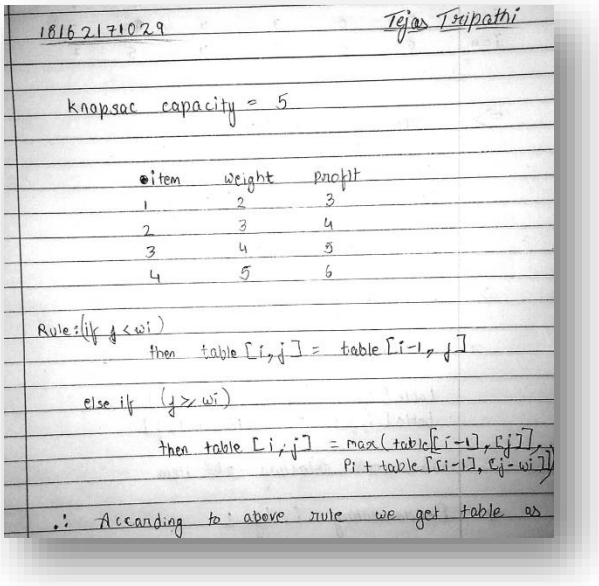
Tejas Tripathi

Sub: Algorithm Analysis and Design

You are a thief carrying a single knapsack with limited (W = 5) capacity. The museum you stole had (n = 4) artifact that you could steal. Unfortunately you might not be able to steal the entire artifact because of your limited knapsack capacity.

You have to cherry pick the artifact in order to maximize the total value (<=W) of the artifacts you stole.

- First solve the given below example:
  Let n = 4, W=5
  (P1, P2, P3, P4) = (3,4,5,6)
  (w1, w2,w3,w4) = (2,3,4,5)



Tejas Tripathi

	j=0	j=1	j=2	j=3	j=4	j=5
i=0	0	0	0	0	0	0
i=1	0	0	3	3	3	3
i=2	0	0	3	4	4	7
i=3	0	0	3	4	5	7
i=4	0	0	3	4	5	7

⇒ Starting from i=4, j=5

table[4, 5] = table[3, 5]
table[i, j] = table[i-1, j]

∴ not selecting 4th item

⇒ Decrementing i by 1

i = i-1 = 4-1 = 3

table[3, 5] = table[2, 5]
table[i, j] = table[i-1, j]

∴ not selecting 3rd item

⇒ Decrementing i by 1

i = i-1 = 3-1 = 2

table[2, 5] ≠ table[1, 5]

∴ selecting 2nd item

⇒ Now setting $i = i-1$ and $W = W - wi$ (Here $wi = 3$)

$\therefore i = 2-1$ and $W = 5-3$

$\therefore i = 1 \quad W = 2$

$\therefore i = 1 \quad j = 2$

table $[i,j] \neq$ table $[i-1, j]$

table $[1,2] \neq$ table $[0,2]$

$\therefore$ selecting 1st item

⇒ Now setting $i = i-1 = 1-1 = 0$ $\therefore$ not possible

$\therefore$ item 1 and item 2 will get selected

$\therefore$ profit of 1st item and 2nd item =

$3 + 4 = 7$

CODE:

```
knap_sac_capacity=int(input('Enter Knapsak capacity: '))
weights=[]
profits=[]
profits.append(0)
weights.append(0)
we=[int(i) for i in input('Enter the weights: ').split(' ')]
for i in we:
    weights.append(i)
pr=[int(i) for i in input('Enter the Profits: ').split(' ')]
for i in pr:
    profits.append(i)

matrix=[]
for i in range(len(weights)):
    matr=[]
    for j in range(knap_sac_capacity+1):
        matr.append(0)
    matrix.append(matr)

if(knap_sac_capacity>=weights[0]):

    for i in range(len(weights)):
        w=weights[i]
        for j in range(knap_sac_capacity+1):
            if i == 0 or j == 0 :
                matrix[i][j]=0
            elif(j<w):
              matrix[i][j]=matrix[i-1][j]
```

```
            else:
              matrix[i][j]=max(matrix[i-1][j], profits[i]+matrix[i-1][j-w])

print('\n')
for i in matrix:
   print(*i)

row=len(matrix)-1
column=knap_sac_capacity

list_weights=[]
while(row>-1 and column>-1):

  if(row==0 and column-weights[row]==0):
    list_weights.append(weights[row])
    break
  elif(column==0):

    break
  elif(matrix[row][column]==matrix[row-1][column]):
    row=row-1
  else:
    list_weights.append(weights[row])
    column=column-weights[row]

list_weights.sort()
for i in range(len(list_weights)):
   list_weights[i]=list_weights[i]-1
print("\nThere items selected will be:",*list_weights)

to=0
```

```
    for i in list_weights:
        to+=profits[i]


    print(f'Therefore Total Profit will be:{to}')
```

OUTPUT:

```
In [21]: runfile('C:/Users/Admin/study
material/sem5/Practicals/Algorithms/
Practical-7/knap_sac.py', wdir='C:/Users/Admin/
study material/sem5/Practicals/Algorithms/
Practical-7')

Enter Knapsak capacity: 5

Enter the weights: 2 3 4 5

Enter the Profits: 3 4 5 6


0 0 0 0 0 0
0 0 3 3 3 3
0 0 3 4 4 7
0 0 3 4 5 7
0 0 3 4 5 7

There items selected will be: 1 2
Therefore Total Profit will be:7

In [22]:
```