PART B

|  |  |
| --- | --- |
| Roll No. | Name: |
| Class : | Batch : |
| Date of Experiment: | Date of Submission |
| Grade : | Time of Submission: |
| Date of Grading: |  |

**B.1 Software Code written by student:**

# Name: Naman Garg

# Roll No: B032

# Aim of Experiment: Implementation of Backtracking Algorithm Design. Write a program to find solution for Sum of subs Problem.

# k: index of the weight being considered

# r: sum of remaining weights

# s: current sum

def SUM\_OF\_SUBS(s, k, r):

    # Generating the left child

    arr[k] = 1

    if s+w[k] == m:

        # Subset is found

        listOfSubsets = []

        for i in range(n):

            if arr[i] == 1:

                listOfSubsets.append(w[i])

        subs.append(listOfSubsets)

    elif s+w[k]+w[k+1] <= m:

        # Bounding function

        SUM\_OF\_SUBS(s+w[k], k+1, r-w[k])

    # Generate right child

    arr[k] = 0

    if (s+r-w[k] >= m) and (s+w[k+1] <= m):

        # Bounding function

        SUM\_OF\_SUBS(s, k+1, r-w[k])

# Driver program to test above function

n = int(input('Enter number of weights (n): '))

w = list(map(int, input('Enter the weights (wi): ').split()))

m = int(input('Enter the required sum (m): '))

w.sort()    # Sorting just in case user did not enter the values in ascending order

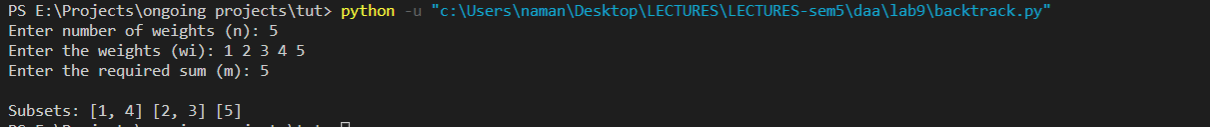
arr = [0]\*n    # List to keep track of the numbers selected from the subset

subs = []

SUM\_OF\_SUBS(0, 0, sum(w))

print('\nsubsets--->>>', \*subs)

**B.2 Input and Output:**

**B.3 Observations and learning:**

***(Students are expected to comment on the output obtained with clear observations and learning for each task/ sub part assigned)***

**B.4 Conclusion:**

*(****Students must write the conclusion as per the attainment of individual outcome listed above and learning/observation noted in section B.3)***

**B.5 Question of Curiosity**

***(To be answered by student based on the practical performed and learning/observations)***

Q.1 Identify & discuss the real life applications Sum of Subsets Problem.

Q.2 Compare different techniques of algorithm design - Divide & Conquer, Greedy, Dynamic Programming and Backtracking.

Q.3 List different problems that can be solved by backtracking technique of algorithm design..

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*