1. Which of the following best describes polymorphism?  
   a) Ability of a class to derive members of another class as a part of its own definition  
   b) Means of bundling instance variables and methods in order to restrict access to certain class members  
   c) Focuses on variables and passing of variables to functions  
   d) Allows for objects of different types and behaviour to be treated as the same general type

Answer: d  
Explanation: Polymorphism is a feature of object-oriented programming languages. It allows for the implementation of elegant software that is well designed and easily modified.

1. What is the biggest reason for the use of polymorphism?  
   a) It allows the programmer to think at a more abstract level  
   b) There is less program code to write  
   c) The program will have a more elegant design and will be easier to maintain and update  
   d) Program code takes up less space

Answer: c  
Explanation: Polymorphism allows for the implementation of elegant software.

1. What is the use of duck typing?  
   a) More restriction on the type values that can be passed to a given method  
   b) No restriction on the type values that can be passed to a given method  
   c) Less restriction on the type values that can be passed to a given method  
   d) Makes the program code smaller

Answer: c  
Explanation: In Python, any set of classes with a common set of methods can be treated similarly. This is called duck typing. Hence duck typing imposes less restrictions.

4. What will be the output of the following Python code?

class A:

def \_\_str\_\_(self):

return '1'class B(A):

def \_\_init\_\_(self):

super().\_\_init\_\_()class C(B):

def \_\_init\_\_(self):

super().\_\_init\_\_()def main():

obj1 = B()

obj2 = A()

obj3 = C()

print(obj1, obj2,obj3)

main()

a) 1 1 1  
b) 1 2 3  
c) ‘1’ ‘1’ ‘1’  
d) An exception is thrown

Answer: a  
Explanation: The super().\_\_init\_\_() in the subclasses has been properly invoked and none of other subclasses return any other value. Hence 1 is returned each time the object is created and printed.

5. What will be the output of the following Python code?

class Demo:

def \_\_init\_\_(self):

self.x = 1

def change(self):

self.x = 10class Demo\_derived(Demo):

def change(self):

self.x=self.x+1

return self.xdef main():

obj = Demo\_derived()

print(obj.change())

main()

a) 11  
b) 2  
c) 1  
d) An exception is thrown

Answer: b  
Explanation: The derived class method change() overrides the base class method.

1. A class in which one or more methods are only implemented to raise an exception is called an abstract class.  
   a) True  
   b) False

Answer: a  
Explanation: A class in which one or more methods are unimplemented or implemented for the methods throw an exception is called an abstract class.

1. Overriding means changing behaviour of methods of derived class methods in the base class.  
   a) True  
   b) False

Answer: b  
Explanation: Overriding means if there are two same methods present in the superclass and the subclass, the contents of the subclass method are executed.

8. What will be the output of the following Python code?

class A:

def \_\_repr\_\_(self):

return "1"class B(A):

def \_\_repr\_\_(self):

return "2"class C(B):

def \_\_repr\_\_(self):

return "3"

o1 = A()

o2 = B()

o3 = C()print(obj1, obj2, obj3)

a) 1 1 1  
b) 1 2 3  
c) ‘1’ ‘1’ ‘1’  
d) An exception is thrown

Answer: b  
Explanation: When different objects are invoked, each of the individual classes return their individual values and hence it is printed.

9. What will be the output of the following Python code?

class A:

def \_\_init\_\_(self):

self.multiply(15)

print(self.i)

def multiply(self, i):

self.i = 4 \* i;class B(A):

def \_\_init\_\_(self):

super().\_\_init\_\_()

def multiply(self, i):

self.i = 2 \* i;

obj = B()

a) 15  
b) 60  
c) An exception is thrown  
d) 30

Answer: d  
Explanation: The derived class B overrides base class A.

10. What will be the output of the following Python code?

class Demo:

def check(self):

return " Demo's check "

def display(self):

print(self.check())class Demo\_Derived(Demo):

def check(self):

return " Derived's check "

Demo().display()

Demo\_Derived().display()

a) Demo’s check Derived’s check  
b) Demo’s check Demo’s check  
c) Derived’s check Demo’s check  
d) Syntax error

Answer: a  
Explanation: Demo().display() invokes the display() method in class Demo and Demo\_Derived().display() invokes the display() method in class Demo\_Derived.

11. What will be the output of the following Python code?

class A:

def \_\_init\_\_(self):

self.multiply(15)

def multiply(self, i):

self.i = 4 \* i;class B(A):

def \_\_init\_\_(self):

super().\_\_init\_\_()

print(self.i)

def multiply(self, i):

self.i = 2 \* i;

obj = B()

a) 15  
b) 30  
c) An exception is thrown  
d) 60

Answer: b  
Explanation: The derived class B overrides base class A.

12. What will be the output of the following Python code?

class Demo:

def \_\_check(self):

return " Demo's check "

def display(self):

print(self.check())class Demo\_Derived(Demo):

def \_\_check(self):

return " Derived's check "

Demo().display()

Demo\_Derived().display()

a) Demo’s check Derived’s check  
b) Demo’s check Demo’s check  
c) Derived’s check Demo’s check  
d) Syntax error

Answer: b  
Explanation: The method check is private so it can’t be accessed by the derived class. Execute the code in the Python shell.

13. What will be the output of the following Python code?

class A:

def \_\_init\_\_(self, x, y):

self.x = x

self.y = y

def \_\_str\_\_(self):

return 1

def \_\_eq\_\_(self, other):

return self.x \* self.y == other.x \* other.y

obj1 = A(5, 2)

obj2 = A(2, 5)print(obj1 == obj2)

a) False  
b) 1  
c) True  
d) An exception is thrown

Answer: c  
Explanation: Since 5\*2==2\*5, True is printed. Execute it in the Python shell to verify.

14. What will be the output of the following Python code?

class A:

def one(self):

return self.two()

def two(self):

return 'A' class B(A):

def two(self):

return 'B'

obj2=B()print(obj2.two())

a) A  
b) An exception is thrown  
c) A B  
d) B

Answer: d  
Explanation: The derived class method two() overrides the method two() in the base class A.

15. Which of the following statements is true?  
a) A non-private method in a superclass can be overridden  
b) A subclass method can be overridden by the superclass  
c) A private method in a superclass can be overridden  
d) Overriding isn’t possible in Python

Answer: a  
Explanation: A public method in the base class can be overridden by the same named method in the subclass.

1. Which of these is not a fundamental feature of OOP?

|  |  |
| --- | --- |
|  | Encapsulation |
|  | Inheritance |
|  | Instantiation |
|  | Polymorphism |

Answer: C

1. Which among the following is the language which supports classes but not polymorphism?
2. SmallTalk
3. Java
4. C++
5. Ada

Answer: d

1. If same message is passes to objects of several different classes and all of those can respond in a different way, what is this feature called?
2. Inheritance
3. Overloading
4. Polymorphism
5. Overriding
6. Which type of function among the following shows polymorphism?
7. Inline function
8. Virtual function
9. Undefined function
10. Class member function

Ans: b

1. In case of using abstract class or function overloading. Which function is supposed to be called first?
2. Local function
3. Function with highest priority in compiler
4. Global function
5. Function with lowest priority

ANS: B

1. Which among the following cant be used for polymorphism?
2. Static member function
3. Member functions overloading
4. Predefined operator overloading
5. Constructor overloading

ANS: A

1. Which among the following can show polymorphism?
2. Overloading ||
3. Overloading +=
4. Overloading <<
5. Overloading &&

Ans: C

1. Which problem may arise if we use abstract class function for polymorphism?
2. All classes are converted as abstract class
3. Derived class must be of abstract type
4. All the derived classes must implement the undefined functions
5. Derived classes cant redefine the function

Ans: C

1. Polymorphism is possible in C language.
2. True
3. False

Ans: A

1. Which among the following is not true for polymorphism?
2. It is feature of OOP
3. Ease in readability of program
4. Help in redefining the same functionality
5. Increase overhead of function definition always

Ans: D