1] Write a program to calculate Fibonacci numbers and find its step count.

1A] nterms = int(input("How many terms?"))

n1, n2 = 0,1

count= 0

if nterms <= 0:

print("Please enter a positive integer")

elif nterms == 1:

print("f Fibonacci sequence up to",nterms, ":")

print (n1)

else:

print("Fibonacci sequence:")

while count < nterms:

print(n1)

nth = n1+n2

n1 = n2

n2 = nth

count += 1

1B] def fibonacci(n):

if n <= 0:

return []

elif n == 1:

return [0]

elif n == 2:

return [0, 1]

else:

seq = fibonacci(n - 1)

seq. append(seq[-1] + seq[-2])

return seq

nterms = int(input("How many terms? "))

if nterms <= 0:

print("Please enter a positive integer" )

else:

print("Fibonacci sequence: ")

fib\_sequence = fibonacci (nterms)

for num in fib\_sequence:

print (num)

4] Write a program to solve a 0-1 Knapsack problem using dynamic programming or branch and bound strategy.

def knapsack\_01(n, values, weights, W):

dp = [[0] \* (W + 1) for \_ in range(n + 1)]

for i in range(n + 1):

for w in range(W + 1):

if i == 0 or w == 0:

dp[i][w] = 0

elif weights[i - 1] <= w:

dp[i][w] = max(dp[i - 1][w], dp[i - 1][w - weights[i - 1]] + values[i - 1])

else:

dp[i][w] = dp[i - 1][w]

selected\_items = []

i, w = n, W

while i > 0 and w > 0:

if dp[i][w] != dp[i - 1][w]:

selected\_items.append(i - 1)

w -= weights[i - 1]

i -= 1

return dp[n][W], selected\_items

n = int(input("Enter the number of items: "))

values = list(map(int, input("Enter the values of the items separated by space: ").split()))

weights = list(map(int, input("Enter the weights of the items separated by space: ").split()))

W = int(input("Enter the maximum capacity of the knapsack: "))

max\_value, selected\_items = knapsack\_01(n, values, weights, W)

print("Maximum value:", max\_value)

print("Selected items (0-indexed):", selected\_items)