Reading / Source Material

- Mitchell, Chapter 2

Objectives

1. understand the meaning of a partial function and its relevance to programming
   (a) be able to explain the difference between a partial and total function
   (b) be able to determine whether a simple function is a partial or total function
   (c) be able to explain the relationship between noncomputable functions and nontermination

2. understand and be able to apply the Turing-Church thesis to reasoning about programming languages
   (a) be able to recall $\lambda$-calculus and the Turning as two separate formal notions of computable functions
   (b) be able to describe informally the some of the kinds of language features (recursion with parameters or loops + variable assignment) that can make a language Turing complete
   (c) be able to explain the Church-Turing thesis (all programming languages are Turing complete)

3. understand and be able to apply the notion of undecidability
   (a) be able to explain the definition of undecidability
   (b) understand that some problems are undecidable
   (c) be able to explain the proof of the Halting Problems undecidability

Vocabulary

termination, nontermination, diverge, partial function, partial recursive functions, computable, Turing machine, Turing-Complete, $\lambda$-calculus, undecidability, and the Halting Problem.

Outline

1. partial functions
2. “effectively computable functions” and Turing-Completeness
3. the halting problem