Q2

(a)

dp[1..2][0⋯n]: dp[i][j] denotes the max the sum of popularities of subtable(1..2, 1⋯j) when table(i,j) is put patio heater.

1. find the dp[x][y] with max value
2. table(x, y) is put patio heater
3. if dp[x][y] = p[x, y]+ dp[x][y-2] then y = y-2
4. if dp[x][y] = p[x, y]+ dp[3-x][y-2] then x=3-x, y = y-2
5. else if dp[x][y] = p[x, y]+ dp[3-x][y-1] then x = 3-x, y=y-1
6. repeat 2-5 until y<=0

(b)

when i=1..2, j=0⋯n

if j=0, then dp[i][j] = 0

if j=1, then dp[i][j] = p[i,j]

else dp[1][j] = p[1, j]+ max(dp[1][j-2], dp[2][j-2], dp[2][j-1])

    dp[2][j] = p[2, j]+max(dp[1][j-2], dp[2][j-2], dp[1][j-1])

which can be presented in :

        dp[i][j] = p[i, j]+ max(dp[3-i][j-2], dp[i][j-2], dp[3-i][j-1])

when j=0, it is obviously dp[i][j] = 0

when j=1, it is obviously dp[i][j] = p[i,j]

when j>1, if table(i,j) has been put patio heater then table(3-i, j) and table(i, j-1) must not been put patio heater, so the nearest tables are dp[3-i][j-2], dp[i][j-2] and dp[3-i][j-1]. Hence, dp[i][j] = p[i, j]+ max(dp[3-i][j-2], dp[i][j-2], dp[3-i][j-1]), which is correct.

(c)

```
algorithm(p[1..2, 1⋯n])
    dp[1..2][0..n] initial empty
    dp[1][0]=dp[2][0]=0
    dp[1][1] = p[1,1]
    dp[2][1] = p[2,1]

    for j=2 to n do:
        for j=1 to 2 do:
            dp[i][j] = p[i, j]+ max(dp[i][j-2], dp[3-i][j-1])
        end for
    end for
    return dp
```

We need to fill in table dp[1..2][0..n], so:

running time: O(n)

space time : O(n)