Dynamic Programming
Day 6

- Priyansh Agarwal

Solving Homework Problems

Problem 1: Link

- State
  - dp[i] = max profit from [0…. i] in sorted list of projects
- Transition
  - dp[i] = max(dp[i - 1], profit[i] + dp[j] s.t. j is the max index for which end[j] < start[i])
- Base Case
  - dp[0] = profit[0]
- Final Subproblem
  - dp[n - 1]

DP with Bitmasking

- Bitmasks
- Basic operations on Bitmasks
- Limitations on “N” (You will need 2^n integers to represent all the subsets)

Problem 1:

Given a list of points on a 2D plane, rearrange these points in any way such that in the final permutation of points, the sum of distances of the adjacent elements is minimized.

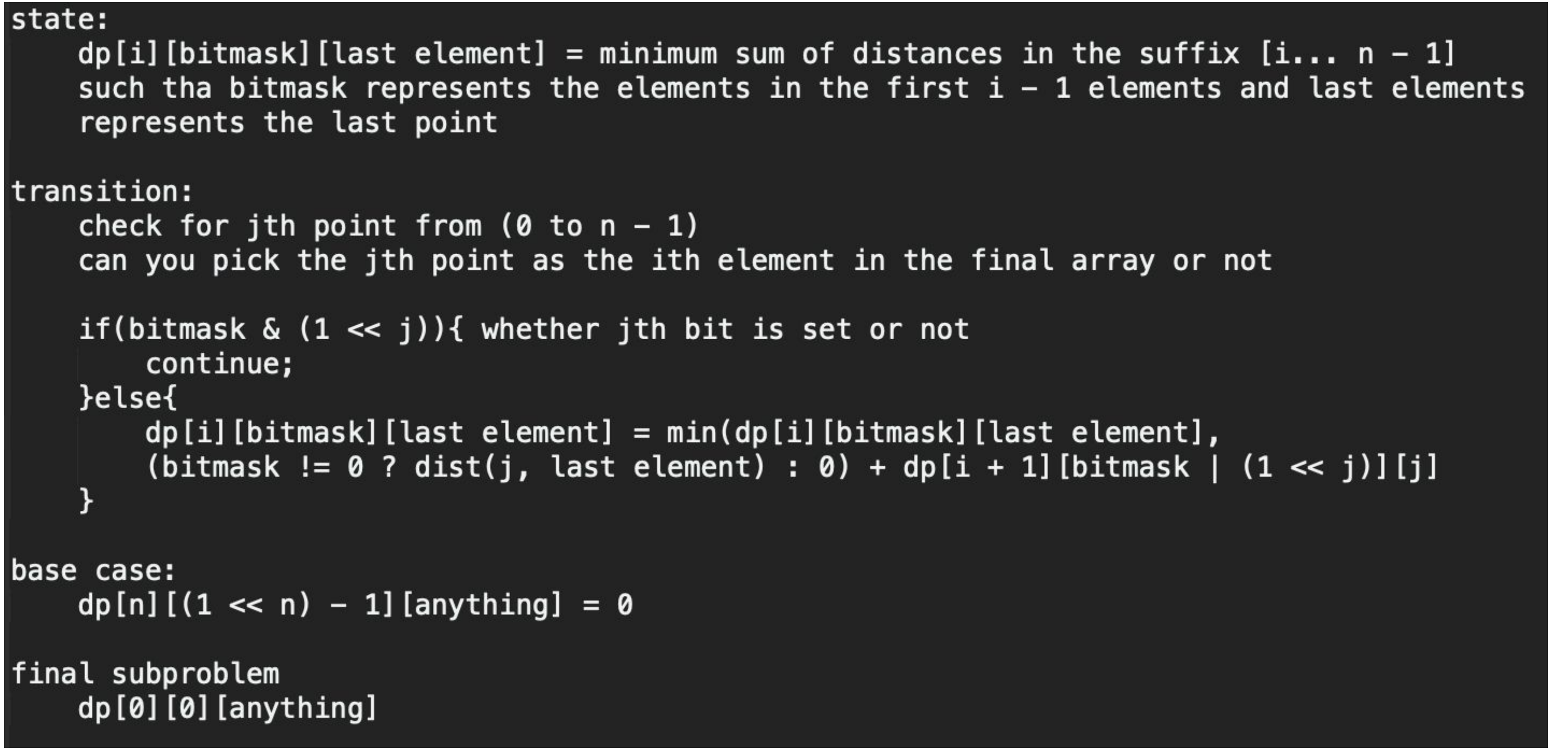
Constraints: [N <= 15], [-1e9 <= Xi, Yi <=1e9]

Points : [{0, 0}, {5, 6}, {1, 2}]

Best permutation -> [{0, 0}, {1, 2}, {5, 6}]]

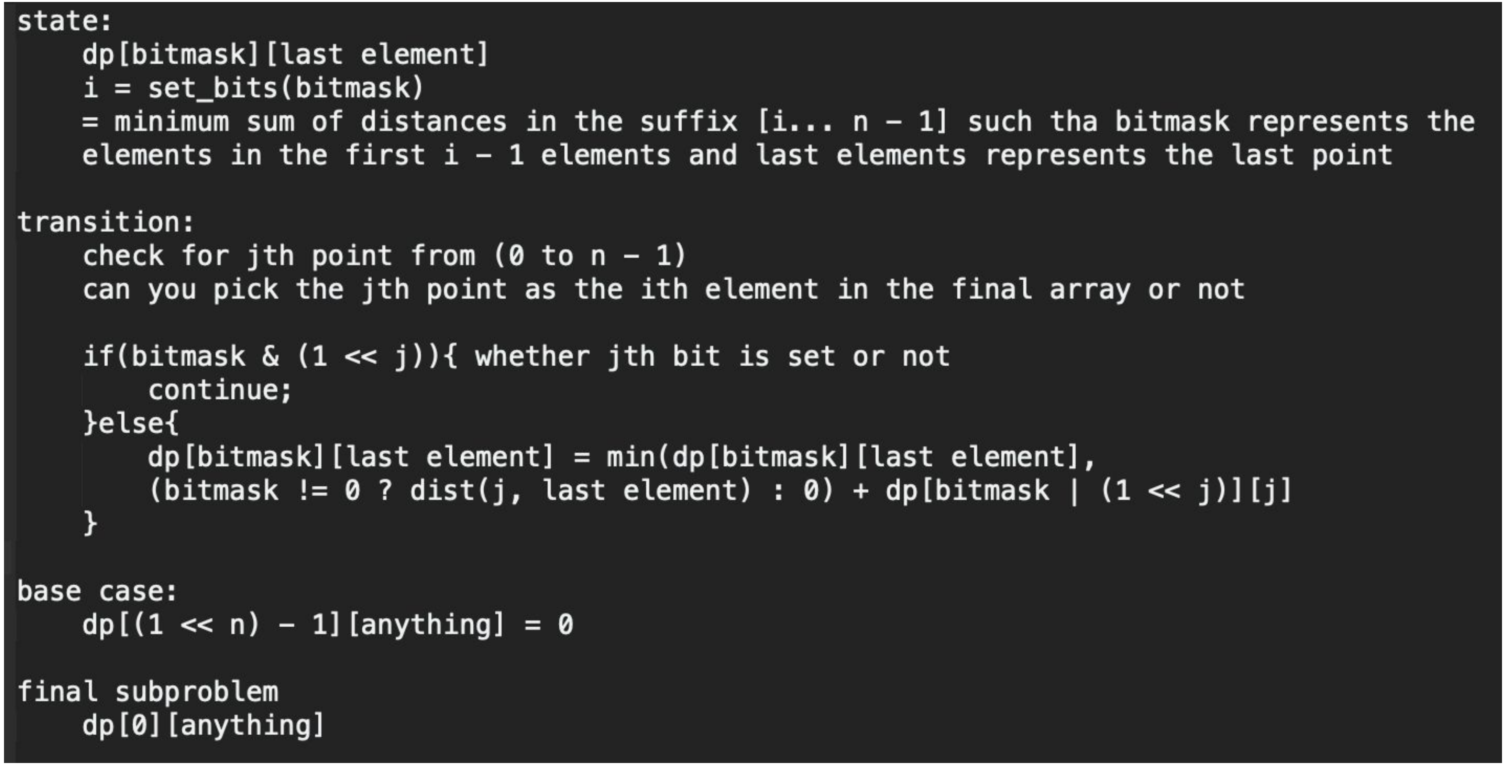
Ans = Dist(P1, P3) + Dist(P3, P2)

Problem 1: TC: $O(n^3 2^n)$, SC: $O(n^2 2^n)$



```
state:
    dp[i][bitmask][last element] = minimum sum of distances in the suffix [i... n - 1]
    such tha bitmask represents the elements in the first i - 1 elements and last elements
    represents the last point

transition:
    check for jth point from (0 to n - 1)
    can you pick the jth point as the ith element in the final array or not

    if(bitmask & (1 << j)){ whether jth bit is set or not
        continue;
    }else{
        dp[i][bitmask][last element] = min(dp[i][bitmask][last element],
        (bitmask != 0 ? dist(j, last element) : 0) + dp[i + 1][bitmask | (1 << j)][j]
    }

base case:
    dp[n][(1 << n) - 1][anything] = 0

final subproblem
    dp[0][0][anything]
```

Problem 1: TC: $O(n^2 2^n)$, SC: $O(n*2^n)$



```
state:
    dp[bitmask][last element]
    i = set_bits(bitmask)
    = minimum sum of distances in the suffix [i... n - 1] such tha bitmask represents the
    elements in the first i - 1 elements and last elements represents the last point

transition:
    check for jth point from (0 to n - 1)
    can you pick the jth point as the ith element in the final array or not

    if(bitmask & (1 << j)){ whether jth bit is set or not
        continue;
    }else{
        dp[bitmask][last element] = min(dp[bitmask][last element],
        (bitmask != 0 ? dist(j, last element) : 0) + dp[bitmask | (1 << j)][j]
    }

base case:
    dp[(1 << n) - 1][anything] = 0

final subproblem
    dp[0][anything]
```

Problem 2: Link (Homework)

- State
  -
- Transition
  -
- Base Case
  -
- Final Subproblem
  -