Solution of Question 4.

Part (i)

The asymptotic time complexity of this problem depends on the iterations of for loop.

i.e.

For first iteration, i = 1 = 2^0

For second iteration I = 2 = 2^1

For last iteration, i=n = 2^x

Where x = $\log_2(n)$.

Therefore, Time Complexity is $O(\log_2(n))$.

Part(iia)

Outer for loop runs for n times. Inner for loop also runs for n times and it contains function i.e that also runs for n times. So n(nxn) i.e

$O(n^3)$

Upper bound is: $O(n^3)$

Lower bound is: $\Omega(n^3)$

Part(iib)

For i < j, W[i][j] contains the sum P[i]+P[i+1] + . . . + P[j]

Part(iic)

We can use the value of W already computed in the previous iteration. This will

improve the time complexity to $O(n^2)$. It's more like memo function.

```
for (x = 0; x < n; x++) {
W[x][x] = P[x];
for (y = x+1; y < n; y++) {
W[x][y] = W[x][y-1] + P[y];
}}
```