Welcome to CSC324!

David Liu, Department of Computer Science

https://www.cs.toronto.edu/~david/csc324

Who is David?

- Born in Ottawa, grew up in Toronto
- Studied math, CS, and education in undergrad
- In my fifth (!!) year as a teaching-stream faculty member
- Programming languages enthusiast (but not expert)
Administrative Matters

Contact time
Lectures: Wednesdays and Fridays
Labs: Mondays (starts next Monday!)

Assessments
10 weekly exercises
2 large assignments
1 midterm
1 final exam

New programming languages (Racket and Haskell)
Start learning early!
**Academic integrity**

Key warnings

1. Discuss ideas, but do not share code.
2. Make use of online documentation, but do not copy-and-paste code.

**Getting help**

Keep up with labs and exercises

Ask questions on Piazza and during office hours (Mon 2–4, Wed 3–5)

---

Principles of Programming Languages

---

David, alone in office hours
In this course, you will learn to:

1. Define, analyze, and modify syntactic features of a programming language.
2. Define and analyze semantic features of a programming language.
3. Design and implement programs that operate on other programs.
4. Write programs using the functional programming paradigm.

Defining a Programming Language: Syntax and Semantics

**Syntax**

the form of the elements of a language

**Grammar**

the set of rules that specifies the syntax of a language
expression

a syntactically valid element of a language
(i.e., a string that can be generated by applying the rules of the language's grammar)

parse

transform a string representation of an expression into a “more structured” representation
(3 + 4) * (5 – (10 / 2))

abstract syntax tree
a tree-based representation of a computer program

semantics
the meaning of the elements of a language

A tale of two models
The Turing machine
A model of computation whose semantics is based on reading from and writing to mutable state.

The lambda calculus
A model of computation whose semantics is based on the evaluation of pure functions.
expr = ID                        (* Identifier *) 
    | "(" "λ" ID "." expr ")"   (* Function value *) 
    | "(" expr expr ")"         (* Function call *) 
    ;

Stressed is Desserts spelled backwards