Preguntas PCAP (Tanda 2)

Which of the following is the output of the below Python code?

```
list1 = [1, 3]
list2 = list1
list1[0] = 4
print(list2)
```

- ◯ **[1, 3]**
- ◯ **[4, 3]**
- ◯ **[1, 4]**
- ◯ **[1, 3, 4]**

- ○ [1, 3]
- ○ [4, 3] (Correcto)
- ○ [1, 4]
- ○ [1, 3, 4]

Which is the output of the following Python code fragment?

```
x = True
y=False
z= False

if not x or y:
print (1)
elif not x or not y and z:
print (2)
elif not x or y or not y and x:
print (3)
else:
print (4)
```

- ◯ **4**
- ◯ **2**
- ◯ **3**
- ◯ **None**

- ○ 4
- ○ 2
- ○ 3 (Correcto)
- ○ None

What will be the output of the following code?

```
class Test:
def __init__(self, s):
self.s = s

def print(self):
print(s)

a = Test("Python Class")
a.print()
```

- ◯ **The program gives an error because there is no constructor for class Test.**
- ◯ **Signature for the print method is incorrect, so an error is thrown.**
- ◯ **The correct output is .**
- ◯ **The above code will execute correctly on changing print(s) to print(self.s).**

- ○ The program gives an error because there is no constructor for class Test.
- ○ Signature for the print method is incorrect, so an error is thrown.
- ○ The correct output is .
- ○ The above code will execute correctly on changing print(s) to print(self.s). (Correcto)

What will be the output of the following code?

```
class Test:
    def __init__(self, s):
        self.s = s

    def print(self):
        print(self.s)

msg = Test()
msg.print()
```

- ◯ **The program has an error because class Test does not have a constructor.**
- ◯ **The above code produces an error because the definition of print(s) does not include .**
- ◯ **It executes successfully but prints nothing.**
- ◯ **The program has an error because of the constructor call is made without an argument.**

- ○ The program has an error because class Test does not have a constructor.
- ○ The above code produces an error because the definition of print(s) does not include .
- ○ It executes successfully but prints nothing.
- ○ The program has an error because of the constructor call is made without an argument. (Correcto)

What will be the output of the following code?

```
class Test:
    def __init__(self, s = "Welcome"):
        self.s = s

    def print(self):
        print(self.s)

msg = Test()
msg.print()
```

- ◯ **The program has an error because the constructor is not present in class Test.**
- ◯ **The above code produces an error because the definition of print(s) does not contain .**
- ◯ **It executes successfully but prints nothing.**
- ◯ **The program has an error because of the constructor call is made without an argument.**
- ◯ **The program executes successfully and prints Welcome.**

- ○ The program has an error because the constructor is not present in class Test.
- ○ The above code produces an error because the definition of print(s) does not contain .
- ○ It executes successfully but prints nothing.
- ○ The program has an error because of the constructor call is made without an argument.
- ○ The program executes successfully and prints Welcome. (Correcto)

What will be the output of the following code snippet?

```
class Test:
    def __init__(self):
        self.x = 1
        self.__y = 1

    def getY(self):
        return self.__y

val = Test()
print(val.x)
```

- ◯ **The program has an error because x is private and cannot be accessed outside of the class.**
- ◯ **The program has an error because you cannot name a variable using .**
- ◯ **The program runs fine and prints 1.**
- ◯ **The program runs fine and prints nothing.**

- ○ The program has an error because x is private and cannot be accessed outside of the class.
- ○ The program has an error because you cannot name a variable using .
- ○ The program runs fine and prints 1. (Correcto)
- ○ The program runs fine and prints nothing.

What will be the output of the following code snippet?

```
class Test:
    def __init__(self):
        self.x = 1
        self.__y = 1

    def getY(self):
        return self.__y

val = Test()
print(val.__y)
```

- ◯ **The program has an error because y is private and should not access it from outside the class.**
- ◯ **The program has an error because you cannot name a variable using __y.**
- ◯ **The program runs fine and prints 1.**
- ◯ **The program runs fine and prints nothing**

- ◯ The program has an error because y is private and should not access it from outside the class. (Correcto)
- ◯ The program has an error because you cannot name a variable using __y.
- ◯ The program runs fine and prints 1.
- ◯ The program runs fine and prints nothing

What will be the output of the following code snippet?

```
class Test:
    def __init__(self):
        self.x = 1
        self.__y = 1

    def getY(self):
        return self.__y

val = Test()
val.x = 45
print(val.x)
```

- ◯ **The program has an error because x is private and should not access it from outside the class.**
- ◯ **The program has an error because you cannot name a variable using __y.**
- ◯ **The program runs fine and prints 1.**
- ◯ **The program runs fine and prints 45.**

- ○ The program has an error because x is private and should not access it from outside the class.
- ○ The program has an error because you cannot name a variable using __y.
- ○ The program runs fine and prints 1.
- ○ The program runs fine and prints 45. (Correcto)

Which of the following is a private data field in the given code snippet?

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

- ○ __a
- ○ __b
- ○ __c__
- ○ __d__

- ○ __a
- ○ __b (Correcto)
- ○ __c__
- ○ __d__

What will be the output of the following code snippet?

```
class Test:
    def __init__(self):
        self.x = 1
        self.__y = 1

    def getY(self):
        return self.__y

val= Test()
val.__y = 45
print(val.getY())
```

- ◯ **The program has an error because y is private and should not access it from outside the class.**
- ◯ **The program has an error because you cannot name a variable using __y.**
- ◯ **The code runs fine and prints 1.**
- ◯ **The code executes successfully and prints 45.**

- ◯ The program has an error because y is private and should not access it from outside the class. (Correcto)
- ◯ The program has an error because you cannot name a variable using __y.
- ◯ The code runs fine and prints 1.
- ◯ The code executes successfully and prints 45.

What will be the output of the following code snippet?

```
def main():
    myCounter = Counter()
    num = 0

    for i in range(0, 100):
        increment(myCounter, num)

    print("myCounter.counter =", myCounter.counter, ", number of times =", num)

def increment(c, num):
    c.counter += 1
    num += 1

class Counter:
    def __init__(self):
        self.counter = 0

main()
```

- ◯ **counter is 101 , number of times is 0**
- ◯ **counter is 100, number of times is 0**
- ◯ **counter is 100, number of times is 100**
- ◯ **counter is 101, number of times is 101**

- ○ counter is 101 , number of times is 0
- ○ counter is 100, number of times is 0 (Correcto)
- ○ counter is 100, number of times is 100
- ○ counter is 101, number of times is 101

What code can we put at the third line of the definition of class B to invoke its superclass's constructor?

```
class A:
    def __init__(self, i = 1):
        self.i = i

class B(A):
    def __init__(self, j = 2):
        _____________________
        self.j = j

def main():
    b = B()
    print(b.i, b.j)

main()
```

- ◯ **super().__init__(self)**
- ◯ **super().__init__()**
- ◯ **A.__init__()**
- ◯ **A.__init__(self)**
- ◯ **B and D**

- ○ super().__init__(self)
- ○ super().__init__()
- ○ A.__init__()
- ○ A.__init__(self)
- ○ B and D (Correcto)

What will be the output of the following code snippet?

```
class A:
    def __init__(self, x = 1):
        self.x = x

class B(A):
    def __init__(self, y = 2):
        super().__init__()
        self.y = y

def main():
    b = B()
    print(b.x, b.y)

main()
```

- ◯ **0 0**
- ◯ **0 1**
- ◯ **1 2**
- ◯ **0 2**

- ○ 00
- ○ 01
- ○ 12 (Correcto)
- ○ 02

What will be the output of the following code snippet?

```
class A:
    def __init__(self):
        self.__x = 1
        self.y = 10

    def print(self):
        print(self.__x, self.y)

class B(A):
    def __init__(self):
        super().__init__()
        self.__x = 2
        self.y = 20

c = B()
c.print()
```

- ◯ **1 10**
- ◯ **1 20**
- ◯ **2 10**
- ◯ **2 20**

- ○ 1 10
- ○ 1 20 (Correcto)
- ○ 2 10
- ○ 2 20

What will be the output of the following code snippet?

```
class A:
    def __init__(self, x = 0):
        self.x = x

    def func1(self):
        self.x += 1

class B(A):
    def __init__(self, y = 0):
       A.__init__(self, 3)
        self.y = y

    def func1(self):
        self.y += 1

def main():
    b = B()
    b.func1()
    print(b.x, b.y)

main()
```

- ○ 20
- ○ 31
- ○ 40
- ○ 30

- ○ 20
- ○ 31 (Correcto)
- ○ 40
- ○ 30

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

```
class A:
    def __new__(self):
        self.__init__(self)
        print("A's __new__() invoked")

    def __init__(self):
        print("A's __init__() invoked")

class B(A):
    def __new__(self):
        print("B's __new__() invoked")

    def __init__(self):
        print("B's __init__() invoked")

def main():
    b = B()
    a = A()

main()
```

- ○ **B's __new__() invoked A's __init__() invoked**
- ○ **B's __new__() invoked A's __new__() invoked**
- ○ **B's __new__() invoked A's __init__() invoked A's __new__() invoked**
- ○ **A's __init__() invoked A's __new__() invoked**

- ○ B's __new__() invoked A's __init__() invoked
- ○ B's __new__() invoked A's __new__() invoked
- ○ B's __new__() invoked A's __init__() invoked A's __new__() invoked (Correcto)
- ○ A's __init__() invoked A's __new__() invoked

What will be the output of the following code snippet?

```
class A:
    def __init__(self, num):
        self.x = num

class B(A):
    def __init__(self, num):
        self.y = num

obj = B(11)
print ("%d %d" % (obj.x, obj.y))
```

- ◯ **None 11**
- ◯ **11 None**
- ◯ **11 11**
- ◯ **AttributeError: 'B' object has no attribute 'x'**

- ○ None 11
- ○ 11 None
- ○ 11 11
- ○ AttributeError: 'B' object has no attribute 'x' (Correcto)

What will be the output of the following code snippet?

```
class A:
    def __init__(self):
        self.x = 1

    def func(self):
        self.x = 10

class B(A):
    def func(self):
        self.x += 1
        return self.x

def main():
    b = B()
    print(b.func())

main()
```

- ○ **1**
- ○ **2**
- ○ **10**
- ○ **x is not accessible from the object of classB.**

- ○ 1
- ○ 2 (Correcto)
- ○ 10
- ○ x is not accessible from the object of classB.

What will be the output of the following code snippet?

```
class A:
    def __str__(self):
        return "A"

class B(A):
    def __init__(self):
        super().__init__()

class C(B):
    def __init__(self):
        super().__init__()

def main():
    b = B()
    a = A()
    c = C()
    print(a, b, c)

main()
```

- ○ **BBB**
- ○ **ABC**
- ○ **CBA**
- ○ **AAA**

- ○ BBB
- ○ ABC
- ○ CBA
- ○ AAA (Correcto)

What will be the output of the following code snippet?

```
class A:
    def __init__(self, x = 2, y = 3):
        self.x = x
        self.y = y

    def __str__(self):
        return "A"

    def __eq__(self, num ):
        return self.x * self.y == num.x * num.y

def main():
    a = A(1, 2)
    b = A(2, 1)
    print(a == b)

main()
```

- ◯ **True**
- ◯ **False**
- ◯ **2**
- ◯ **1**

- ○ True (Correcto)
- ○ False
- ○ 2
- ○ 1