1. **The function below returns the n largest or smallest elements of a list.**

def sorted\_elements (x, desc = True, n = 2)

new\_x = sorted (x, reverse = desc) [0:n]

return new\_x

**What is the output of this code?**

a = [ 5, 5, 12, 12, 14, 7 ]

print (sorted\_elements (a, desc = False))

**Select the output:**

1. [ 14 ]
2. [ 5 , 5 ]
3. [ 5, 5, 7]
4. def square ( s ) :

“”” Returns area & perimeter of a square “””

\_**\_\_\_**

a = s \* s

p = 4 \* s

return a , p

**\_\_\_\_** TypeError :

print ( ‘s cannot be a string ’ )

square(‘4’)

**Output:**

s cannot be a string

**FILL IN THE BLANKS**

1. except
2. try :
3. return :
4. **Complete the code to return the output**

def easy\_print (**\_\_\_\_\_\_\_\_**) :

for key, value in x.items():

print ( “ The value of “ + “ str ( key ) + “ is “ + ( value ) )

(easy\_print ( a = 10) , easy\_print ( b = 20 ) )

**Output:**

The value of a is 10

The value of b is 20

**Fill in the blanks**

1. \*\*x
2. \*x
3. x

1. def rectangle \_\_\_\_\_\_\_

“ Returns the area and perimeter of a rectangle “

a = length \* width

p = 2 \* ( length + width )

return a, p

area, perimeter = rectangle ( 15, 3 )

print ( ( area, perimeter ))

**Output:**

( 45 , 36 )

**Fill in the blanks**

1. (area), (perimeter) :
2. (a , p ):
3. [ length , width ] :
4. (length , width )
5. (length , width ) :

1. **The following function converts yards to feet if feet is True, and to inches if feet is False.**

def convert\_yards ( y , feet = True ) :

if feet is True :

new\_y = y \* 3

else:

new\_y = y \* 36

return new\_y

**What is the output of this code ?**

print ( convert\_yards ( 20 ) )

**Select the Output:**

1. 3
2. 720
3. 60
4. **In the following class definition which variable is static ?**

class Duck:

animal = True

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

self.animal\_group = ‘Bird’

self.sound = ‘Quack!’

self.legs = 2

**Select the correct answer**

1. self.animal\_group
2. animal
3. self.sound
4. self.legs
5. **Which of these options is not a component of a class:**

**Select the correct answer**

1. Class Templates
2. Class Methods
3. Class Attributes
4. Class Constructor Method
5. Class Lion :

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

self.animal\_group = ‘Mammal’

self. sound = ‘Roar!!!’

def MakeSound (self) :

print ( self.sound )

lion\_object = Lion ()

lion\_object = lion\_object.MakeSound()

**Output:**

Roar!!!

**Fill in the blanks**

1. cons
2. init
3. \_\_cons\_\_
4. \_\_str\_\_
5. \_\_init\_\_
6. Complete the following statement:

“ O.O.P is a way to make logical groups of variables and functions…”

Select the correct answer

1. Unique for generalized tasks
2. Reusable for generalized tasks
3. Non-reusable for generalized tasks, as opposed to Imperative Programming
4. Reusable for generalized tasks, just like Imperative programming
5. What is the output of this code ?

Class Cat:

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

print(‘Cat object: Meow! ‘ )

cat\_obj = Cat()

Select the output

1. Cow object: Moo!
2. Cat object: Meow!
3. Dog object: Woof!
4. None
5. What is a “docstring” ?
6. It is an alternative name for Instance Variables of type String appearing in Classes
7. It is a synonym for code comments
8. It is a string literal specified in source code that is used to explain or document a specific segment of code
9. It is a Class Method that allows working with String documents.