Mechanics of Functions (Addendum)

Exercise: Generating Prime Factorizations

• A more computationally intense problem is to generate the prime factorization of a positive integer \( n \).
• An integer is prime if it’s greater than 1 and has no positive integer divisors other than 1 and itself.
  - 5 is prime: it’s divisible only by 1 and 5.
  - 6 is not prime: it’s divisible by 1, 2, 3, and itself.
• Some prime factorizations:
  - \( 501 = 3 \cdot 167 \)
  - \( 502 = 2 \cdot 251 \)
  - \( 503 = 503 \)
  - \( 504 = 2 \cdot 2 \cdot 2 \cdot 3 \cdot 3 \cdot 7 \)
  - \( 505 = 5 \cdot 101 \)
  - \( 506 = 2 \cdot 11 \cdot 23 \)
  - \( 507 = 3 \cdot 13 \cdot 13 \)
  - \( 508 = 2 \cdot 2 \cdot 127 \)
  - \( 509 = 509 \)
  - \( 510 = 2 \cdot 3 \cdot 5 \cdot 17 \)
  - \( 511 = 7 \cdot 73 \)
  - \( 512 = 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \)

Some thought questions and exercises:
• The solution relies on a single Boolean called \( \text{first} \). What problem is \( \text{first} \) solving for us?
• During our trace of \( \text{constructFactorization}(180) \), \( \text{factor} \) assumed the values of 2, 3, 4, and 5. 2, 3, and 5 are prime numbers and therefore qualified to appear in a factorization? How does the implementation guarantee 4 will never make an appearance in the returned factorization?
• What is returned by \( \text{constructFactorization}(1) \)? How could you have changed the implementation to return “1 = 1” as a special case return value?
• Trace through the execution of \( \text{constructFactorization}(363) \) as we did for \( \text{constructFactorization}(180) \).
• Our implementation relies on a parameter named \( n \) to accept a value from the caller, and then proceeds to destroy \( n \) by repeatedly dividing it down to 1. Does this destruction of \( n \) confuse \( \text{PrimeFactorizations} \)'s for loop? Note that its counting variable is also named \( n \).

Exercise: Drawing A Checkerboard

• For the rest of lecture, we’ll collectively design and decompose (and to the extent we have time, implement) a graphics program that draws the initial configuration for a game of checkers.