Algorithm 1 Knapsack Algorithm - Dynamic Programming

```
1:  procedure KNAPSACK(M)
2:      $cost \leftarrow M + 1$ lenght array of 0's
3:      $best \leftarrow M + 1$ lenght array of 0's
4:      for i from 1 to N do
5:          for k from size[i] to M do
6:              if val[i] + cost[k-size[i]] > cost[k] then
7:                  cost[k] = val[i] + cost[k-size[i]]
8:                  best[k] = i
9:      print(cost[M])
10:     for k from M to 0 step size[best[k]] do
11:         print(best[k])
```

Algorithm 2 Held-Karp - Dynamic Programming

```
1:  procedure TSP(G, n)
2:      for k := 2 to n do
3:          $C(\{k\}, k) := d_{1,k}$
4:      for s := to n - 1 do
5:          for all S $\subseteq \{2, ..., n\}, |S| = s$ do
6:              for all $k \in S$ do
7:                  C(S, k) := $min_{m \neq k, m \in S}$ [C(S
8:  {k}, m + $d_{m,k}$]
9:      return $min_{k \neq 1}[C(\{2, ..., n\}, k) + d_{k,1}]$
```