Last time:
- RSA algorithm
  - Choosing public key
  - Encryption and decryption
  - Fast exponentiation
  - Assumption: factoring is computationally hard

Announcements:
- Exam grades out; regrade requests due Monday
- Course grade estimates soon
- Discussion sessions open to all
Inductively defined set / BNF notation: \( x \in X ::= \cdots \mid \cdots x \cdots \mid \cdots x_1 \cdots x_2 \cdots \) means elements of \( X \) are formed by applying rules 1, 2, and 3 a finite number of times (replacing \( x \)s with elements of \( X \) already formed)

Examples:

- **Strings**: \( x \in \Sigma^* ::= \varepsilon \mid xa \quad a \in \Sigma \)
- **Naturals**: \( n \in \mathbb{N} ::= Z \mid S \ n \) (Note: \( S \) stands for “successor”)
- **Expressions**: \( e \in \text{Expr} ::= n \mid e_1 + e_2 \mid \text{if } e_1 = 0 \text{ then } e_2 \text{ else } e_3 \quad n \in \mathbb{N} \)

To define \( f : X \to Y \) where \( X \) is inductively defined, define \( f(x) \) for \( x \)s formed using each rule; you may apply \( f \) to any substructure of \( x \)

- **Strings**: define \( f(\varepsilon) \) and \( f(xa) \) (in terms of \( f(x) \))
- **Expressions**: define \( f(n) \), \( f(e_1 + e_2) \) in terms of \( f(e_1) \) and \( f(e_2) \), and \( f(\text{if } e_1 = 0 \text{ then } e_2 \text{ else } e_3) \) in terms of \( f(e_1) \), \( f(e_2) \), and \( f(e_3) \)

To prove \( \forall x \in X, P(x) \) by structural induction, prove \( P(x) \) for \( x \)s formed using each rule, assuming \( P \) of each substructure of \( x \)

- **Strings**: prove \( P(\varepsilon) \) and \( P(xa) \) assuming \( P(x) \)
- **Expressions**: prove \( P(n) \), \( P(e_1 + e_2) \) assuming \( P(e_1) \) and \( P(e_2) \), and \( P(\text{if } e_1 = 0 \text{ then } e_2 \text{ else } e_3) \) assuming \( P(e_1) \), \( P(e_2) \), and \( P(e_3) \)