SOLUTION TO QUESTION 1.17

Two integers x and y are given. x and y require n bit and m bit respectively. Two algorithms to compute x^y need to be analyzed.

Iterative: The pseudocode for this algorithm is as follows.

```
product = x;
for i = 2 to y do
    product = product * x
```

The result after the multiplication of two integers, one p bits long and another one q bits long requires pq bits to to store. Therefore, for a given i, we are multiplying one $(i-1)n$ bits long integer with an n bit long integer. This multiplication cost is $(i-1)n^2$.The resulting number after the multiplication is $i.n$ bits long. Thus adding up all the multiplication cost we get

$$n^2 + 2n^2 + 3n^2 + \ldots + (y-1)n^2$$

. Thus the complexity of the iterative algorithm is $O(n^2y^2)$ which exponential in the input length of y which is $m = \log_2 y$.

Recursive: The pseudocode for this algorithm is as follows:

function recursive(x,y)
if y is even then return $(x^{(\lfloor \frac{y}{2} \rfloor)})^2$.
if y is odd then return $x * (x^{(\lfloor \frac{y}{2})\rfloor})^2$.

Computing $(x^{(\lfloor \frac{y}{2} \rfloor)})^2$ requires a multiplication involving two $\frac{y}{2}n$ bits integers. The cost of this operation is $\frac{y^2}{4}.n^2$ which is $O(y^2n^2)$. The recurrence relation for this recursive routine is

$T(y) = O(n)$ when y is 1-bit long.
$T(y) = T(\frac{y}{2}) + O(y^2n^2)$ otherwise.

Applying the Master Theorem we can conclude that $T(y) \in O(y^2n^2)$. The height of the recursion tree is $log_2 m$ and has $log_2 m$ nodes.

Thus both the algorithms have the same worst case complexity.