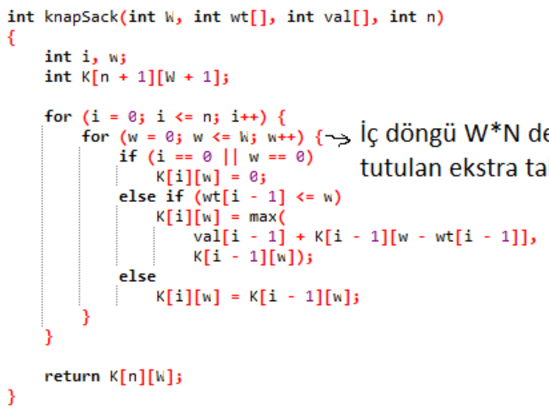
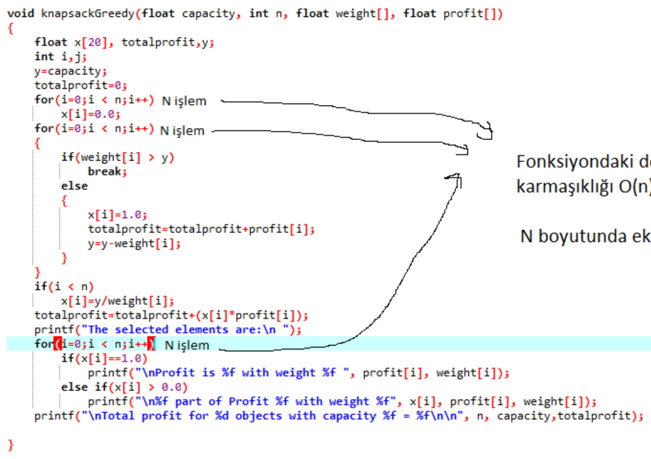
Dynamic Programming Knapsack Karmaşıklık



```c
int knapSack(int W, int wt[], int val[], int n)
{
    int i, w;
    int K[n + 1][W + 1];

    for (i = 0; i <= n; i++) {
        for (w = 0; w <= W; w++) {
            if (i == 0 || w == 0)
                K[i][w] = 0;
            else if (wt[i - 1] <= w)
                K[i][w] = max(
                    val[i - 1] + K[i - 1][w - wt[i - 1]],
                    K[i - 1][w]);
            else
                K[i][w] = K[i - 1][w];
        }
    }

    return K[n][W];
}
```

İç döngü W*N defa döndüğü için zaman karmaşıklığı O(W*N) olur, W*N boyutunda tutulan ekstra tablodan dolayı alan karmaşıklığı da aynı şekilde O(W*N) olur

Greedy Knapsack Karmaşıklık

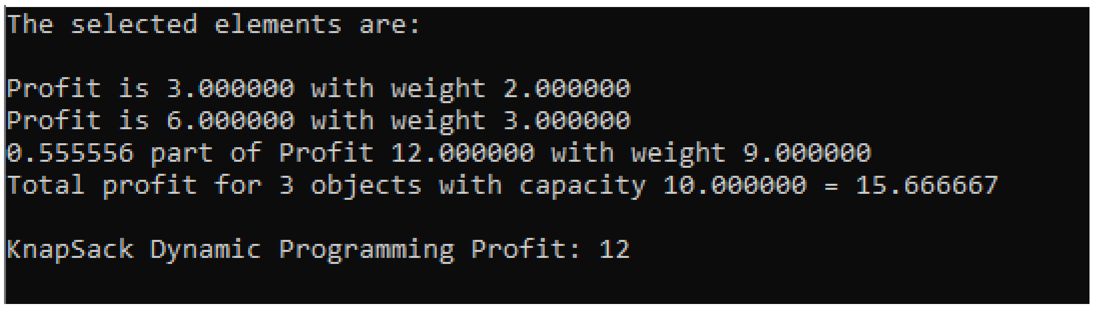


```c
void knapsackGreedy(float capacity, int n, float weight[], float profit[])
{
    float x[20], totalprofit,y;
    int i,j;
    y=capacity;
    totalprofit=0;
    for(i=0;i < n;i++) N işlem
        x[i]=0.0;
    for(i=0;i < n;i++) N işlem
    {
        if(weight[i] > y)
            break;
        else
        {
            x[i]=1.0;
            totalprofit=totalprofit+profit[i];
            y=y-weight[i];
        }
    }
    if(i < n)
        x[i]=y/weight[i];
    totalprofit=totalprofit+(x[i]*profit[i]);
    printf("The selected elements are:\n ");
    for(i=0;i < n;i++) N işlem
        if(x[i]==1.0)
            printf("\nProfit is %f with weight %f ", profit[i], weight[i]);
        else if(x[i] > 0.0)
            printf("\n%f part of Profit %f with weight %f", x[i], profit[i], weight[i]);
    printf("\nTotal profit for %d objects with capacity %f = %f\n\n", n, capacity,totalprofit);

}
```

Fonksiyondaki döngüler N defa döner bundan dolayı zaman karmaşıklığı O(n) dir.

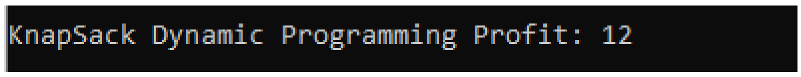
N boyutunda ek diziler tutulduğu için Alan karmaşıklığı da O(n) olur.

İnput 1 Greedy



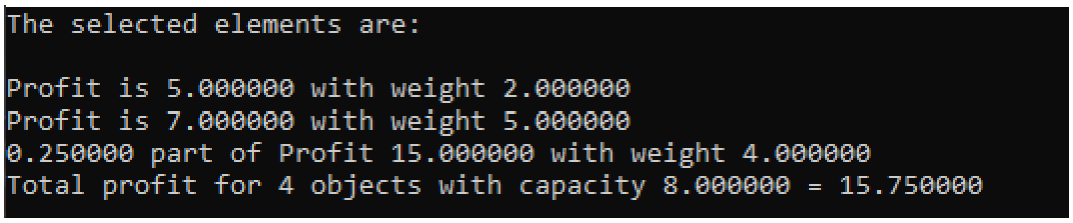
```
The selected elements are:

Profit is 3.000000 with weight 2.000000
Profit is 6.000000 with weight 3.000000
0.555556 part of Profit 12.000000 with weight 9.000000
Total profit for 3 objects with capacity 10.000000 = 15.666667

KnapSack Dynamic Programming Profit: 12
```

İnput1 Dynamic



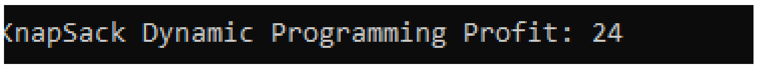
```
KnapSack Dynamic Programming Profit: 12
```

İnput2 Greedy



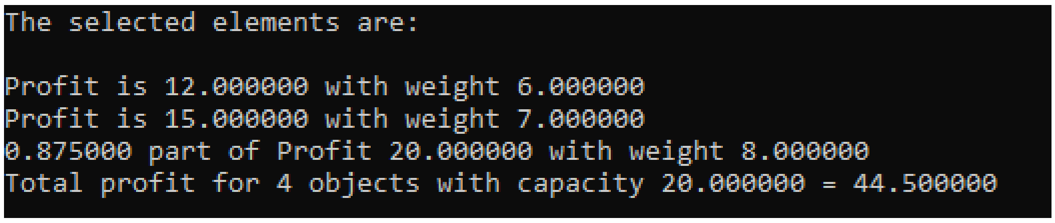
```
The selected elements are:

Profit is 5.000000 with weight 2.000000
Profit is 7.000000 with weight 5.000000
0.250000 part of Profit 15.000000 with weight 4.000000
Total profit for 4 objects with capacity 8.000000 = 15.750000
```

İnput2 Dynamic



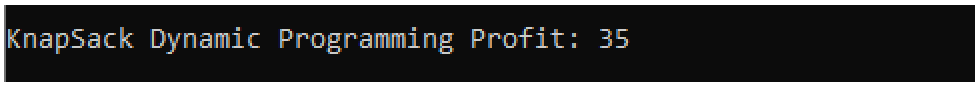
```
KnapSack Dynamic Programming Profit: 24
```

İnput 3 Greedy



```
The selected elements are:

Profit is 12.000000 with weight 6.000000
Profit is 15.000000 with weight 7.000000
0.875000 part of Profit 20.000000 with weight 8.000000
Total profit for 4 objects with capacity 20.000000 = 44.500000
```

İnput3 Dynamic



```
KnapSack Dynamic Programming Profit: 35
```