COMPETITIVE PROGRAMMING

Assignment-06

2303A51463

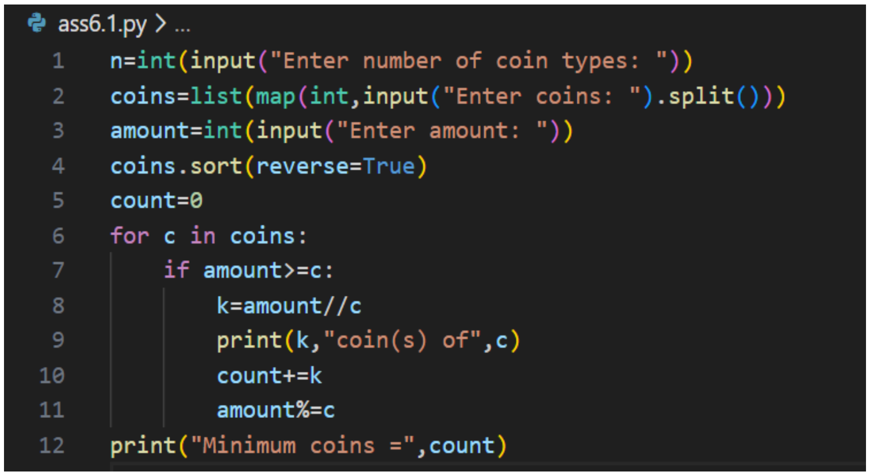
B-07

Greedy Approach

1.Coin Change Problem

Algorithm:

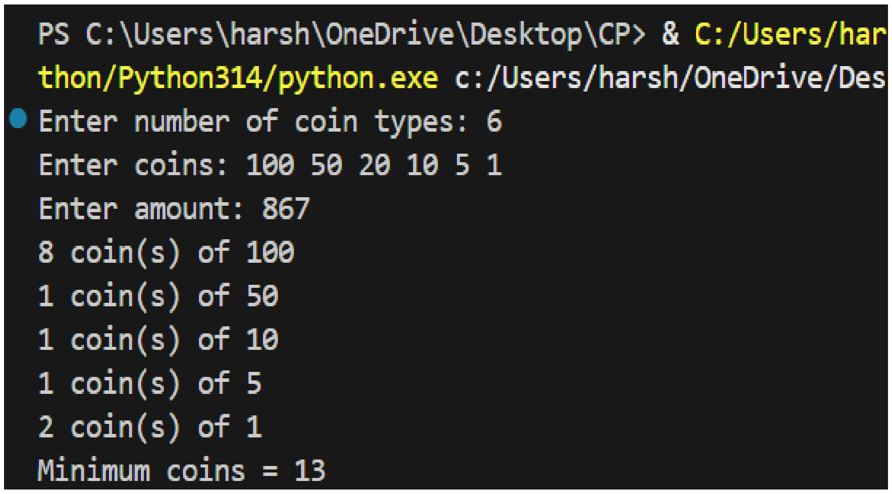
```
1. Start
2. Read number of coin types n
3. Read coin denominations
4. Read amount A
5. Sort coin denominations in descending order
6. Set count = 0
7. For each coin c
    a. If A ≥ c
        i. k = A // c
        ii. Print k coins of c
        iii. count = count + k
        iv. A = A % c
8. Print minimum coins
9. Stop
```



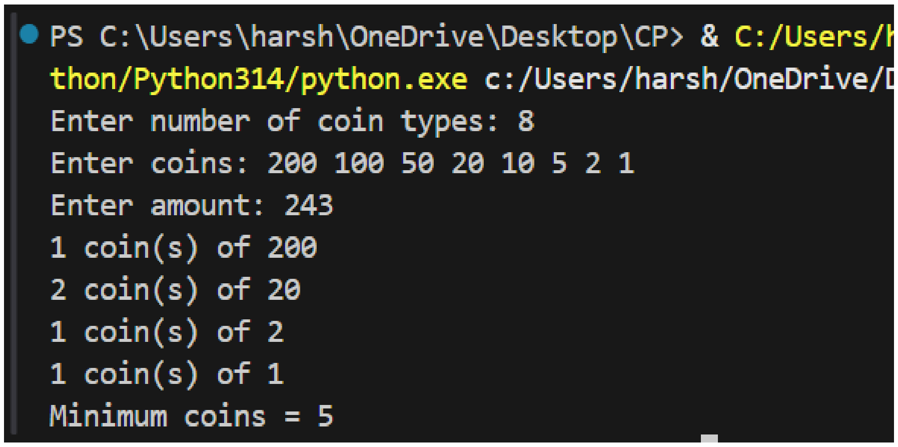
ass6.1.py > ...

```python
n=int(input("Enter number of coin types: "))
coins=list(map(int,input("Enter coins: ").split()))
amount=int(input("Enter amount: "))
coins.sort(reverse=True)
count=0
for c in coins:
    if amount>=c:
        k=amount//c
        print(k,"coin(s) of",c)
        count+=k
        amount%=c
print("Minimum coins =",count)
```

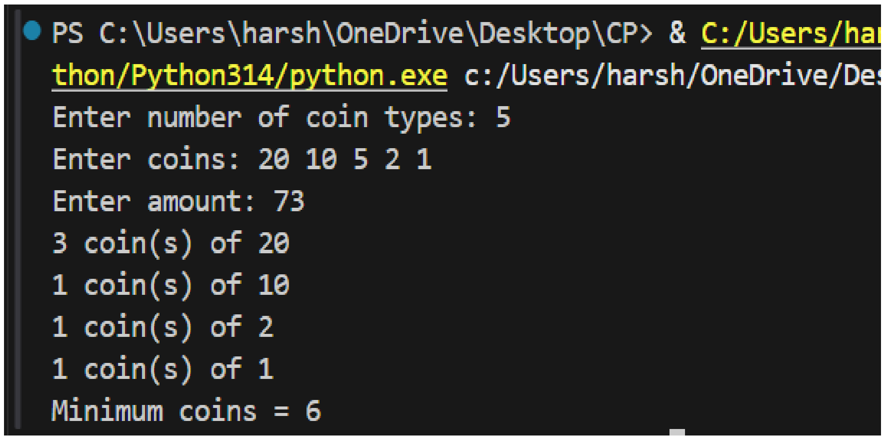
Output (Temple Donation Counter):


```
PS C:\Users\harsh\OneDrive\Desktop\CP> & C:/Users/har
thon/Python314/python.exe c:/Users/harsh/OneDrive/Des
Enter number of coin types: 6
Enter coins: 100 50 20 10 5 1
Enter amount: 867
8 coin(s) of 100
1 coin(s) of 50
1 coin(s) of 10
1 coin(s) of 5
2 coin(s) of 1
Minimum coins = 13
```

Output (Cinema Ticket Counter):


```
PS C:\Users\harsh\OneDrive\Desktop\CP> & C:/Users/h
thon/Python314/python.exe c:/Users/harsh/OneDrive/D
Enter number of coin types: 8
Enter coins: 200 100 50 20 10 5 2 1
Enter amount: 243
1 coin(s) of 200
2 coin(s) of 20
1 coin(s) of 2
1 coin(s) of 1
Minimum coins = 5
```

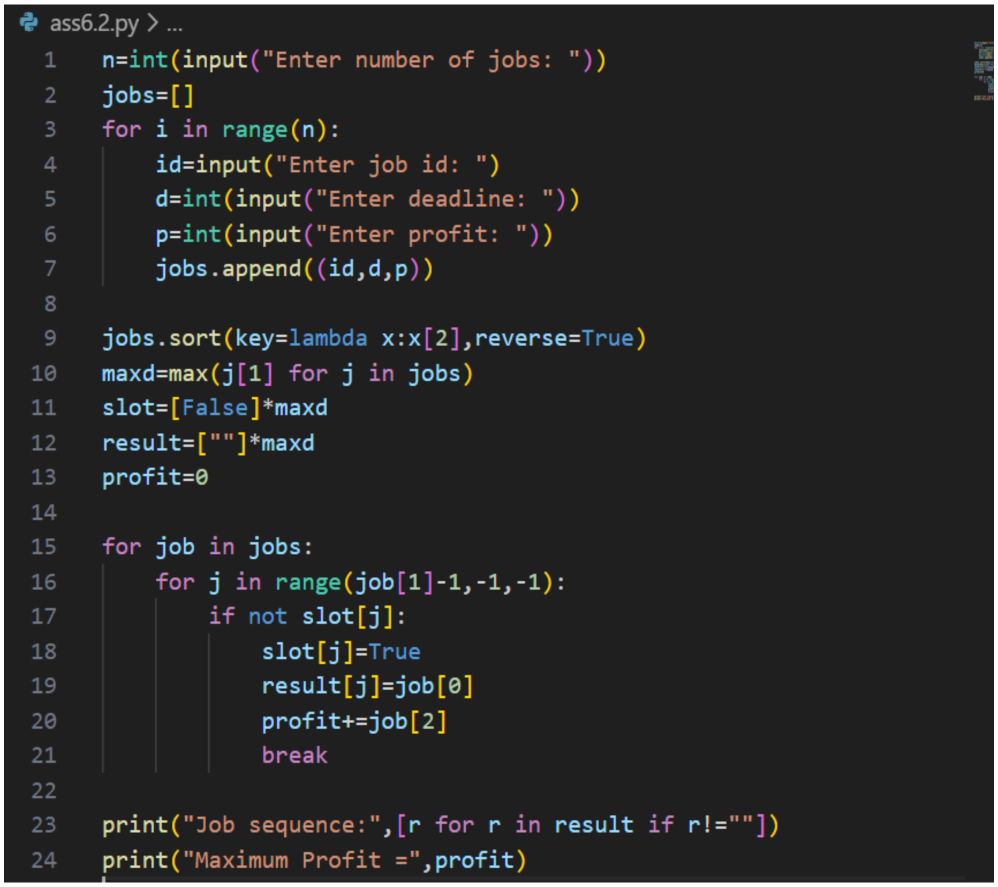
Output (Ice-Cream Shop):


```
PS C:\Users\harsh\OneDrive\Desktop\CP> & C:/Users/har
thon/Python314/python.exe c:/Users/harsh/OneDrive/Des
Enter number of coin types: 5
Enter coins: 20 10 5 2 1
Enter amount: 73
3 coin(s) of 20
1 coin(s) of 10
1 coin(s) of 2
1 coin(s) of 1
Minimum coins = 6
```

2.Job Sequencing with Deadlines – Story Scenarios

Algorithm:

```
1. Start
2. Read number of jobs n
3. Read job id, deadline, profit
4. Sort jobs in decreasing order of profit
5. Find maximum deadline maxD
6. Create maxD empty slots
7. Set profit = 0
8. For each job
    a. Check slots from deadline to 1
    b. If slot is free
        i. Assign job
        ii. Add job profit
9. Print job sequence and total profit
10. Stop
```



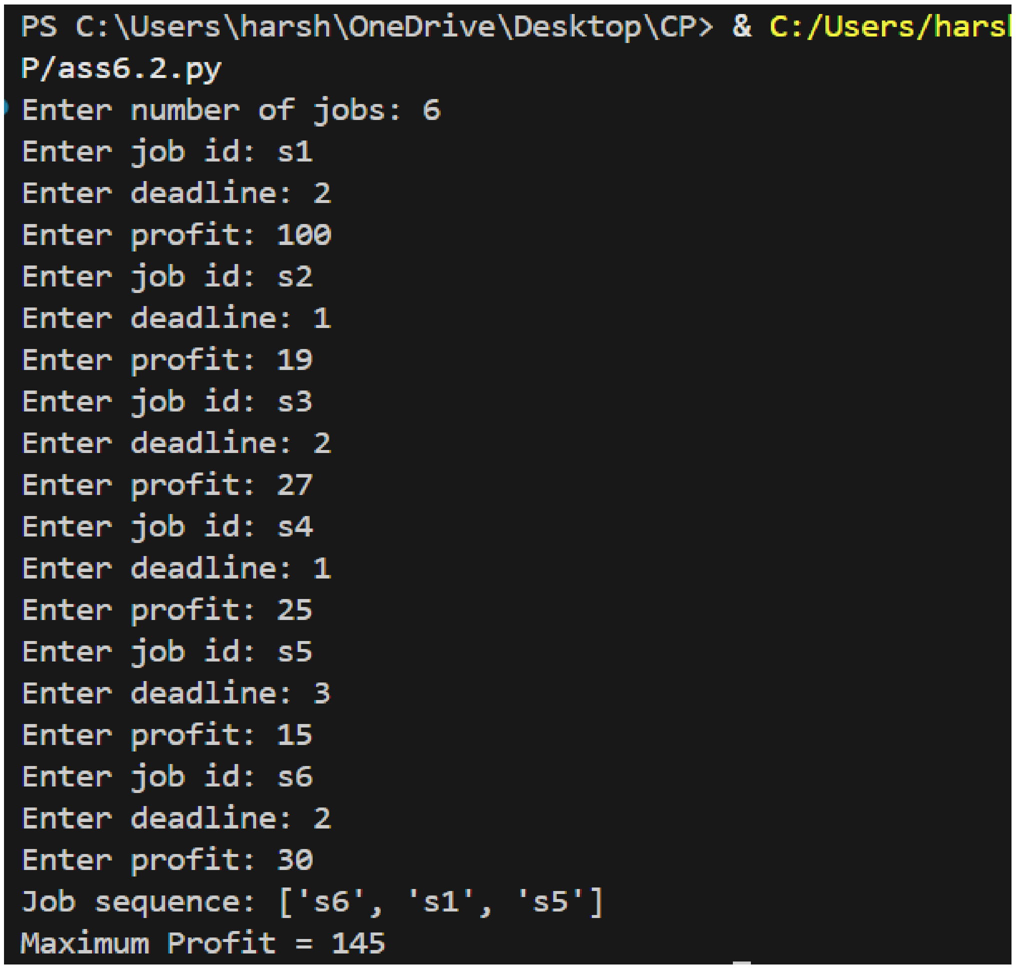
ass6.2.py > ...

```python
n=int(input("Enter number of jobs: "))
jobs=[]
for i in range(n):
    id=input("Enter job id: ")
    d=int(input("Enter deadline: "))
    p=int(input("Enter profit: "))
    jobs.append((id,d,p))

jobs.sort(key=lambda x:x[2],reverse=True)
maxd=max(j[1] for j in jobs)
slot=[False]*maxd
result=[""]*maxd
profit=0

for job in jobs:
    for j in range(job[1]-1,-1,-1):
        if not slot[j]:
            slot[j]=True
            result[j]=job[0]
            profit+=job[2]
            break

print("Job sequence:",[r for r in result if r!=""])
print("Maximum Profit =",profit)
```

Output (Hospital Operation Scheduling):

Assumed Input

Surgery	Deadline	Profit
S1	2	100
S2	1	19
S3	2	27
S4	1	25
S5	3	15
S6	2	30

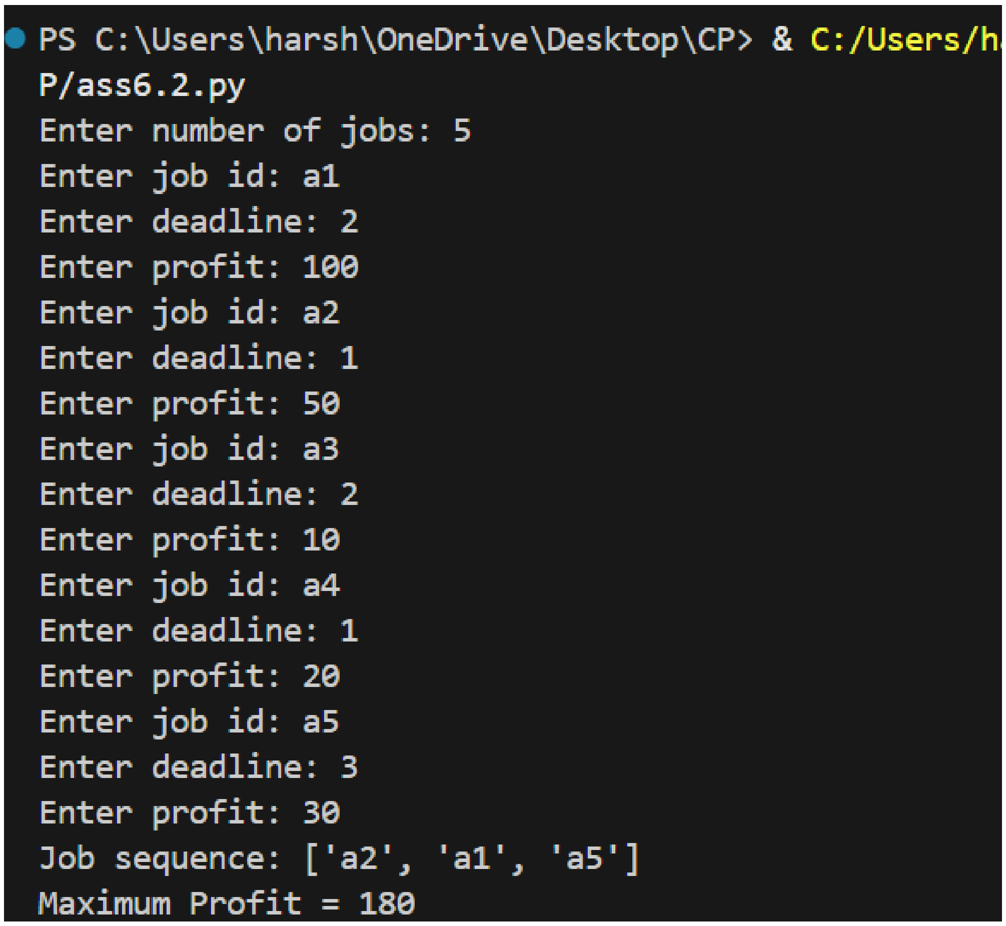


```
PS C:\Users\harsh\OneDrive\Desktop\CP> & C:/Users/hars
P/ass6.2.py
Enter number of jobs: 6
Enter job id: s1
Enter deadline: 2
Enter profit: 100
Enter job id: s2
Enter deadline: 1
Enter profit: 19
Enter job id: s3
Enter deadline: 2
Enter profit: 27
Enter job id: s4
Enter deadline: 1
Enter profit: 25
Enter job id: s5
Enter deadline: 3
Enter profit: 15
Enter job id: s6
Enter deadline: 2
Enter profit: 30
Job sequence: ['s6', 's1', 's5']
Maximum Profit = 145
```

Output (Newspaper Printing):

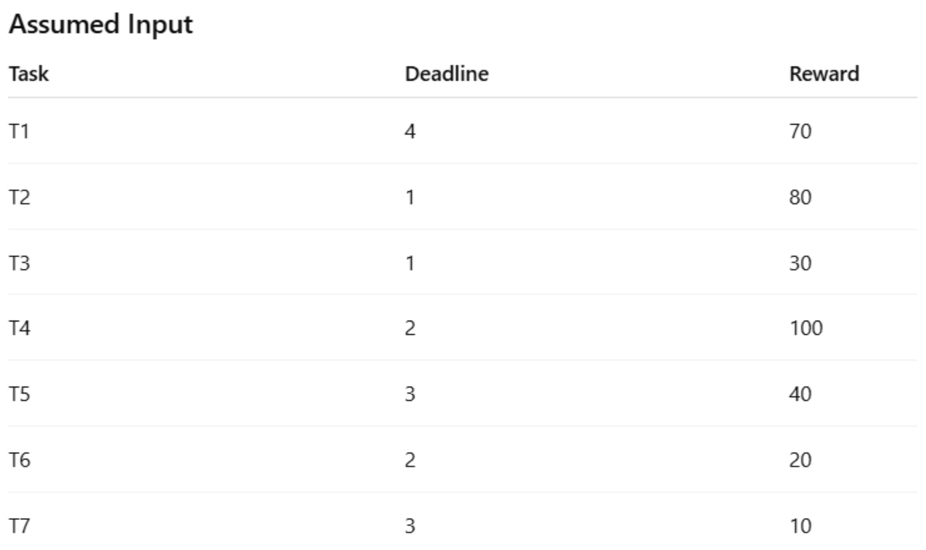
Assumed Input

Advertisement	Deadline	Payment
A1	2	100
A2	1	50
A3	2	10
A4	1	20
A5	3	30



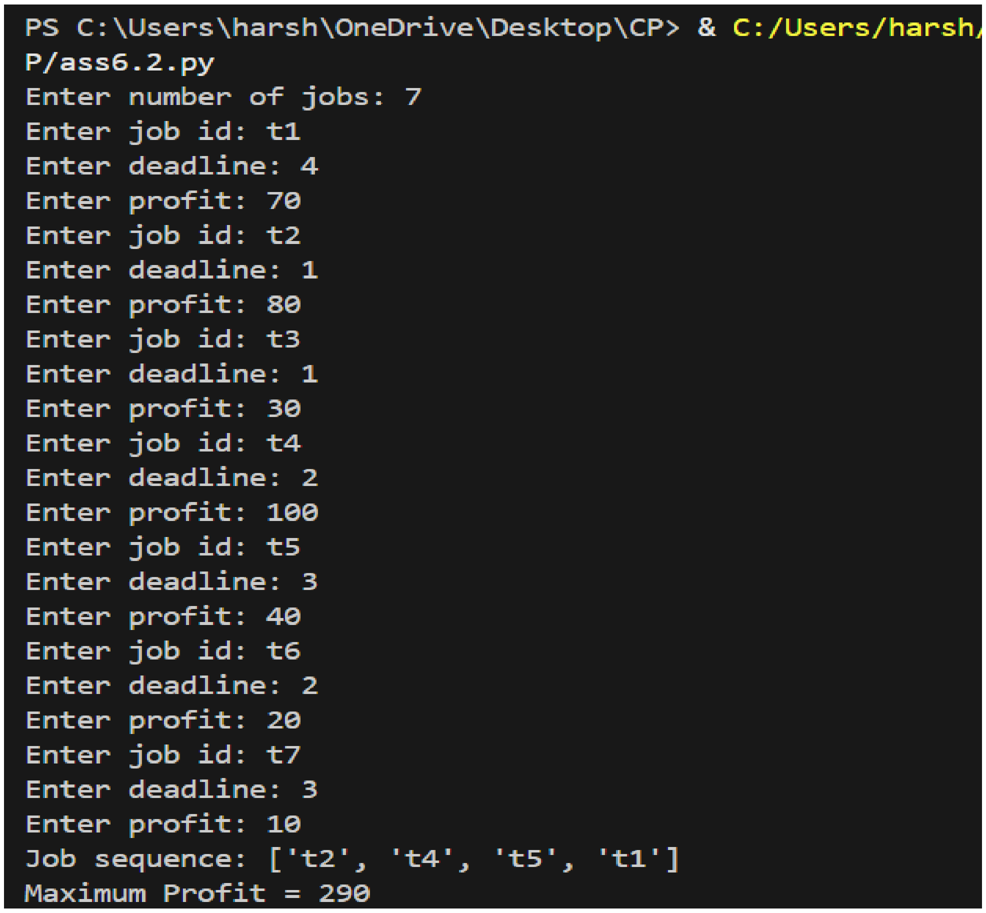
```
PS C:\Users\harsh\OneDrive\Desktop\CP> & C:/Users/h
P/ass6.2.py
Enter number of jobs: 5
Enter job id: a1
Enter deadline: 2
Enter profit: 100
Enter job id: a2
Enter deadline: 1
Enter profit: 50
Enter job id: a3
Enter deadline: 2
Enter profit: 10
Enter job id: a4
Enter deadline: 1
Enter profit: 20
Enter job id: a5
Enter deadline: 3
Enter profit: 30
Job sequence: ['a2', 'a1', 'a5']
Maximum Profit = 180
```

Output (Online Course Grading):



Assumed Input

Task	Deadline	Reward
T1	4	70
T2	1	80
T3	1	30
T4	2	100
T5	3	40
T6	2	20
T7	3	10

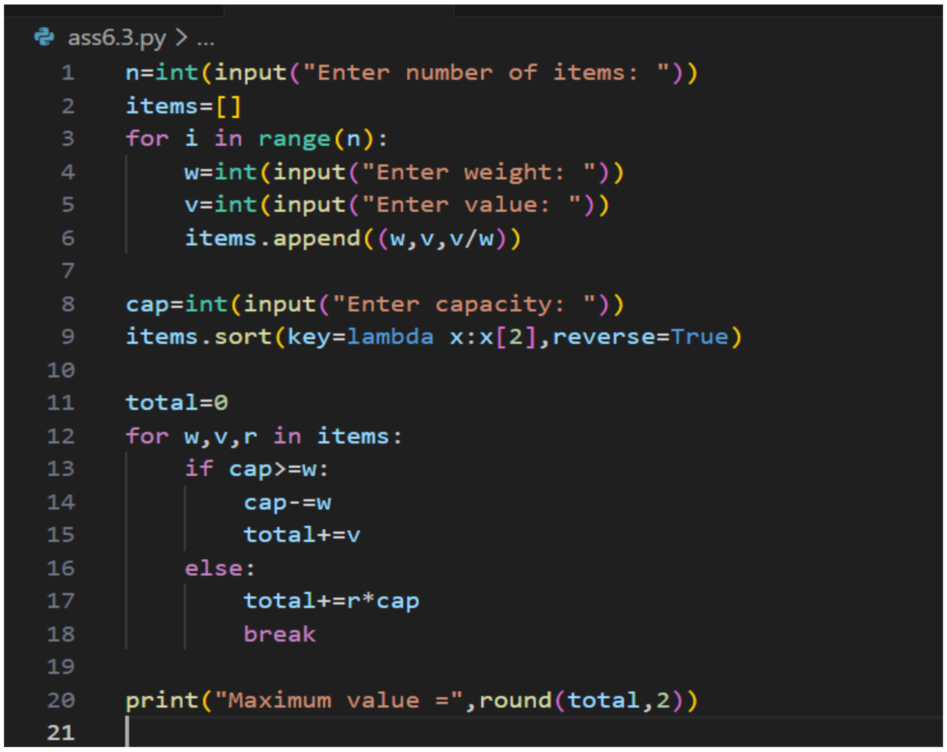


```
PS C:\Users\harsh\OneDrive\Desktop\CP> & C:/Users/harsh/
P/ass6.2.py
Enter number of jobs: 7
Enter job id: t1
Enter deadline: 4
Enter profit: 70
Enter job id: t2
Enter deadline: 1
Enter profit: 80
Enter job id: t3
Enter deadline: 1
Enter profit: 30
Enter job id: t4
Enter deadline: 2
Enter profit: 100
Enter job id: t5
Enter deadline: 3
Enter profit: 40
Enter job id: t6
Enter deadline: 2
Enter profit: 20
Enter job id: t7
Enter deadline: 3
Enter profit: 10
Job sequence: ['t2', 't4', 't5', 't1']
Maximum Profit = 290
```

3.Fractional Knapsack

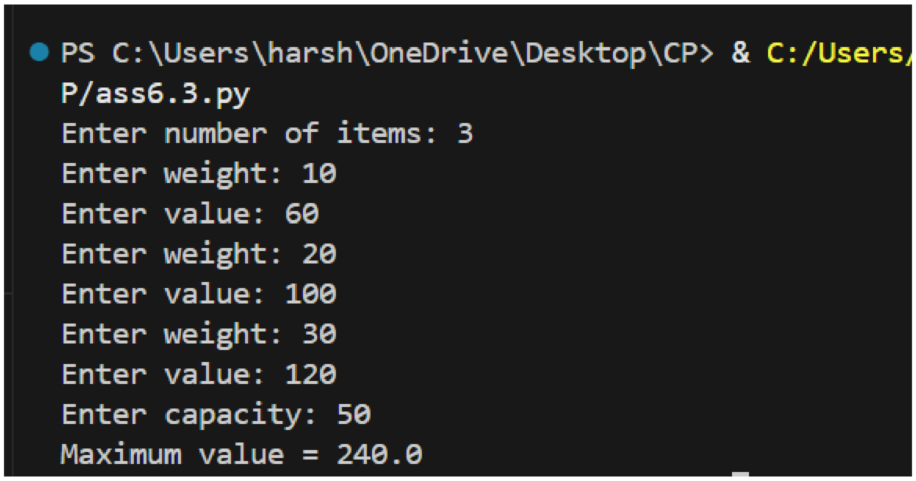
Algorithm:

```
1.  Start
2.  Read number of items n
3.  Read weight and value of each item
4.  Compute ratio = value / weight
5.  Sort items in decreasing ratio
6.  Read knapsack capacity W
7.  Set totalValue = 0
8.  For each item
        a. If W ≥ weight
                i.   Take full item
                ii.  Reduce capacity
        b. Else
                i.   Take fraction of item
                ii.  Add fractional value
                iii. Stop loop
9.  Print maximum value
10. Stop
```


ass6.3.py > ...

```python
n=int(input("Enter number of items: "))
items=[]
for i in range(n):
    w=int(input("Enter weight: "))
    v=int(input("Enter value: "))
    items.append((w,v,v/w))

cap=int(input("Enter capacity: "))
items.sort(key=lambda x:x[2],reverse=True)

total=0
for w,v,r in items:
    if cap>=w:
        cap-=w
        total+=v
    else:
        total+=r*cap
        break

print("Maximum value =",round(total,2))
```

Output :



```
● PS C:\Users\harsh\OneDrive\Desktop\CP> & C:/Users
  P/ass6.3.py
  Enter number of items: 3
  Enter weight: 10
  Enter value: 60
  Enter weight: 20
  Enter value: 100
  Enter weight: 30
  Enter value: 120
  Enter capacity: 50
  Maximum value = 240.0
```