Q1

We can handle this problem by a divide and conquer recursion and use repeated squaring.

When we want to compute M^n, we could compute $y = M^{\lfloor \frac{n}{2} \rfloor}$. Here, we compute $\frac{n}{2}$ by flooring. Then assign $\lfloor \frac{n}{2} \rfloor$ as the new N and do next recursion. Every recursion we return a value, if n is even, we return y^2, otherwise n is odd, we return $y^2 * M$ as we do floor operation for $n/2$.

Doing those, until $n = 0$ and this is a boundary condition and return 1. Since each recursion reduces the exponent by half, the number of recursive layers is $O(\log n)$, and the algorithm can get results in a very short time.

The Pseudocode shows below:

Algorithm 1: An algorithm

```
Function Main(M, n):
    return quickMul(M,n)
end
Function quickMul (M,n):
    if n equals 0 then
        return 1
    end
    y = quickMul(n//2)
    if n is even then
        return y * y
    end
    if n is odd then
        return y * y * M
    end
end
```