HOMEWORK 2 SOLUTIONS

1) \( T(n) = 8T(n/2) + n^2, \ T(1) = 1 \) and \( n = 2^k \) Solution:

\[
T(n) = 8T(n/2) + n^2
= 8[8T(n/4) + (n/2)^2] + n^2
= 8[8[8T(n/8) + (n/4)^2] + (n/2)^2] + n^2
= 8^3T(n/8) + 8^2(n/4)^2 + 8(n/2)^2 + n^2
= 8^kT(n/2^k) + n^2(1 + 2 + \ldots + 2^{k-1})
= (2^k)^3T(1) + n^2(2^k - 1)
= n^3 + n^2(n-1)
= \Theta(n^3)
\]

2) \( T(n) = 2T(n/2) + n, \ T(1) = 1 \) and \( n = 2^k \)

Refer Recurrence Relations slide on the webpage of the class for solution.

3) \( T(n) = 2T(n - 1) + 1, \ T(1) = 1 \)

Refer Recurrence Relations slide on the webpage of the class for solution

4) Outline an algorithm to compute \( 5^n \) in \( O(\log(n)) \) time

Solution:

```
COMPUTE (n)
// To COMPUTE 5^n

If (n = 0) then Compute \( \leftarrow 1 \)
else

If (n = 1) then Compute \( \leftarrow 1 \) else

If (n is even) then
```
X ← Compute (n/2)
Compute ← X * X

If (n is odd) then
X ← Compute (n-1/2)
Compute ← 5 * X * X

The recurrence relation for above algorithm is T(n) = T(n/2) + 1
Thus, T(n) = O (log n). So, the time complexity is O (log n)