UNIT-V
OBJECT ORIENTED PROGRAMMING

Multiple Choice Questions

1.Which of the following is correct with respect to OOP concept in Python?

A. Objects are real world entities while classes are not real.
B. Classes are real world entities while objects are not real.
C. Both objects and classes are real world entities.
D. Both object and classes are not real.

Ans : A

Explanation: In OOP, classes are basically the blueprint of the objects. They doesnot have physical existence.

2. How many objects and reference variables are there for the given Python code?

```
class  A:
          print("Inside  class")
A()
A()
obj=A()
```

A. 2 and 1
B. 3 and 3
C. 3 and 1
D. 3 and 2

Ans : C

Explanation: obj is the reference variable here and an object will be created each time A() is called.So there will be 3 objects created.

3. Which of the following is False with respect Python code?

```
class  Student:

          def  __init__(self,id,age):

                    self.id=id
```

```
                        self.age=age

std=Student(1,20)
```

A. "std" is the reference variable for object Student(1,20)
B. id and age are called the parameters.
C. Every class must have a constructor.
D. None of the above

Ans : C

Explanation: It is not mandatory for a class to have a constructor.

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

```
class  Student:

            def  __init__(self,name,id):

                        self.name=name

                        self.id=id

                        print(self.id)

std=Student("Simon",1)

std.id=2

print(std.id)
```

A. 1
   1
B. 1
   2
C. 2
   1
D. 2
   2

Ans : B

Explanation: When object with id =1 is created for Student, constructor is invoked and it prints 1. Later, id has been changed to 2, so next print statement prints 2.

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

```
class  A():
            def  __init__(self,count=100):
                        self.count=count
obj1=A()
obj2=A(102)
print(obj1.count)
print(obj2.count)
```

A. 100
   100
B. 100
   102
C. 102
   102
D. Error

Ans : B

Explanation: By default,the value of "count" is 100, so obj1.count=100. For second object, value of "count" is 102, so obj2.count=102.

6. Which of the following is correct?

```
class  A:

      def  __init__(self,name):

             self.name=name

a1=A("john")

a2=A("john")
```

A. id(a1) and id(a2) will have same value.
B. id(a1) and id(a2) will have different values.
C. Two objects with same value of attribute cannot be created.
D. None of the above

Ans : B

Explanation: Although both a1 and a2 have same value of attributes,but these two point to two different object. Hence, their id will be different.

7. Which of the following is correct?

```
class  A:

            def  __init__(self):
```

```
                        self.count=5

                        self.count=count+1

a=A()

print(a.count)
```

A. 5
B. 6
C. 0
D. Error

Ans : D

Explanation: It will throw an error as inside constructor, "count" is not defined.

8. Which of the following is correct?

```
class  Book:

            def  __init__(self,author):

                        self.author=author

book1=Book("V.M.Shah")

book2=book1
```

A. Both book1 and book2 will have reference to two different objects of class Book.
B. id(book1) and id(book2) will have same value.
C. It will throw error as multiple references to same object is not possible.
D. None of the above

Ans : B

Explanation: book1 and book2 will reference to the same object. Hence, id(book1) and id(book2) will have same value.

9. In python, what is method inside class?

A. attribute
B. object
C. argument
D. function

Ans : D

Explanation: In OOP of Python, function is known by "method".

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

```
class  A:

            def  __init__(self,num):

                        num=3

                        self.num=num

            def  change(self):

                        self.num=7

a=A(5)

print(a.num)

a.change()

print(a.num)
```

A. 5
   7
B. 5
   5
C. 3
   3
D. 3
   7

Ans : D

Explanation: Inside constructor, self.num=3. First print statement prints 3. As, method change() is invoked, self.num=7 and hence second print statement prints 7.

11. Which of these is not a fundamental features of OOP?
a) Encapsulation
b) Inheritance
c) Instantiation
d) Polymorphism

Answer: c
Explanation: Instantiation simply refers to creation of an instance of class. It is not a fundamental feature of OOP.

12. Which of the following is the most suitable definition for encapsulation?
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 implementation of elegant software that is well designed and easily modified

Answer: b
Explanation: The values assigned by the constructor to the class members is used to create the object.

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

```
class Demo:
    def __init__(self):
        self.a = 1
        self.__b = 1

    def display(self):
        return self.__b
obj = Demo()print(obj.a)
```

a) The program has an error because there isn’t any function to return self.a
b) The program has an error because b is private and display(self) is returning a private member
c) The program runs fine and 1 is printed
d) The program has an error as you can’t name a class member using __b

Answer: c
Explanation: The program has no error because the class member which is public is printed. 1 is displayed. Execute in python shell to verify.

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

```
class Demo:
    def __init__(self):
        self.a = 1
        self.__b = 1
    def display(self):
        return self.__b


obj = Demo()print(obj.__b)
```

a) The program has an error because there isn’t any function to return self.a
b) The program has an error because b is private and display(self) is returning a private member
c) The program has an error because b is private and hence can’t be printed
d) The program runs fine and 1 is printed

Answer: c
Explanation: Variables beginning with two underscores are said to be private members of the class and they can’t be accessed directly.

15. Methods of a class that provide access to private members of the class are called as ______ and ______
a) getters/setters
b) __repr__/__str__
c) user-defined functions/in-built functions
d) __init__/__del__

Answer: a
Explanation: The purpose of getters and setters is to get(return) and set(assign) private instance variables of a class.

16. Which of these is a private data field?

```
def Demo:def __init__(self):
    __a = 1
    self.__b = 1
    self.__c__ = 1
    __d__= 1
```

a) __a
b) __b
c) __c__
d) __d__

Answer: b
Explanation: Variables such as self.__b are private members of the class.

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

```
class Demo:
    def __init__(self):
        self.a = 1
        self.__b = 1

    def get(self):
        return self.__b

obj = Demo()print(obj.get())
```

a) The program has an error because there isn’t any function to return self.a
b) The program has an error because b is private and display(self) is returning a private member
c) The program has an error because b is private and hence can’t be printed
d) The program runs fine and 1 is printed

Answer: d
Explanation: Here, get(self) is a member of the class. Hence, it can even return a private member of the class. Because of this reason, the program runs fine and 1 is printed.

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

```
class Demo:
    def __init__(self):
        self.a = 1
        self.__b = 1
    def get(self):
        return self.__b
obj = Demo()
obj.a=45print(obj.a)
```

a) The program runs properly and prints 45
b) The program has an error because the value of members of a class can’t be changed from outside the class
c) The program runs properly and prints 1
d) The program has an error because the value of members outside a class can only be

changed as self.a=45

Answer: a
Explanation: It is possible to change the values of public class members using the object of the class.

19. Private members of a class cannot be accessed.
a) True
b) False

Answer: b
Explanation: Private members of a class are accessible if written as follows: obj._Classname__privatemember. Such renaming of identifiers is called as name mangling.

20. The purpose of name mangling is to avoid unintentional access of private class members.
a) True
b) False
Answer: a
Explanation: Name mangling prevents unintentional access of private members of a class, while still allowing access when needed. Unless the variable is accessed with its mangled name, it will not be found.

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

```
class fruits:
    def __init__(self):
        self.price = 100
        self.__bags = 5
    def display(self):
        print(self.__bags)
obj=fruits()
obj.display()
```

a) The program has an error because display() is trying to print a private class member
b) The program runs fine but nothing is printed
c) The program runs fine and 5 is printed
d) The program has an error because display() can’t be accessed

Answer: c
Explanation: Private class members can be printed by methods which are members of the class.

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

```
 class student:
     def __init__(self):
         self.marks = 97
         self.__cgpa = 8.7
     def display(self):
         print(self.marks)
obj=student()print(obj._student__cgpa)
```

a) The program runs fine and 8.7 is printed
b) Error because private class members can't be accessed
c) Error because the proper syntax for name mangling hasn't been implemented
d) The program runs fine but nothing is printed

Answer: a
Explanation: Name mangling has been properly implemented in the code given above and hence the program runs properly.

23. Which of the following is false about protected class members?
a) They begin with one underscore
b) They can be accessed by subclasses
c) They can be accessed by name mangling method
d) They can be accessed within a class

Answer: c
Explanation: Protected class members can't be accessed by name mangling.

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

```
class objects:
    def __init__(self):
        self.colour = None
        self._shape = "Circle"
    def display(self, s):
        self._shape = s
obj=objects()print(obj._objects_shape)
```

a) The program runs fine because name mangling has been properly implemented
b) Error because the member shape is a protected member
c) Error because the proper syntax for name mangling hasn’t been implemented
d) Error because the member shape is a private member

Answer: b
Explanation: Protected members begin with one underscore and they can only be accessed within a class or by subclasses.

25. Which feature of OOP indicates code reusability?
a) Encapsulation
b) Inheritance
c) Abstraction
d) Polymorphism

Answer: b
Explanation: Inheritance indicates the code reusability. Encapsulation and abstraction are meant to hide/group data into one element. Polymorphism is to indicate different tasks performed by a single entity.

26. If a function can perform more than 1 type of tasks, where the function name remains same, which feature of OOP is used here?
a) Encapsulation
b) Inheritance
c) Polymorphism
d) Abstraction

Answer: c
Explanation: For the feature given above, the OOP feature used is Polymorphism. Example of polymorphism in real life is a kid, who can be a student, a son, a brother depending on where he is.

27. If different properties and functions of a real world entity is grouped or embedded into a single element, what is it called in OOP language?
a) Inheritance
b) Polymorphism
c) Abstraction
d) Encapsulation

Answer: d

Explanation: It is Encapsulation, which groups different properties and functions of a real world entity into single element. Abstraction, on other hand, is hiding of functional or exact working of codes and showing only the things which are required by the user.

28. Which of the following is not a feature of pure OOP?
a) Classes must be used
b) Inheritance
c) Data may/may not be declared using object
d) Functions Overloading

Answer: c

Explanation: Data must be declared using objects. Object usage is mandatory because it in turn calls its constructors, which in turn must have a class defined. If object is not used, it is a violation of pure OOP concept.

29. Which among the following doesn't come under OOP concept?
a) Platform independent
b) Data binding
c) Message passing
d) Data hiding

Answer: a

Explanation: Platform independence is not feature of OOP. C++ supports OOP but it's not a platform independent language. Platform independence depends on programming language.

30. Which feature of OOP is indicated by the following code?

```
class student{   int marks;   };class topper:public student{   int age;   topper(int age){ this.age=age; } };
```

a) Inheritance
b) Polymorphism
c) Inheritance and polymorphism
d) Encapsulation and Inheritance

Answer: d

Explanation: Encapsulation is indicated by use of classes. Inheritance is shown by inheriting the student class into topper class. Polymorphism is not shown here because we have defined the constructor in the topper class but that doesn't mean that default constructor is overloaded.

31. Which feature may be violated if we don’t use classes in a program?
a) Inheritance can’t be implemented
b) Object must be used is violated
c) Encapsulation only is violated
d) Basically all the features of OOP gets violated

Answer: d
Explanation: All the features are violated because Inheritance and Encapsulation won’t be implemented. Polymorphism and Abstraction are still possible in some cases, but the main features like data binding, object use and etc won’t be used hence the use of class is must for OOP concept.

32. How many basic features of OOP are required for a programming language to be purely OOP?
a) 7
b) 6
c) 5
d) 4
Answer: a
Explanation: There are 7 basic features that define whether a programing language is pure OOP or not. The 4 basic features are inheritance, polymorphism, encapsulation and abstraction. Further, one is, object use is must, secondly, message passing and lastly, Dynamic binding.

33. The feature by which one object can interact with another object is _____________
a) Data transfer
b) Data Binding
c) Message Passing
d) Message reading

Answer: c
Explanation: The interaction between two object is called the message passing feature. Data transfer is not a feature of OOP. Also, message reading is not a feature of OOP.

34. ___________ underlines the feature of Polymorphism in a class.
a) Nested class
b) Enclosing class
c) Inline function
d) Virtual Function

Answer: d
Explanation: Virtual Functions can be defined in any class using the keyword virtual. All the classes which inherit the class containing the virtual function, define

the virtual function as required. Redefining the function on all the derived classes according to class and use represents polymorphism.

35. Which feature in OOP is used to allocate additional function to a predefined operator in any language?
a) Operator Overloading
b) Function Overloading
c) Operator Overriding
d) Function Overriding

Answer: a
Explanation: The feature is operator overloading. There is not a feature named operator overriding specifically. Function overloading and overriding doesn't give addition function to any operator.

36. Which among doesn't illustrates polymorphism?
a) Function overloading
b) Function overriding
c) Operator overloading
d) Virtual function

Answer: b
Explanation: Function overriding doesn't illustrate polymorphism because the functions are actually different and theirs scopes are different. Function and operator overloading illustrate proper polymorphism. Virtual functions show polymorphism because all the classes which inherit virtual function, define the same function in different ways.

37. Exception handling is a feature of OOP.
a) True
b) False

Answer: a
Explanation: Exception handling is a feature of OOP as it includes classes concept in most of the cases. Also it may come handy while using inheritance.

38. Which among the following, for a pure OOP language, is true?
a) The language should follow 3 or more features of OOP
b) The language should follow at least 1 feature of OOP
c) The language must follow only 3 features of OOP
d) The language must follow all the rules of OOP

Answer: d
Explanation: The language must follow all the rules of OOP to be called a purely OOP language. Even if a single OOP feature is not followed, then it's known to be a partially OOP language.

39. Does OOP provide better security than POP?
a) Always true for any programming language
b) May not be true with respect to all programming languages
c) It depends on type of program
d) It's vice-versa is true
Answer: a

Explanation: It is always true as we have the facility of private and protected access specifiers. Also, only the public and global data are available globally or else the program should have proper permission to access the private data.

40. Which among the following best describes polymorphism?
a) It is the ability for a message/data to be processed in more than one form
b) It is the ability for a message/data to be processed in only 1 form
c) It is the ability for many messages/data to be processed in one way
d) It is the ability for undefined message/data to be processed in at least one way

Answer: a
Explanation: It is actually the ability for a message / data to be processed in more than one form. The word polymorphism indicates many-forms. So if a single entity takes more than one form, it is known as polymorphism.

41. What do you call the languages that support classes but not polymorphism?
a) Class based language
b) Procedure Oriented language
c) Object-based language
d) If classes are supported, polymorphism will always be supported

Answer: c
Explanation: The languages which support classes but doesn't support polymorphism, are known as object-based languages. Polymorphism is such an important feature, that is a language doesn't support this feature, it can't be called as a OOP language.

42. Which among the following is the language which supports classes but not polymorphism?
a) SmallTalk
b) Java
c) C++
d) Ada

Answer: d
Explanation: Ada is the language which supports the concept of classes but doesn’t support the polymorphism feature. It is an object-based programming language. Note that it’s not an OOP language.

43. If same message is passed to objects of several different classes and all of those can respond in a different way, what is this feature called?
a) Inheritance
b) Overloading
c) Polymorphism
d) Overriding

Answer: c
Explanation: The feature defined in question defines polymorphism features. Here the different objects are capable of responding to the same message in different ways, hence polymorphism.

44. Which class/set of classes can illustrate polymorphism in the following code?

```
abstract class student{
   public : int marks;
   calc_grade();}class topper:public student{
    public : calc_grade()
    {
        return 10;
    }};class average:public student{
     public : calc_grade()
     {
         return 20;
     }};class failed{ int marks; };
```

a) Only class student can show polymorphism
b) Only class student and topper together can show polymorphism
c) All class student, topper and average together can show polymorphism
d) Class failed should also inherit class student for this code to work for polymorphism

Answer: c
Explanation: Since Student class is abstract class and class topper and average are inheriting student, class topper and average must define the function named

calc_grade(); in abstract class. Since both the definition are different in those classes, calc_grade() will work in different way for same input from different objects. Hence it shows polymorphism.

45. Which type of function among the following shows polymorphism?
a) Inline function
b) Virtual function
c) Undefined functions
d) Class member functions

Answer: b
Explanation: Only virtual functions among these can show polymorphism. Class member functions can show polymorphism too but we should be sure that the same function is being overloaded or is a function of abstract class or something like this, since we are not sure about all these, we can't say whether it can show polymorphism or not.

46. In case of using abstract class or function overloading, which function is supposed to be called first?
a) Local function
b) Function with highest priority in compiler
c) Global function
d) Function with lowest priority because it might have been halted since long time, because of low priority

Answer: b
Explanation: Function with highest priority is called. Here, it's not about the thread scheduling in CPU, but it focuses on whether the function in local scope is present or not, or if scope resolution is used in some way, or if the function matches the argument signature. So all these things define which function has the highest priority to be called in runtime. Local function could be one of the answer but we can't say if someone have used pointer to another function or same function name.

47. Which among the following can't be used for polymorphism?
a) Static member functions
b) Member functions overloading
c) Predefined operator overloading
d) Constructor overloading

Answer: a
Explanation: Static member functions are not property of any object. Hence it can't be considered for overloading/overriding. For polymorphism, function must be property of object, not only of class.

48. What is output of the following program?

```
class student{
     public : int marks;
          void disp()
          {
                    cout<<”its base class”
          };
          class topper:public student
          {
                    public :
                    void disp()
                    {
                              cout<<”Its derived class”;
                    }
          }
          void main() { student s; topper t;
          s.disp();
          t.disp();}
```

a) Its base classIts derived class
b) Its base class Its derived class
c) Its derived classIts base class
d) Its derived class Its base class

Answer: a
Explanation: You need to focus on how the output is going to be shown, no space will be given after first message from base class. And then the message from derived class will be printed. Function disp() in base class overrides the function of base class being derived.

49. Which among the following can show polymorphism?
a) Overloading ||
b) Overloading +=
c) Overloading <<
d) Overloading &&

Answer: c
Explanation: Only insertion operator can be overloaded among all the given options. And the polymorphism can be illustrated here only if any of these is applicable of being overloaded. Overloading is type of polymorphism.

50. Find the output of the following program.

```
class education{
          char name[10];
          public : disp()
          {
                    cout<<"Its education system";
          }
          class school:public education
          {
                    public: void dsip()
                    {
                              cout<<"Its school education system";
                    }
     };
          void main()
          {
                     school s;
                     s.disp();
          }}
```

a) Its school education system
b) Its education system
c) Its school education systemIts education system
d) Its education systemIts school education system

Answer: a
Explanation: Notice that the function name in derived class is different from the function name in base class. Hence when we call the disp() function, base class function is executed. No polymorphism is used here.

51. Polymorphism is possible in C language.
a) True
b) False

Answer: a
Explanation: It is possible to implement polymorphism in C language, even though it doesn’t support class. We can use structures and then declare pointers which in turn points to some function. In this way we simulate the functions like member functions but not exactly member function. Now we can overload these functions, hence implementing polymorphism in C language.

52. Which problem may arise if we use abstract class functions for polymorphism?
a) All classes are converted as abstract class
b) Derived class must be of abstract type
c) All the derived classes must implement the undefined functions
d) Derived classes can’t redefine the function

Answer: c
Explanation: The undefined functions must be defined is a problem, because one may need to implement few undefined functions from abstract class, but he will have to define each of the functions declared in abstract class. Being useless task, it is a problem sometimes.

53. Which among the following is not true for polymorphism?
a) It is feature of OOP
b) Ease in readability of program
c) Helps in redefining the same functionality
d) Increases overhead of function definition always

Answer: d
Explanation: It never increases function definition overhead, one way or another if you don’t use polymorphism, you will use the definition in some other way, so it actually helps to write efficient codes.

54. If 2 classes derive one base class and redefine a function of base class, also overload some operators inside class body. Among these two things of function and operator overloading, where is polymorphism used?
a) Function overloading only
b) Operator overloading only
c) Both of these are using polymorphism
d) Either function overloading or operator overloading because polymorphism can be applied only once in a program

Answer: d
Explanation: Both of them are using polymorphism. It is not necessary that

polymorphism can be used only once in a program, it can be used anywhere, any number of times in a single program.

Q. 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
Answer. B

Q. 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
Answer. A

Q. Which of the following keywords mark the beginning of the class definition?

A. def
B. return
C. class
D. All of the above.
Answer. C

Q. Which of the following is required to create a new instance of the class?

A. A constructor
B. A class
C. A value-returning method
D. A None method
Answer. A

Q. Which of the following statements is most accurate for the declaration x = Circle()?

A. x contains an int value.
B. x contains an object of the Circle type.
C. x contains a reference to a Circle object.
D. You can assign an int value to x.

Answer. C

Q. 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
Answer. C

Q. Which of the following statements are correct?

A. A reference variable is an object.
B. A reference variable refers to an object.
C. An object may contain other objects.
D. An object can contain the references to other objects.
Answer. B , D

Q. Which of the following can be used to invoke the __init__ method in B from A, where A is a subclass of B?

A. super().__init__()
B. super().__init__(self)
C. B.__init__()
D. B.__init__(self)
Answer. A, D

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

```
class  test:
     def  __init__(self,a=""Hello  World""):
          self.a=a
     def  display(self):
          print(self.a)
obj=test()
```

```
obj.display()
```

(A) The program has an error because constructor can’t have default arguments
(B) Nothing is displayed
(C) “Hello World” is displayed
(D) The program has an error display function doesn’t have parameters
Answer: C
Explanation: The program has no error. “Hello World” is displayed. Execute in python shell to verify.

Q. Which of the following is not a class method?

(A) Non-static
(B) Static
(C) Bounded
(D) Unbounded
Answer: A
Explanation: The three different class methods in Python are static, bounded and unbounded methods.

Q. What are the methods which begin and end with two underscore characters called?

(A) Special methods
(B) In-built methods
(C) User-defined methods
(D) Additional methods
Answer: A
Explanation: Special methods like __init__ begin and end with two underscore characters.

Q. Special methods need to be explicitly called during object creation.

(A) TRUE
(B) FALSE
Answer:B
Explanation: Special methods are automatically called during object creation.

Q. __del__ method is used to destroy instances of a class.

(A) TRUE
(B) FALSE
Answer: A
Explanation: ___del__ method acts as a destructor and is used to destroy objects of classes.

Q. Suppose B is a subclass of A, to invoke the __init__ method in A from B, what is the line of code you should write?

(A) A.__init__(self)
(B) B.__init__(self)
(C) A.__init__(B)

(D) B.__init__(A)
Answer: A
Explanation: To invoke the __init__ method in A from B, either of the following should be written: A.__init__(self) or super().__init__(self).

Q. What will be the output of below Python code?

```
class  Student:
        def  __init__(self,name,id):
                self.name=name
                self.id=id
                print(self.id)

std=Student("Simon",1)
std.id=2
print(std.id)
```

A. 1
   1
B. 1
   2
C. 2
   1
D. 2
   2
Ans : B

Explanation: When object with id =1 is created for Student, constructor is invoked and it prints 1. Later, id has been changed to 2, so next print statement prints 2.

Q. What will be the output of below Python code?

```
class  A():
        def  __init__(self,count=100):
                self.count=count
obj1=A()
obj2=A(102)
print(obj1.count)
print(obj2.count)
```

A. 100
   100
B. 100
   102
C. 102
   102
D. Error
Ans : B

Explanation: By default,the value of "count" is 100, so obj1.count=100. For second object, value of "count" is 102, so obj2.count=102.

Q. In python, what is method inside class?

A. attribute
B. object
C. argument
D. function
Ans : D

Explanation: In OOP of Python, function is known by "method".

Q. Study the following program:

```
class Std_Name:
    def __init__(self, Std_firstName, Std_Phn, Std_lastName):
        self.Std_firstName = Std_firstName
        self. Std_PhnStd_Phn = Std_Phn
        self. Std_lastNameStd_lastName = Std_lastName

Std_firstName = "Wick"
name = Std_Name(Std_firstName, 'F', "Bob")
Std_firstName = "Ann"
name.lastName = "Nick"
print(name.Std_firstName, name.Std_lastName)
```

What will be the output of this statement?

A. Ann Bob

B. Ann Nick

C. Wick Bob

D. Wick Nick

Answer: (d) Wick Nick

Q. Which of the following option is not a core data type in the python language?

A. Dictionary
B. Lists
C. Class
D. All of the above

Answer: C

Q. Which of these is a private data field?

```
def Demo:
def __init__(self):
    __a = 1
    self.__b = 1
    self.__c__ = 1
    __d__= 1
```

A. __a

B. __b

C. __c__

D. __d__

Answer: B
Explanation: Variables such as self.__b are private members of the class.

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

```
class demo():
        def __repr__(self):
                return '__repr__ built-in function called'
        def __str__(self):
                return '__str__ built-in function called'
s=demo()
print(s)
```

A. Error
B. Nothing is printed
C. __str__ called
D. __repr__ called
Answer: C
Explanation: __str__ is used for producing a string representation of an object's value that Python can evaluate. Execute in python shell to verify.

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

```
class test:
    def __init__(self,a="Hello World"):
        self.a=a

    def display(self):
        print(self.a)
obj=test()
obj.display()
```

A. The program has an error because constructor can't have default arguments
B. Nothing is displayed
C. "Hello World" is displayed
D. The program has an error display function doesn't have parameters

Answer: C
Explanation: The program has no error. "Hello World" is displayed. Execute in python shell to verify.