1 Recursive Formulae

Suppose that we want to compute $2^n \mod M$ for some numbers $n \geq 0$ and $M \geq 2$. $2^n$ can require a lot of digits to write down for large $n$, and we want to avoid that, since the end result is less than $M$.

Our first attempt avoids multiplication and only uses addition modulo $M$. We use the fact that $2^{2^n} = 2^{2^n}$ (mod $M$).

```plaintext
function PowerOf2Mod(M):
    if $n = 0$ then
        return 1
    return (PowerOf2Mod($n - 1$, M) + PowerOf2Mod($n - 1$, M)) mod M
```

What is the runtime of the above algorithm?

$\Theta(2^n)$

$\Theta(\log n)$

Correct

Now let us replace this algorithm with an iterative one that stores the results:

```plaintext
A = array indexed with $0, \ldots, n$
A[0] = 1
for $i = 1, \ldots, \log n$ do
    $A[i] = (A[i - 1] + A[i - 1]) \mod M$
return $A[\log n]$
```

What is the value of $A[i]$ in the above algorithm?

$2^i$ mod $M$

$2^i$ mod $M$

Correct

What is the runtime of this algorithm?

$\Theta(n)$

$\Theta(2^n)$

$\Theta(\log n)$

Correct

What if we are allowed to use multiplication? Suppose that $n$ is a power of two.

```plaintext
B = array indexed with $0, \ldots, \log n$
B[0] = 2
for $i = 1, \ldots, \log n$ do
    $B[i] = (B[i - 1] - B[i - 1]) \mod M$
return $B[\log n]$
```

What is the value of $B[i]$ in the above algorithm?

$2^i$ mod $M$

$2^i$ mod $M$

Correct

What is the runtime of this algorithm?

$\Theta(n)$

$\Theta(2^n)$

$\Theta(\log n)$

Correct

Remark: A clever algorithm inspired by the above can compute Fibonacci($n$) modulo a desired number $M$, in time $O(\log n)$. As a challenge, try to see the following identity involving Fibonacci numbers and matrix multiplication, to come up with this $O(\log n)$ algorithm.

```plaintext
Fibonacci($n$) = $[0 \ 1] \cdot [Fibonacci(n - 1) \ Fibonacci(n - 2)]$
```

2 Shortest Paths

Suppose that we have a weighted graph with $n$ vertices and $m$ edges and no negative cycles (so shortest paths are well-defined). Suppose for the below questions that our implementation of Dijkstra uses red-black trees (and not Fibonacci heaps).

If $m = \alpha n^3$, and we want to find the shortest path between some $u$ and $v$ which algorithms should we use? We prefer algorithms with the smallest worst-case runtime.

Suppose that we have a weighted graph with $n$ vertices and $m$ edges that all have nonnegative weights. Which algorithm should we use to find the shortest path between all pairs of vertices?

Correct

Suppose that we want to compute Fibonacci($n$) modulo a desired number $M$, in time $O(\log n)$. As a challenge, try to see the following identity involving Fibonacci numbers and matrix multiplication, to come up with this $O(\log n)$ algorithm.