(a) Give the pseudocode for your implementation of a dynamic programming solution to find the above mentioned minimum antenna range r. Mention the bases cases and recurrence equation clearly and briefly explain what it represents.

Build a n×k matrix r[t, j]. Each element of matrix represents the minimum antenna range for first t houses and j base stations.

Base case 1: If there is only one base station, then r[t,1] = (P(t)-P(1))/2

Base case 2: If the number of base stations is larger than or equal to number of houses, then r[t ,j]=0.

Recurrence equation: r[t, j]= min{ max{ r[t-x,j-1], (P(t)-P(t-x+1))/2}}

, where x is from 1 to t.

The left element in max represents the minimum antenna range for the subproblem. The right element in max represents the antenna range for the j^{th} base station. We go through every possible situation where the j^{th} base station may cover 1 house, 2 houses, 3 houses … and so on. Then we find the minimum of all these antenna ranges, which is the minimum range for r[t, j].

We compute every element of this n×k matrix and finally the solution is obtained in r[n,k].

(b) Justify the runtime of the algorithm in (a).

To computer a single element of this matrix, we need O(n). Because there are at most n cases we need to consider. And for each case we only need constant time to handle (The operations are simply to subtract, divide and compare. Keeping track of the minimum value also only takes constant time).

There are totally n×k elements to be computed. Therefore, the runtime should be $O(n^2*k)$.

(c) Give the pseudocode for your implementation of a dynamic programming solution to find the set of base station positions, C.

Repeat codes in (a).

Build a n×k matrix c[t, j]. Each element of matrix represents a set of base stations for the first t houses and j base stations.

Base case 1: If there is only one base station, then c[t,1] = (P(t)+P(1))/2.

Base case 2: If the number of base stations is larger than or equal to number of houses, then c[t ,j]= the positions of the first t houses + positions of extra base stations (They can be placed anywhere. I will put them in position of house 1).

Recurrence equation: c[t, j] = c[t-x,j-1] + (P(t)+P(t-x+1))/2

This part will require the value of x, which is obtained when computing r[t, j]. The left element represents the set of base stations for the subproblem. The right element represents position for the j^{th} base station. We compute every element of this n×k matrix and finally the solution is obtained in c[n, k].

(d) Justify the runtime of the algorithm in (c).

We repeat codes of (a) in (c). The extra step in (c) is to first add the previous computed element to current element and then add one more position to it, which takes O(k) time. So the runtime should be O(n*k*(n+k)).