22.3.5

```
result = a;
i = 2;
while (i <= n) {
    result = result * result;
    i *= 2;
}
```

Assume that $2^{k-1} <= n < 2^k$. The while loop is executed k-1 times. The for loop is executed at most $2^k - 2^{k-1} = 2^{k-1}$ times. so, the total complexity is O(n)

Consider another implementation:

```
public static int f(int a, int n) {
    if (n == 1) {
        return a;
    }
    else {
        int temp = f(a, n/2);
        if (n % 2 == 0) {
            return temp * temp;
        }
        else {
            return a * temp * temp;
        }
    }
}
```

this performs logarithmic reduction.

this implementation results in $O(\log(n))$ complexity