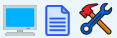
CS 381: Programming Language Fundamentals*

Spring 2025

*In the language of the U.S. government: 

Poll ...

Poll 1

What was the first programming language you learned?

Poll 2

What is your favorite programming language?

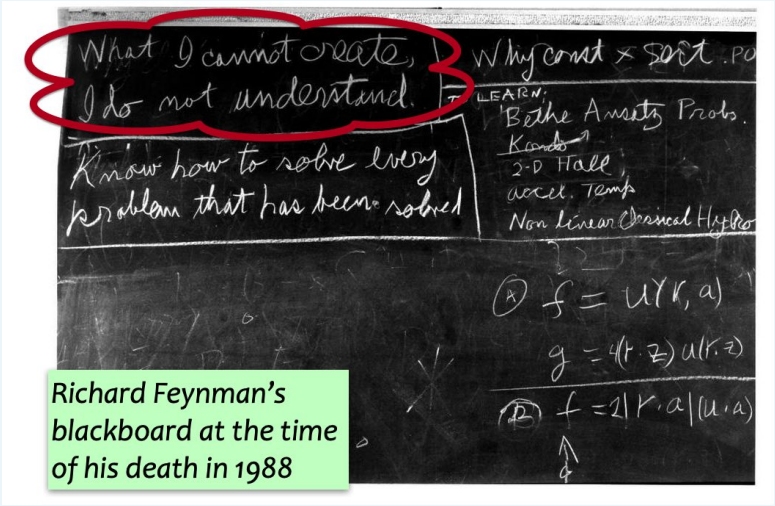
About CS 381

Programming Language ***Fundamentals*** *means*
Theory *of Programming Languages*
(CS 381 is ***not*** *a programming course!)*

How to be Successful in CS 381

- *Start programming in Elm immediately*
- *Take the homework seriously*
- ***Ask*** *(and* ***answer****!) questions on Canvas*
- ***Ask*** *questions during and after class and in office hours*

Maybe the Most Important Slide …



Richard Feynman's blackboard at the time of his death in 1988

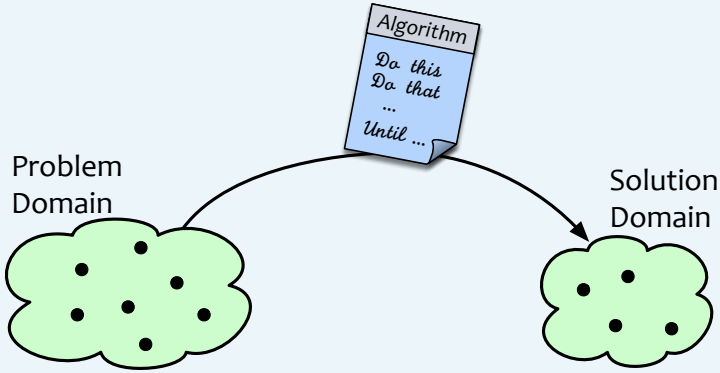
1. Introduction

What is Computer Science?

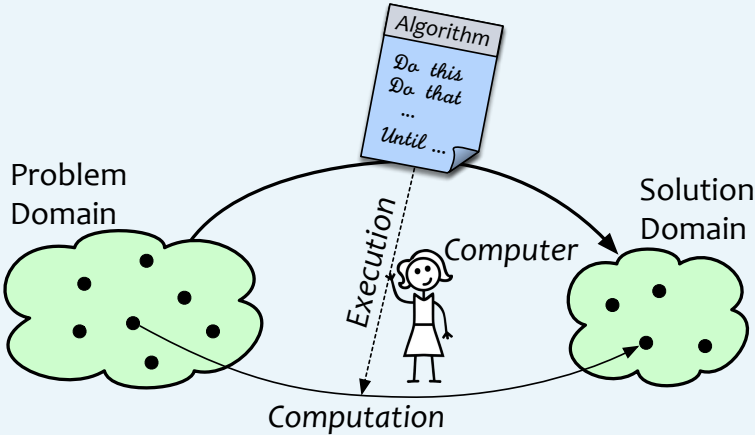
The Role of Programming Languages in CS

How to Study Programming Languages

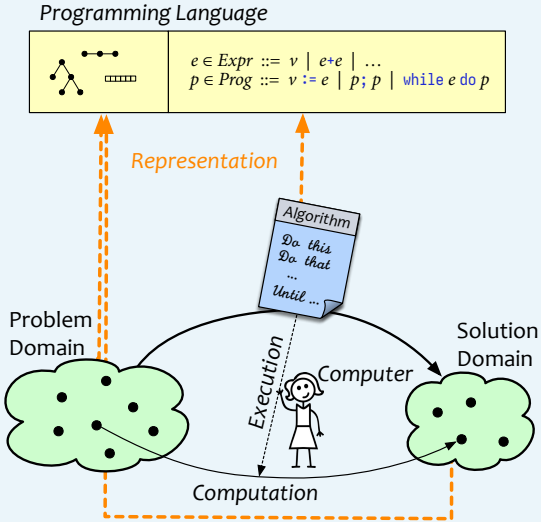
Systematic Problem Solving



Computation = Algorithm Execution



Automated Computation



1. Introduction

What is Computer Science?

The Role of Programming Languages in CS

How to Study Programming Languages

Talking to the Machines

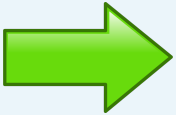
PL = HCI

A programming language is a human-computer interface

The Language of Computing

Fact

Programming languages are computer scientists' most basic tool



Obligation

Computer scientists should understand the principles of programming languages

People ↔ Technology

Motor Vehicles (cars, trucks, buses, ...)	
Passenger	Everyone
Driving	Licensed
Understanding	Mechanics
Designing	Engineers

Programming Languages (C, Python, Elm, ...)	
End User	Everyone
Programming	Programmers
Understanding	Computer Scientists
Designing	CS Researchers

Programming Languages – Only Tools?

Science vs. Engineering

Science: *tries to understand and explain*
Engineering: *applies science to build stuff*

Science		***Applications***
Physics	*separate* →	Mechanical, Civil Engineering
Chemistry	*separate* →	Chemical Engineering
Biology	*separate* →	Agriculture, Medicine
Computer Science	← *part of*	Software Engineering

Central Role of PL in Computer Science

PL supports both aspects of Computer Science:

- *To understand and explain (science):*
  *We need **languages** to describe and reason about computations*

- *To build cool stuff (engineering):*
  *We need **languages** to describe computations for a machine to execute*

1. Introduction

What is Computer Science?

The Role of Programming Languages in CS

How to Study Programming Languages

Lots of Languages

Estimate:
700 - 9,000
languages

Learn them all …

CS = Science of Abstraction

One Algorithm/Program ⇄ Many Computations
(Execution / Abstraction)

One Language ⇄ Many Algorithms/Programs
(Describes / Abstraction)

One Metalanguage ⇄ Many Programming Languages
(Describes / Abstraction)

Abstracting from Individual Languages

Focus on:

Programming Language Concepts

- *Syntax*
- *Semantics*
- *Type System*
- *Binding & Scope*
- *Evaluation & Parameter Passing*

PL Abstraction Hierarchy

Abstraction Level		Example
4	Meta Language	Regular Expressions, Grammars, Inference Rules
3b	Feature	Syntax, Semantics (scope, types, evaluation)
3a	Model/Paradigm	Lambda Calculus, Turing Machine, Predicate Calculus
2	Language	Elm, Lisp, Scheme, C, Java, Python, Prolog
1	Program	`fac n = if n==1 then 1 else n*fac (n-1)`
0	Computation	`fac 3` → `3*fac 2` → `3*2*fac 1` → `3*2*1` → `6`

Functional Paradigm: Computation ≡ sequence of expression, resulting in a value

PL Abstraction Hierarchy: To be learned in CS 381

Abstraction Level		Example
4	Meta Language	Regular Expressions, *Grammars*, Inference Rules
3b	Feature	*Syntax*, *Semantics (scope, types, evaluation)*
3a	Model/Paradigm	Lambda Calculus, Turing Machine, Predicate Calculus
2	Language	*Elm*, Lisp, Scheme, C, Java, Python, *Prolog*
1	Program	`fac n = if n==1 then 1 else n*fac (n-1)`
0	Computation	`fac 3` → `3*fac 2` → `3*2*fac 1` → `3*2*1` → `6`

Abstraction Users and Creators

	Objects at Level	Who Creates Them?	Who Uses Them?
3-4	Meta & Feature & Paradigm	CS Researchers	*Computer Scientists*
2	Language	*Computer Scientists*	*Programmers*
1	Program	*Programmers*	Everyone
0	Computation	Everyone	Everyone

Abstraction Level(Computer Scientist) > Abstraction Level(Programmer)

Our Focus: Programming Language Concepts and Theory

Focus on how to ***define*** programming languages

For several toy languages, we will:

- Define the ***structure*** of its programs
- Define the ***meaning*** of its programs
- Identify the ***features*** that are common to many languages

Role of Metalanguages

***Metalanguage*:** A language to define the structure and meaning of another language!

Metalanguages in CS 381

- *Grammars*
- *Elm*
- *English*

Summary

Focus on programming language ***concepts***

- *Define **abstract syntax** of languages*
- *Define **semantics** of languages*
- *Compare different **language features***

Use ***metalanguages*** for examining concepts

- *Definitions using **grammars***
- *Interpreters in **Elm***