Discrete Mathematics - CSCE 531 Fall 2018
In-Class Work, Day 8 (29 October 2018)

From Section 5.3

1. (Inspired by Problem 11) Give a recursive definition of $P_m(n)$, the product of $m \in \mathbb{N}$ and $n \in \mathbb{N}$.

$$P_0(0) = 0$$
$$P_m(n) = P_m(n-1) + m$$

Or, if we want the definition to apply to all m,

$$P_0(0) = 0$$
$$P_m(n) = P_m(n-1) + m$$
$$P_m(n) = P_{m-1}(n) + n$$

$$\begin{aligned} P_3(2) &= P_3(1) + 3 \\ &= P_3(0) + 3 + 3 \\ &= P_2(0) + 0 + 3 + 3 \\ &= P_1(0) + 0 + 0 + 3 + 3 \\ &= P_0(0) + 0 + 0 + 0 + 3 + 3 \\ &= 0 + 0 + 0 + 0 + 3 + 3 \\ &= 6 \end{aligned}$$

2. (Inspired by Problem 13) Prove that $f_1 + f_3 + \cdots + f_{2n-1} = f_{2n}$ whenever $n \in \mathbb{Z}^+$. In this problem, f_n is the nth Fibonacci number, where $f_0 = 0$, $f_1 = 1$ and $f_n = f_{n-1} + f_{n-2}$. *Hint: use proof by induction and the definition of the Fibonacci sequence.*

Proof: Let $P(n)$ be the proposition $f_1 + f_3 + \cdots + f_{2n-1} = f_{2n}$.

Base case: $P(1)$ is true because $f_1 = 1 = 0 + 1 = f_0 + f_1 = f_2$.

Inductive hypothesis: For some $k \in \mathbb{Z}^+$, $P(k)$ holds, i.e. it is the case that $f_1 + f_3 + \cdots + f_{2k-1} = f_{2k}$.

Inductive step: We must show that $P(k) \to P(k+1)$ holds, i.e. that $f_1 + f_3 + \cdots + f_{2k+1} = f_{2(k+1)}$ whenever $f_1 + f_3 + \cdots + f_{2k-1} = f_{2k}$. By the definition of the Fibonacci sequence, $f_{2k} + f_{2k+1} = f_{2k+2} = f_{2(k+1)}$. Thus, making the penultimate term in the left hand side of $P(k+1)$ explicit, we have

$$\begin{aligned} f_1 + f_3 + \cdots + f_{2k+1} &= f_1 + f_3 + \cdots + f_{2k-1} + f_{2k+1} \\ &= f_{2k} + f_{2k+1} \\ &= f_{2(k+1)}. \blacksquare \end{aligned}$$

3. (Inspired by Problem 35) An **alphabet**, often denoted Σ, is a finite non-empty set for which the elements are called **symbols**. A **string** over Σ is a sequence $\sigma_1\sigma_2\cdots\sigma_n$, where $n \in \mathbb{N}$ and $\forall i(\sigma_i \in \Sigma)$. The case $n = 0$ corresponds to the **empty string**, denoted λ (or ϵ), and the set of all possible strings over Σ is denoted Σ^*. For example, if $\Sigma = \{0,1\}$, then $\{\lambda, 0,1,00,01,10,11\} \subset \Sigma^*$. The **concatenation** of a string $\sigma_1\sigma_2\cdots\sigma_n$ with a symbol σ_{n+1} is the string $\sigma_1\sigma_2\cdots\sigma_{n+1}$.

   Give a recursive definition of the **reversal** w^R of a string w. The reversal of a string is the string consisting of the symbols of the string in reverse order. For example, $bacaba^R = abacab$.

$$\lambda^R = \lambda$$
$$(x\sigma)^R = \sigma x^R$$

where $\sigma \in \Sigma$ and $x \in \Sigma^*$.

From Section 5.4

4. (Inspired by Problem 7) Give a recursive algorithm (write the pseudocode in the style used by the book) for computing $n \cdot x$ whenever $n \in \mathbb{Z}^+$ and $x \in \mathbb{Z}$, using just addition and subtraction.

```
Procedure mult(n, x: integers with n > 0)
If n = 1 then return x
Else return x + mult(n − 1, x)
{output is n · x}
```

5. (Inspired by Problem 13) Give a recursive algorithm for finding $n! \bmod m$ whenever $n \in \mathbb{N}$ and $m \in \mathbb{Z}^+ - \{1\}$. *Hint: recall that* $a \cdot b \bmod m = (a \bmod m) \cdot (b \bmod m) \bmod m$

```
Procedure modfact(n: non-negative integer, m: positive integer)
If n = 0 then return 1
Else return (n mod m · modfact(n − 1, m)) mod m
{output is n! mod m}
```

Of course, if $n \geq m$, then $m|n!$, so $n!\, mod\ m = 0$.

6. (Inspired by Problem 23) Devise a recursive algorithm for computing n^2 where $n \in \mathbb{Z}^+$, using the fact that $(n+1)^2 = n^2 + 2n + 1$. Then prove the algorithm is correct. *Hint: Use induction. Optional: Make your algorithm work for* $n \in \mathbb{N}$.

```
Procedure square(k: nonnegative integer)
If k = 0 return 0
Else return square(k − 1) + 2k − 1
{output is k^2}
```

Proof: Let $P(n)$ be the proposition that `square(n)` returns n^2.

Base case: P(0) holds because `square(0)` returns $0 = 0^2$.

Inductive hypothesis: $P(k)$ holds, i.e. `square(k)` returns k^2.

Induction step: We must show that $P(k) \to P(k+1)$, i.e. that `square(k+1)` returns $(k+1)^2$ whenever the inductive hypothesis is true. Since $k + 1 > 0$, `square(k+1)` returns $\text{square}\big((k+1) - 1\big) + 2(k+1) - 1 = \text{square}(k) + 2k + 2 - 1 = k^2 + 2k + 1 = (k+1)^2$, where we have used the inductive hypothesis.■