
# Welcome to CS / ENGRD 2110!

Get started by signing into PollEv and answering some questions about yourself!

Use your Cornell email address
(but don't "Sign in with Google")


# Lecture 1: Introduction

CS 2110

September 25, 2026

# Matt Eichhorn

- PhD (Applied Math) at Cornell
- BS (Computer Science, Math)

**Teaching:**
- CS 2110, CS 2800, ENGRI 1101

**Research:**
- Algorithm Design: optimization with unknown / random inputs
- Statistics on networks: how do interventions cascade through populations

**Interests:**
- Crosswords / other puzzles, origami, crosswords, Buffalo Bills

# Goal of CS 2110: Write *Better* Code

What makes code *better* ?

# Course Map

# Course Logistics

## Everything you need is linked from our course website

courses.cis.cornell.edu/courses/cs2110/2026fa/

(find a link on our Canvas page)

- Syllabus, installation/style guides, etc.
- Lecture Notes (with interactive exercises)
- Slides and demo code (after lecture)
- Discussion (mostly after sections)
- Assignments
- Grades

# Class Norms

**Prioritizing Effort and Integrity:**
Assignments are designed for their learning benefit, not their end product
Be pragmatic and transparent with generative AI use
Adhere to the course collaboration policy (syllabus)

**Mutual Respect:**
We all are here with the same goals
Communicate kindly/supportively to peers and course staff
Let us know if anyone behaves disrespectfully

**Inclusivity:**
We come with different backgrounds and needs
Everyone in the course is meant to be here
Let the course staff know how we can best support you

# Learn by Challenging Yourself

# Let the learning begin!

# Today's Learning Outcomes

1. Explain the process of compilation and the benefits of *statically typed* programming languages.

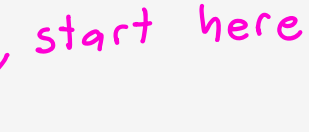
2. Determine the static type of an expression involving one of more operators, method calls, and/or implicit/explicit type coercions.

3. Visualize the state of the runtime stack in a program involving multiple method calls.

# A Basic Python Program

convert.py

```
1  def days_to_minutes(days):
2    hours = 24 * days
3    minutes = 60 * hours
4    return minutes
5
6  if __name__ == "__main__":
7    print("Enter a number of days: ", end="")
8    days = input() # get user input from the console
9    mins = days_to_minutes(days)
10   print("There are " + mins + " minutes in " + days + " days.")
```




If we run this program and type "3" into the console, what will be its output?

# Dynamic vs. Static Typing

The **type** of a variable (or expression) determines its possible values and how it can be used within a program.

**Dynamically Typed** Languages:

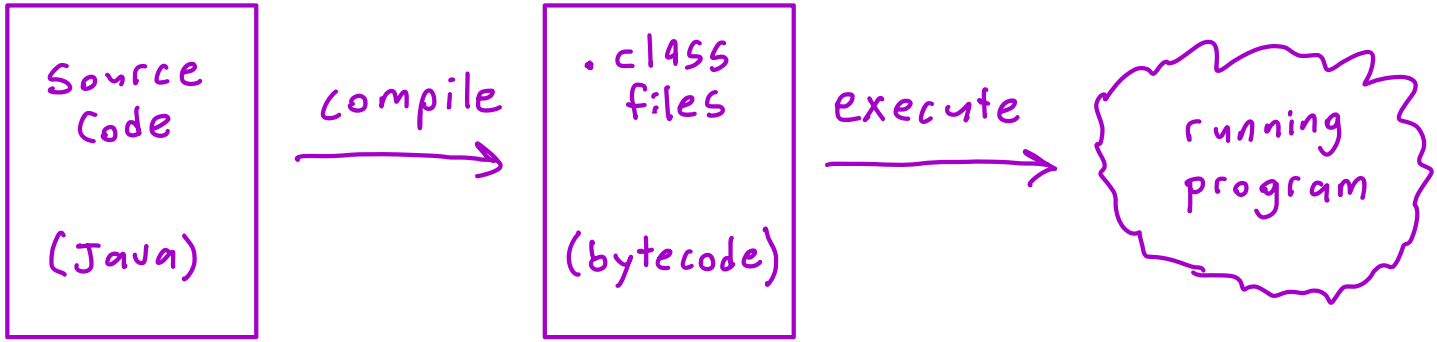
**Statically Typed** Languages:

# **Coding Demo:** Translation to Java

(We include slides like this one in case you want a place to take notes during/about the demo.)

# Compilation

= Intermediary "code translation" step that happens before code runs



# Primitive Types

Java has eight primitive types. The four most important (for us) are:

```
int        double        char        boolean
```

# Expressions

= units of code with an assigned **type** and **value**

# Statements and Assignment

A **statement** is a unit of code that describes an action the program should take (i.e., it produces a **side effect**).

# Primitive Type Coercion

Most built-in java operators preserve primitive types.

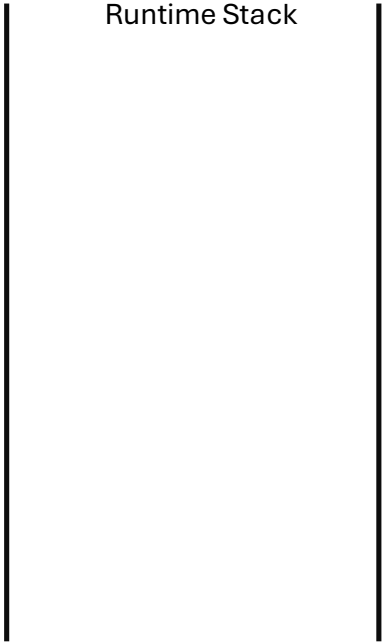
**Coercion** alters the type of an expression. It comes in two forms:

Implicit (**widening**):

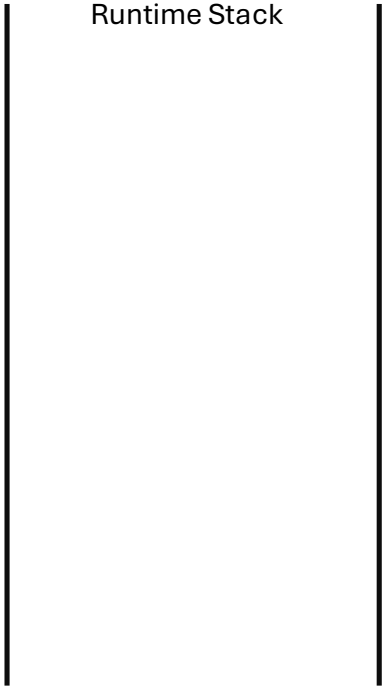
Explicit (**casting**):

# Program Execution

- Starts by calling the `main()` method
- Calling a method creates a **call frame** on the **runtime stack**



# Diagramming our Code




Convert.java

```
1  public class Convert {
2    static int daysToMinutes(int days) {
3      int hours = 24 * days;
4      int minutes = 60 * hours;
5      return minutes;
6    }
7
8    public static void main(String[] args) {
9      Scanner sc = new Scanner(System.in));
10     int days = sc.nextInt();
11     int mins = daysToMinutes(days);
12     System.out.print(…);
13   }
14 }
```