

ASSIGNMENT-8

# PYTHON

# **NAME : MADA SRAGVIN KUMAR**

# **MIS NO : 112315097**

# **GROUP : 3**

**YEAR : 2**

**SECTION : A**

# **1:**

def memoize\_fun():

cache={}

def closure(a):

if a in cache:

return cache[a]

if a==1 or a==0:

k=1

else:

k=closure(a-1)\*a

cache[a]=k

return k

return closure

fact=memoize\_fun()

print(fact(5))

print(fact(5))



# **2:**

def create\_pipeline(a):

def pipe(b):

k=b

for i in a:

k=i(k)

return k

return pipe

def mul(x):

return x\*2

def add(x):

return x+3

pipeline=create\_pipeline([mul,add])

print(pipeline(5))



# **3:**

def multiply(a):

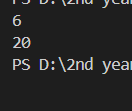
def multiple(b):

return b\*a

return multiple

print(multiply(2)(3))

print(multiply(4)(5))



# **4:**

from functools import reduce

students = [

{'name': 'Alice', 'score': 45},

{'name': 'Bob', 'score': 55},

{'name': 'Charlie', 'score': 65},

{'name': 'David', 'score': 75}

]

def passed(student):

return student['score'] >= 50

def name(student):

return student['name']

def total(x, student):

return x + student['score']

a = filter(passed, students)

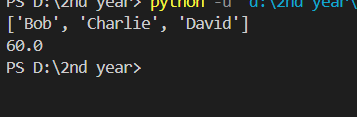
b= map(name, a)

print( list(b))

c = reduce(total, students, 0)

avg = c/ len(students)

print(avg)



# **5:**

def bank\_account(balance):

def deposit(amount):

nonlocal balance

balance=balance+amount

print(f"Deposited:{amount},New Balance:{balance}")

def withdraw(amount):

nonlocal balance

if(amount>balance):

print("Insufficient funds!")

else:

balance=balance-amount

print(f"Withdrew:{amount},New Balance:{balance}")

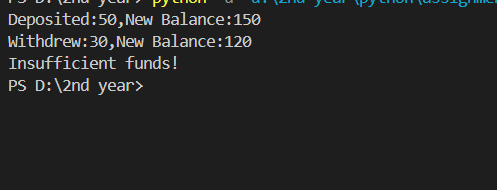
return deposit,withdraw

account=bank\_account(100)

account[0](50)

account[1](30)

account[1](200)

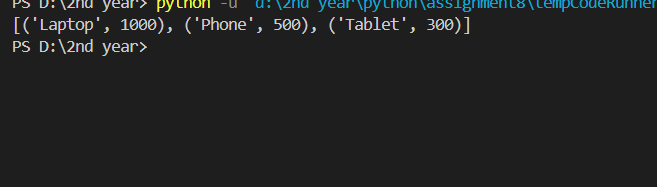


# **6:**

products = [('Laptop', 1000), ('Phone', 500), ('Tablet', 300)]

sorted\_products = sorted(products, key=lambda x: x[1], reverse=True)

print(sorted\_products)



# **7:**

from functools import partial

power\_of\_two=partial(pow,exp=2)

print(power\_of\_two(5))



# **8:**

def fun(\*args):

def pol(x):

k,j=0,len(args)-1

for i in args:

k=k+pow(x,j)\*i

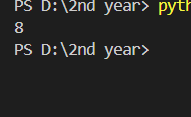
j-=1

return k

return pol

p=fun(3,0,-4)

print(p(2))



# **9:**

a=[1,2,3,46,7]

k=list(filter(lambda x:x%2==0,a))

print(k)



# **10:**

def sqaure(x):

return x\*x

def even(y):

return y%2==0

list1=[1,2,3,4,5,6,7,8,9,10]

a=map(sqaure,list1)

b=filter(even,a)

c=list(b)

print(c)

