1. Which of the following represents a distinctly identifiable entity in the real world?

A. A class

**B. An object**

C. A method

D. A data field

2. What will be the output of the following code snippet?

class Sales:

    def \_\_init\_\_(self, id):

        self.id = id

        id = 100

val = Sales(123)

print (val.id)

A. SyntaxError, this program will not run

B. 100

**C. 123**

D. None of the above

Reason : we are just trying to access the id variable which relates to the sales class. To access it we use self.id.

Here id = 100 means we are actually overwriting the id parameter, it won’t affect self.id

3.Which of the following does not correctly create an object instance?

A. puppy = Dog("Jamie")

B. dog = Dog("Jamie")

C. jamie = Dog()

**D. pupper = new Dog("Jamie")**

Reason : no keyword named “new” is available in python to create an object instance.

4. What does the following code output?

class People():

    def \_\_init\_\_(self, name):

      self.name = name

    def namePrint(self):

      print(self.name)

  person1 = People("Emma")

  person2 = People("Watson")

  person1.namePrint()

**A. Emma**

B. Watson

C. Emma Watson

D. person1

Reason : we are just printing the person1’s name, that is Emma

5. \_\_\_\_\_\_\_\_\_ is not a keyword, but by convention it is used to refer to the current instance (object) of a class.

A. class

B. def

**C. self**

D. init

6.Which of the following is the correct way to define an initializer method?

A. def \_\_init\_\_(title, author):

**B. def \_\_init\_\_(self, title, author):**

C. def \_\_init\_\_():

D. \_\_init\_\_(self, title, author):

Reason : to initialize a constructor in python, we must use self in the parameter.

7.Which of the following represents a template, blueprint, or contract that defines objects of the same type?

**A. A class**

B. An object

C. A method

D. A data field

Reason : A class in a template or blueprint that creates an object

8. class Dog:

    def \_\_init\_\_(self, name, age):

        self.name = name

        self.age = age

 The correct way to instantiate the above Dog class is:

1. **Dog("Rufus", 3)**
2. Dog()
3. Dog.\_\_init\_\_("Rufus", 3)
4. Dog.create("Rufus", 3)

Reason : our constructor takes two parameters.

9.In Python, a function within a class definition is called a:

1. **a method**
2. a class function
3. a callable
4. an operation

10.   
class Person:

    def \_\_init\_\_(self, id):

        self.id = id

sam = Person(100)

sam.\_\_dict\_\_['age'] = 49

print (sam.age + len(sam.\_\_dict\_\_))

A. 1

B. 2

C. 49

**D. 51**

Reason : here sam.age = 49 and length of sam.dict is 2