement gives a variable a variable that refers to no object at all?

type

mc

Section

9.10 Object References

id

testbank-py-1-ch09-77

78. Recall the Cash Register class developed in the textbook. What is output by the following code segment?

```
from cashregister2 import CashRegister

reg1 = CashRegister()
reg1.addItem(3.00)
reg2 = reg1
reg1.addItem(5.00)
print(reg2.getCount())
```

1. 0
2. 1
3. 2

4. The program terminates with a runtime error

Title
Trace code involving object references
type
mc
Section
9.10 Object References
id
testbank-py-1-ch09-78

79. Recall the Cash Register class developed in the textbook. What is output by the following code segment?

```
from cashregister2 import CashRegister

reg1 = CashRegister()
reg2 = reg1
reg1.addItem(3.00)
reg1.addItem(5.00)
reg2.clear()
print(reg1.getCount())
```

1. 0
2. 1
3. 2
4. The program terminates with a runtime error

Title
Trace code involving object references
type
mc
Section
9.10 Object References
id
testbank-py-1-ch09-79

80. Assume that `x` is a variable containing an object reference. Which special method is invoked when the following statement executes?

```
print(x)
```

1. `__init__`
2. `__print__`
3. `__repr__`
4. `__str__`

Title
Which special method is invoked when an object is printed?
type
mc
Section
9.11 Application: Writing a Fraction Class
id
testbank-py-1-ch09-80

81. Which of the following statements determines if `x` currently refers to an object containing an integer?

1. `if int(x) :`
2. `if x == int :`
3. `if x is int :`
4. `if isinstance(x, int) :`

Title
Which statement determines if `x` refers to an integer?
type
mc
Section

9.11 Application: Writing a Fraction Class

id

testbank-py-1-ch09-81

82. Consider the following class:

```
class Pet:
    def __init__(self, name):
        self._name = name
        self._owner = "<unknown>"

    def setOwner(self, owner):
        _____
```

What line of code should be placed in the blank to complete the `setOwner` mutator method that is supposed to update the pet owner's name?

1. `self._name = name`
2. `self._name = owner`
3. `self._owner = owner`
4. `self.owner = _owner`

Title

What code is needed to complete a mutator method?

type

mc

Section

9.6 Implementing Methods

id

testbank-py-2-ch09-82

83. Class variables are also known as:

1. global variables
2. interface variables
3. local variables
4. static variables

Title

Which term is a synonym for "class variable"?

type

mc

Section

9.6 Implementing Methods

id

testbank-py-2-ch09-83

84. Where should instance variables be defined?

1. In accessor methods
2. In constructors
3. In functions
4. In mutator methods

Title

Where should instance variables be defined?

type

mc

Section

9.6 Implementing Methods

id

testbank-py-2-ch09-84

85. Consider the following class:

```
class Pet:
    _____

    def __init__(self, name):
```

```

self._name = name
Pet._lastID = Pet._lastID + 1
self._registrationID = Pet._lastID

```

What line of code should be placed in the blank to create a class variable that keeps track of the most recently used registration identifier?

1. `_lastID = 0`
2. `Pet.append(_lastID)`
3. `Pet._lastID = 0`
4. `self._lastID = 0`

Title

What code is needed to add a class variable to an existing class?

type

mc

Section

9.6 Implementing Methods

id

testbank-py-2-ch09-85

86. What term is used to describe the process of verifying that a class works correctly in isolation, outside of a complete program?

1. Function testing
2. Interface testing
3. Method testing
4. Unit testing

Title

What term is used to describe the process of verifying that a class works correctly in isolation?

type

mc

Section

9.7 Testing a Class

id

testbank-py-2-ch09-86

87. What does an object reference specify?

1. The hashcode of an object
2. The location of an object
3. The name of an object
4. The size of an object

Title

What does an object reference specify?

type

mc

Section

9.10 Object References

id

testbank-py-2-ch09-87

88. What part of the virtual machine is responsible for removing objects that are no longer referenced?

1. The allocator
2. The garbage collector
3. The recycler
4. The regenerator

Title

What part of the virtual machine is responsible for removing objects that are no longer referenced?

type

mc

Section

9.10 Object References

id

testbank-py-2-ch09-88

89. Consider the following class which will be used to represent complex numbers:

```
class Complex:
    def __init__(self, real, imaginary):
        self._real = real
        self._imaginary = imaginary

    def _____:
        real = self._real + rhsValue._real
        imaginary = self._imaginary + rhsValue._imaginary
        return Complex(real, imaginary)
```

What code should be placed in the blank so that two complex numbers can be added using the + operator?

1. +(self, rhsValue)
2. __+__(self, rhsValue)
3. add
4. __add__(self, rhsValue)

Title

What statement should be added to a class so that two instances of the class can be added using a plus sign?

type

mc

Section

9.11 Application: A Fraction Class

id

testbank-py-2-ch09-89