**1 => ++++++**

def fun(n):

s='+'

for i in range(n):

s+=s

yield s

for x in fun(2):

print(x,end='')

**2 => Atrribute Error**

class A:

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

self.\_\_a=v+1

a=A(0)

print(a.\_\_a)

**3 => Object = Class(self)**

class Class:

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

pass

**4=> 3**

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)

**5=> abc**

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='')

**6=> b**

class A:

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

return 'a'

class B(A):

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

return 'b'

class C(B):

pass

o=C()

print(o)

**7=> \*\*\***

def o(p):

def q():

return '\*' \* p

return q

r=o(1)

s=o(2)

print(r()+s())

**8=> b**

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()

**9=> a**

class A:

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

return 'a'

class B:

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

return 'b'

class C(A,B):

pass

o=C()

print(o)

**10 => 2**

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))

**11=> 1**

class A:

v=2

class B(A):

v=1

class C(B):

pass

o=C()

print(o.v)

**12=> True**

class A:

A=1

print(hasattr(A,'A'))

**13=> True**

class A:

pass

class B(A):

pass

class C(B):

pass

print(issubclass(C,A))

**14=> LIFO = Stack**

**15=> ace**

def I():

s='abcdef'

for c in s[::2]:

yield c

for x in I():

print(x,end='')

**16=> TypeError**

class A:

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

pass

a=A(1)

print(hasattr(a,'A'))

**17=> Unhandled exception**

class Ex(Exception):

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

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

try:

raise Ex('ex')

except Ex as e:

print(e)

except Exception as e:

print(e)

**18=> bcac**

def f(x):

try:

x=x/x

except:

print("a",end='')

else:

print("b",end='')

finally:

print("c",end='')

f(1)

f(0)

**19=> A.\_\_init\_\_(self)**

class A:

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

self.a=1

class B(A):

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

A.\_\_init\_\_()

self.b=2

print(issubclass(B,A))

**20=> 3**

try:

raise Exception(1,2,3)

except Exception as e:

print(len(e.args))