Computing a^n efficiently

Idea: use the binary expansion of n, square whenever possible.

1) Write n in binary form (sum of powers of 2):

$$n = 2^r + i_1 2^{r-1} + i_2 2^{r-2} + \ldots + i_{r-1} 2 + i_r,$$

where $i_k = 0$ or 1.

2) Put $x_0 = a$ and compute successively $x_1, x_2, \ldots, x_r$ by the rule

$$x_k = x_{k-1}^2 \cdot a^{i_k} = \begin{cases} x_{k-1}^2, & \text{if } i_k = 0 \\ x_{k-1}^2 \cdot a, & \text{if } i_k = 1. \end{cases}$$

3) Then: $x_r = a^n$.

Variant: Computing powers mod m:

– modify the above procedure by applying $\mathrm{rem}(\cdot, m)$ at each step (to keep numbers small)

$$\begin{aligned} y_0 &= \mathrm{rem}(a, m) \\ y_k &= \mathrm{rem}(y_{k-1}^2 \cdot a^{i_k}, m) \\ \Rightarrow\ y_r &= \mathrm{rem}(a^n, m) \end{aligned}$$