# Very Easy

#Task\_01

# (a)

def factorial(number):

if number==0 or number==1:

return 1

else:

return number\*factorial(number-1)

print(factorial(5))

Output:

120

# (b)

def fibonacci(number):

if number==0 or number==1:

return number

else:

return fibonacci(number-1)+fibonacci(number-2)

print(fibonacci(5))

Output:

5

# (c)

def Array(x,index):

if index<len(x):

print(x[index])

Array(x,index+1)

x=[1,2,3,4,5]

Array(x,0)

Output:

1

2

3

4

5

# (d)

def powerN(base,n):

if base>0:

return base\*\*n

print(powerN(3,2))

Output:

9

# Easy

#Task\_02

# (a)

def Binary(number):

if number>0:

Binary(number//2)

print(number%2,end="")

Binary(4)

Output:

100

class Node:

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

self.value=value

self.head=None

def printNode(self):

print(self.value)

class Linked\_list:

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

self.head = None

self.tail = None

self.size = 0

def add(self,a):

for i in a:

n = Node(i)

n.next = self.head

self.head = n

self.size = self.size+1

if self.size == 1:

self.tail = n

def showlist(self):

n = self.head

if n is None:

print("Empty list")

else:

while (n is not None):

n.printNode()

n = n.next

arr = [3,2,5,1,8]

a1 = Linked\_list()

a1.add(arr)

a1.showlist()

Output:

8

1

5

2

3

# Medium

#Task\_03

def hocBuilder(height):

count=0

#value=3

if height==0:

return 0

#value-=1

elif height==1:

return 8

return 5+hocBuilder(height-1)

count+=1

print(hocBuilder(3))

Output:

18

# Hard

#Task\_04

# (a)

def print\_n(num):

if (num == 0):

return

print\_n(num - 1)

print(num, end = " ")

def pattern(n, i):

if (n == 0):

return

print\_n(i)

print("\n", end = "")

pattern(n - 1, i + 1)

if \_\_name\_\_ == '\_\_main\_\_':

n = 5

pattern(n, 1)

Output:

1

1 2

1 2 3

1 2 3 4

1 2 3 4 5

# (b)

def print\_space(space):

if (space == 0):

return

print(" ", end=" ")

print\_space(space - 1)

def print\_number(num):

if (num == 0):

return

print\_number(num - 1)

print(num, end =" ")

def pattern(n, number):

if (n == 0):

return

print\_space(n - 1)

print\_number(number - n + 1)

print()

pattern(n - 1, number)

if \_\_name\_\_ == '\_\_main\_\_':

n = 5

pattern(n, n)

Output:

1

1 2

1 2 3

1 2 3 4

1 2 3 4 5

# Very Hard

#Task\_05

class FinalQ:

def print(self,array,idx):

if(idx<len(array)):

profit = self.calcProfit(array[idx])

print(str(idx+1) + ". Investment: " + str(array[idx]) + "; Profit: " + str(profit))

self.print(array, idx+1)

def calcProfit(self,investment):

#TO DO

if investment <= 25000:

return 0.0

else:

value = investment - 100000

value/= 100

value = value + value + value + value + value + value + value + value

return value + 3375.0

#Tester

array = [25000,100000,250000,350000]

f = FinalQ()

f.print(array, 0)

Output:

1. Investment: 25000; Profit: 0.0

2. Investment: 100000; Profit: 3375.0

3. Investment: 250000; Profit: 15375.0

4. Investment: 350000; Profit: 23375.0