Homework 02

### 1. A data structure described as LIFO is actually a:

1. **list**
2. **heap**
3. **tree**
4. **stack**

### 2. If the class’s constructor is declared as below, which one of the assignments is valid?

class Class:

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

pass

1. **object = Class(object)**
2. **object = Class(self)**
3. **object = Class**
4. **object = Class()**

### 3. If there is a superclass named A and a subclass named B, which one of the presented invocations should you put instead of a comment?

class A:

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

self.a = 1

class B:

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

# put selected line here

self.a = 2

1. **\_\_init\_\_()**
2. **A.\_\_init\_\_()**
3. **A.\_\_init\_\_(self)**
4. **A.\_\_init\_\_(1)**

### 4. What will be the effect of running the following code?

class A:

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

self.\_\_a = v + 1

a = A(0)

print(a.\_\_a)

1. **it will print 0**
2. **it will print 2**
3. **it will print 1**
4. **it will raise an AttributeError exception**

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

class A:

def \_\_init\_\_(self,v = 1):

self.v = v

def set(self,v):

self.v = v

return v

a = A()

print(a.set(a.v + 1))

1. **3**
2. **0**
3. **1**
4. **2**

### 6. What will be the output of the following code?

class A:

X = 0

def \_\_init\_\_(self,v = 0):

self.Y = v

A.X += v

a = A()

b = A(1)

c = A(2)

print(c.X)

1. **0**
2. **2**
3. **3**
4. **1**

### 7. What will be the output of the following code?

class A:

A = 1

print(hasattr(A,’A’))

1. **0**
2. **False**
3. **1**
4. **True**

### 8. What will be the result of executing the following code?

class A:

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

pass

a = A(1)

print(hasattr(a,’A’))

1. **it will print False**
2. **it will print 1**
3. **it will print True**
4. **it will raise an exception**

### 9. What will be the result of executing the following code?

class A:

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

return ‘a’

class B(A):

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

return ‘b’

class C(B):

pass

o = C()

print(o)

1. **it will raise an exception**
2. **it will print a**
3. **it will print c**
4. **it will print b**

### 10. What will be the result of executing the following code?

class A:

pass

class B(A):

pass

class C(B):

pass

print(issubclass(C,A))

1. **it will raise an exception**
2. **it will print True**
3. **it will print 1**
4. **it will print False**

### 11. What will be the result of executing the following code?

class A:

def a(self):

print(‘a’)

class B:

def a(self):

print(‘b’)

class C(B,A):

def c(self):

self.a()

o = C()

o.c()

1. **it will print c**
2. **it will print a**
3. **it will raise an exception**
4. **it will print b**

### 12. What will be the result of executing the following code?

class A:

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

return ‘a’

class B(A):

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

return ‘b’

class C(B):

pass

o = C()

print(o)

1. **it will print b**
2. **it will raise an exception**
3. **it will print a**
4. **it will print c**

### 13. What will be the result of executing the following code?

class A:

v = 2

class B(A):

v = 1

class C(B):

pass

o = C()

print(o.v)

1. **it will print an empty line**
2. **it will print 2**
3. **it will raise an exception**
4. **it will print 1**

### 14. What will be the result of executing the following code?

def f(x):

try:

x = x / x

except:

print(“a”,end=”)

else:

print(“b”,end=”)

finally:

print(“c”,end=”)

f(1)

f(0)

1. **it will print bcbc**
2. **it will print bcac**
3. **it will print acac**
4. **it will raise an unhandled exception ]**

### 15. What will be the result of executing the following code?

try:

raise Exception(1,2,3)

except Exception as e:

print(len(e.args))

1. **it will print 2**
2. **it will print 1**
3. **it will raise an unhandled exception**
4. **it will print 3**

### 16. What will be the result of executing the following code?

class Ex(Exception):

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

Exception.\_\_init\_\_(self,msg + msg)

self.args = (msg,)

try:

raise Ex(‘ex’)

except Ex as e:

print(e)

except Exception as e:

print(e)

1. **it will raise an unhandled exception**
2. **it will print an empty line**
3. **it will print exex**
4. **it will print ex**

### 17. What will be the result of executing the following code?

class I:

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

self.s = ‘abc’

self.i = 0

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

return self

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

if self.i == len(self.s):

raise StopIteration

v = self.s[self.i]

self.i += 1

return v

for x in I():

print(x,end=”)

1. **it will print cba**
2. **it will print 210**
3. **it will print 012**
4. **it will print abc**

### 18. What will be the result of executing the following code?

def I():

s = ‘abcdef’

for c in s[::2]:

yield c

for x in I():

print(x,end=”)

1. **it will print an empty line**
2. **it will print bdf**
3. **it will print abcdef**
4. **it will print ace**

### 19. What will be the result of executing the following code?

def I(n):

s = ‘+’

for i in range(n):

s += s

yield s

for x in I(2):

print(x,end=”)

1. **it will print +**
2. **it will print +++**
3. **it will print ++++++**
4. **it will print ++**

### 20. What will be the result of executing the following code?

def o(p):

def q():

return ‘\*’ \* p

return q

r = o(1)

s = o(2)

print(r() + s())

1. **it will print \*\*\***
2. **it will print \*\***
3. **it will print \*\*\*\***
4. **it will print \***

### 21. When a file is opened in read mode, it:

1. **it must exist (an exception will be raised otherwise)**
2. **it cannot exist (it has to be created every time)**
3. **it will be deleted if it exists**
4. **it doesn’t have to exist (it will be created if absent)**

### 22. If you want to open a text file in append mode, you would use the following mode string:

1. **t+a**
2. **at**
3. **a+t**
4. **at+**

### 23. The sys.stdin stream is normally associated with a:

1. **null device**
2. **keyboard**
3. **printer**
4. **screen**

### 24. The strerror function comes from the OS module, and it’s designed to:

1. **raise a string exception**
2. **translate an error description from one language to another**
3. **translate an error description into an error number**
4. **translate an error number into an error description**

### 25. If s is a stream opened in read mode, the following line

q = s.read(1)

will:

1. **read 1 buffer from the stream**
2. **read 1 kilobyte from the stream**
3. **read 1 character from the stream**
4. **read 1 line from the stream**

### 26. How does the readline() method react when the end‑of‑file occurs?

1. **it returns eof**
2. **it returns ‑1**
3. **it returns an empty string**
4. **it raises an exception**

### 27. The readlines() method returns a:

1. **list**
2. **dictionary**
3. **tuple**
4. **string**

### 28. Assuming that the open() invocation has gone successfully, the following snippet will:

for x in open(‘file’,’rt’):

print(x)

1. **read the file line by line**
2. **read the file character by character**
3. **cause an exception**
4. **read the whole file at once**

### 29. The byte array class can create objects which are designed to:

1. **build arrays 1 byte in size**
2. **convert tuples into lists**
3. **convert lists into tuples**
4. **store amorphic data organized in bytes**

### 30. If you want to fill a byte array with data read in from a stream, you use the:

1. **read() method**
2. **readinto() method**
3. **readfrom() method**
4. **readbytes() method**