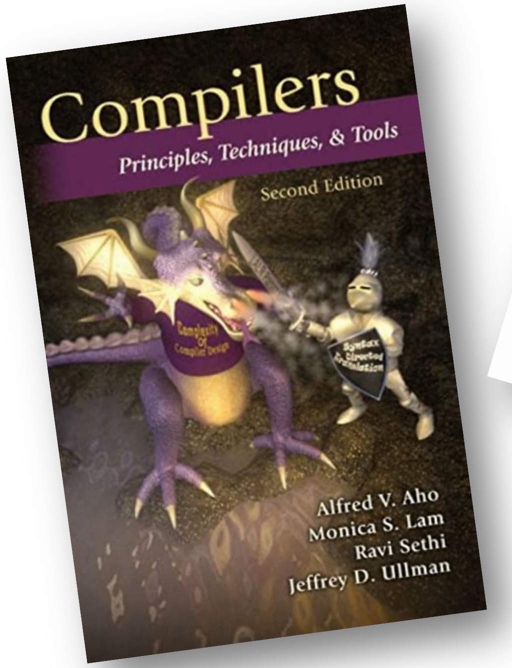
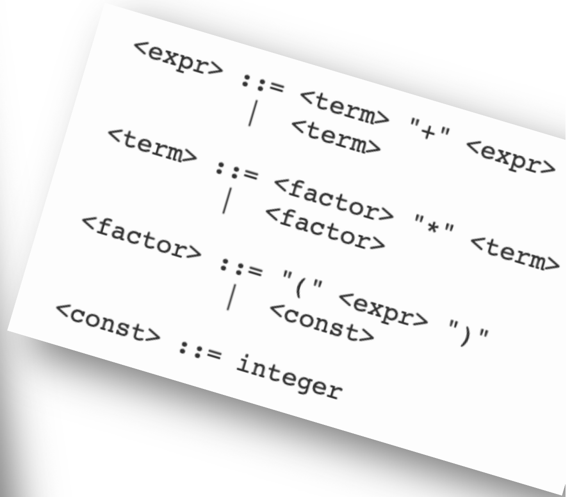
CSE211: Compiler Design

Sept. 29, 2023

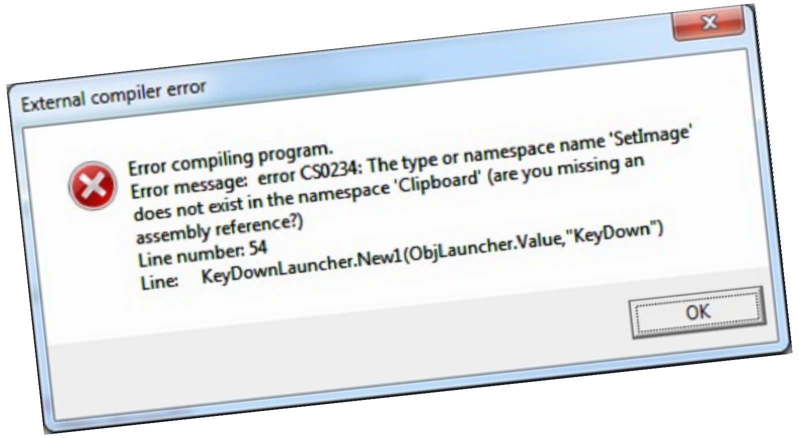
- **Topic**: Course Introduction

- **Questions**:
  - *What is a compiler?*
  - *What are some of your favorite compilers?*
  - *Have you ever built a compiler?*





```
<expr> ::= <term> "+" <expr>
         | <term>
<term> ::= <factor> "*" <term>
         | <factor>
<factor> ::= "(" <expr> ")"
           | <const>
<const> ::= integer
```



External compiler error

Error compiling program.
Error message: error CS0234: The type or namespace name 'SetImage' does not exist in the namespace 'Clipboard' (are you missing an assembly reference?)
Line number: 54
Line: KeyDownLauncher.New1(ObjLauncher.Value,"KeyDown")

OK

LLVM
COMPILER INFRASTRUCTURE

Hi Everyone ☺

https://users.soe.ucsc.edu/~tsorensen/

Room E2-233

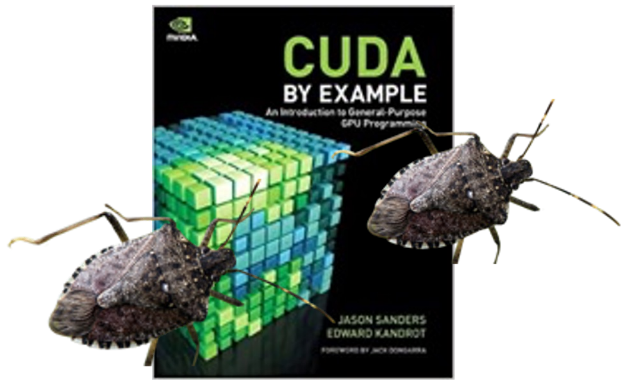
- Tyler Sorensen
  - fourth year faculty at UCSC
    fourth time teaching this class
  - Work on parallel programming languages (especially for GPUs)
  - Invited member of the Khronos Group
  - Consultant for Trail of Bits, a cyber security company

https://traveleering.com/
https://wikipedia.com
https://Princeton.edu

Research Interests

MS: Utah

Nvidia GPU memory consistency testing and specification

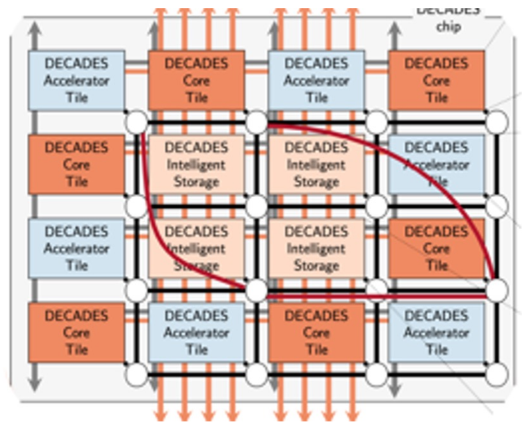


PhD: London

GPU synchronization, including a device-wide barrier implementation

Post Doc: Princeton

Compilers targeting new architectures



UCSC: Heterogeneous Programming Lab

Working on:

- Figuring out what heterogeneous systems do
- How to test them
- How to use them

Areas cover: Semantics, Testing, Performance, etc.

- ***Compilers are a large component of each research topic!***

Today’s class

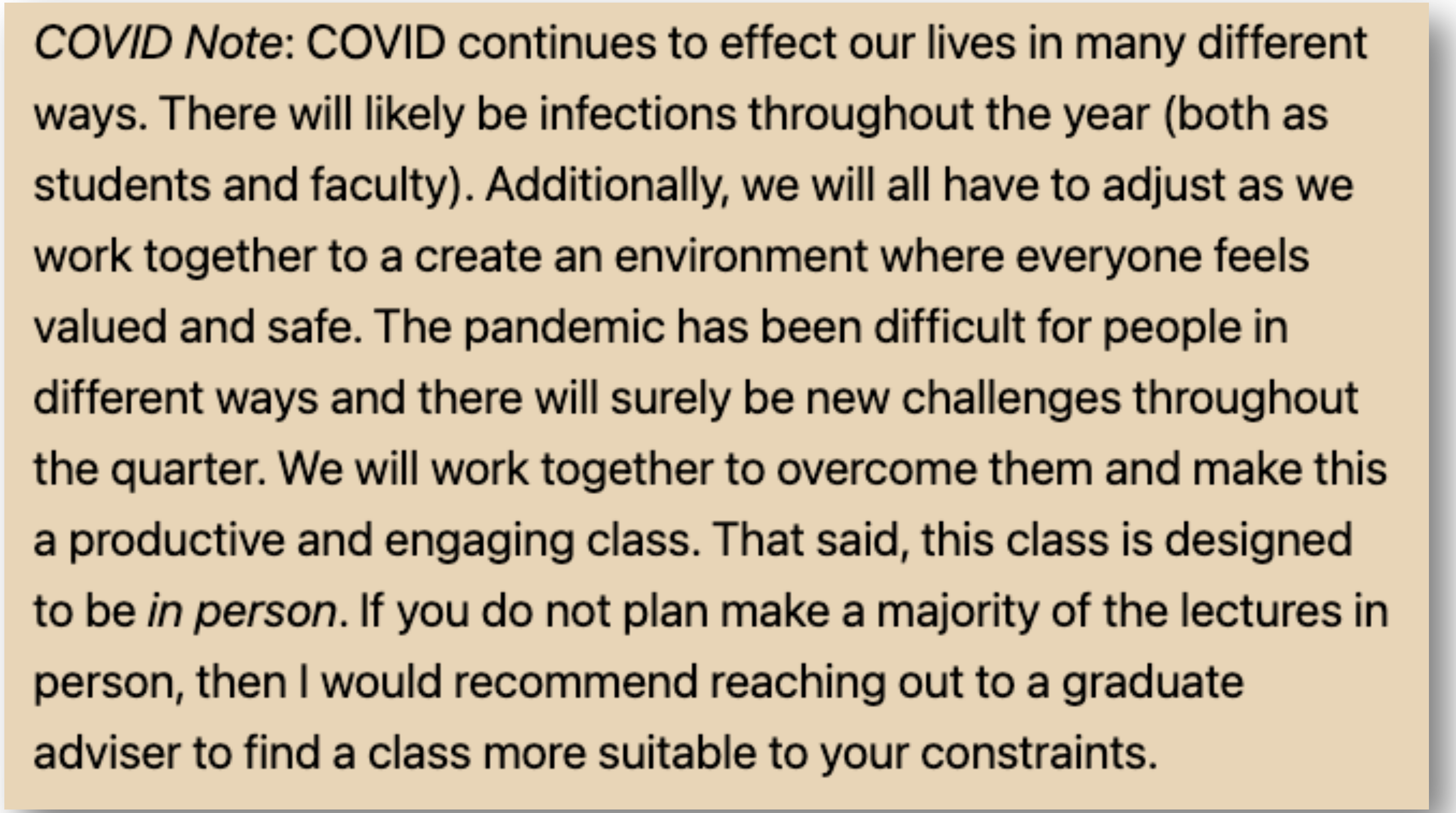
- Class syllabus (I apologize in advance for the text slides)

Enrollment

- We have about 10 spots remaining (Capacity of 45)
  - Grad students: tell your friends!
  - Undergrads: I have permission codes for you
    - Send me an email after class. Try to enroll ASAP

- For reference, this class started out with 8 students 4 years ago!

Are things back to normal?



COVID Note: COVID continues to effect our lives in many different ways. There will likely be infections throughout the year (both as students and faculty). Additionally, we will all have to adjust as we work together to a create an environment where everyone feels valued and safe. The pandemic has been difficult for people in different ways and there will surely be new challenges throughout the quarter. We will work together to overcome them and make this a productive and engaging class. That said, this class is designed to be *in person*. If you do not plan make a majority of the lectures in person, then I would recommend reaching out to a graduate adviser to find a class more suitable to your constraints.

Course resources

- Public course website: https://sorensenucsc.github.io/CSE211-fa2023/index.html
  - Schedule, slides, syllabus, additional resources

- Private course website: Canvas
  - grades, announcements, SETs, homeworks, tests, lecture recordings
  - If you don’t have access, let me know ASAP!

- Piazza:
  - Used for discussion

- Docker Image
  - Used for homework (instructions incoming)

Description

In this class you will learn about advanced topics in compiler design and implementation. In the abstract, compilers explore many of the foundational problems in computer science. In practice, compilers are massive pieces of well-oiled software, and are some of the engineering marvels of the modern world. Given the end of Dennard’s scaling, compilers will play an increasingly important role to achieve further computational gains. *The main focus of this class is how compilers can make your code more efficient and safe on modern (and near-future) processors*.

Background

- Class structure at UCSC is a little messed up
  - Typical prereqs are an undergrad compilers class
  - But this is a grad class with people coming in from many backgrounds
    - So we don’t want to require an undergrad compilers class

- For those of you who took CSE 110A:
  - Some material might be a review
  - Homeworks will be new though, so you will learn new things

- For those of you who have no compiler background:
  - First part of class will go over basics

Background

- Understand basics of parsing:
  - Regular expressions
  - Context free grammars (optional)
- Understand basics of programming languages
  - how are programs structured (e.g. basic blocks, expressions, statements)
- Comfortable using console coding (e.g. emacs or vim) and a terminal
  - Python
  - C/C++
- Some exposure to architecture, especially machine code (e.g. MIPS or X86 or RISCV)
- Comfortable with data structures and algorithms:
  - trees, hash tables, etc.

Modules

- There are 4 modules

- We will aim to spend roughly 2.5 weeks on each
  - We will be flexible. This is a small class and if we need to spend more time on a topic, we will!

- The last module will likely be cut short

- Structure
  - first 3 modules will be given lecture style
  - last module are more like a reading group/discussion

Modules

Module 1: Parsing Overview: This module will go over parsing at a high-level. This includes tokenizing, parsing context-free grammars, parser generators, and how to quickly create a parser for a simple programming language.

This is not a class on implementing parsers! Instead I want you to learn how to easily implement a compilers using parser generators!

Modules

Module 2: Analysis and Optimization: This module will go over different flow analysis (aka static analysis). We will discuss different AST traversals, SSA form, and applications such as identifying use-before-initialized errors.

Modules

Module 3: Parallelization and DSLs: We will discuss how compilers can be used to transform sequential code blocks into equivalent forms that can be executed in parallel. We will explore domain specific languages that further facilitate this automatic translation.

Modules

Module 4: Advanced Topics: We will read impactful papers in the area and discuss them. We will adjust based on class interest, but topics may include optimization evaluation and synthesis.

We will also make time in this module for guest speakers and final project presentations

Class Format

- in-person
- 65 minutes
- *Non-protected materials* are public
- *Protected materials* are on canvas
- *A few classes will be asynchronous due to travel, see the schedule.*

Class Format

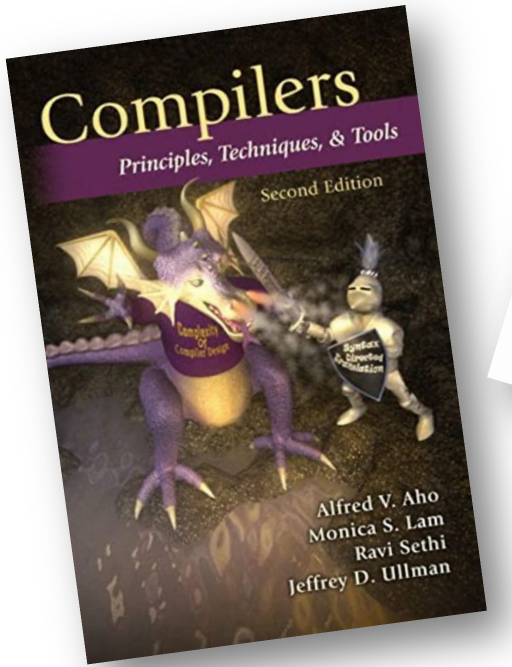
- In-class discussion questions on start of each lecture slide
- I will try to stay 10 minutes after, but we should go out into the hall
    - Not able to stay today because we have a guest arriving

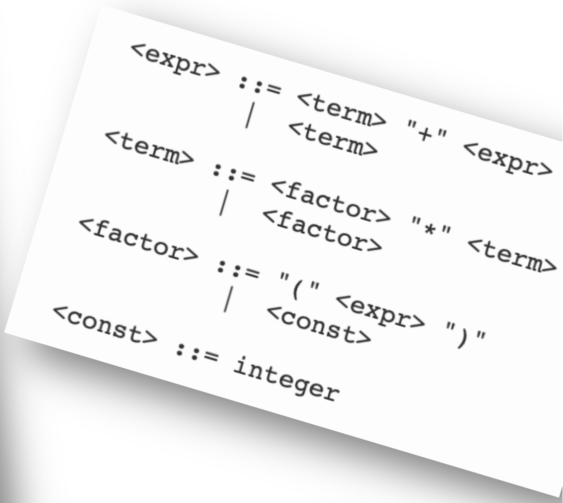
CSE211: Compiler Design

Oct. 1, 2020

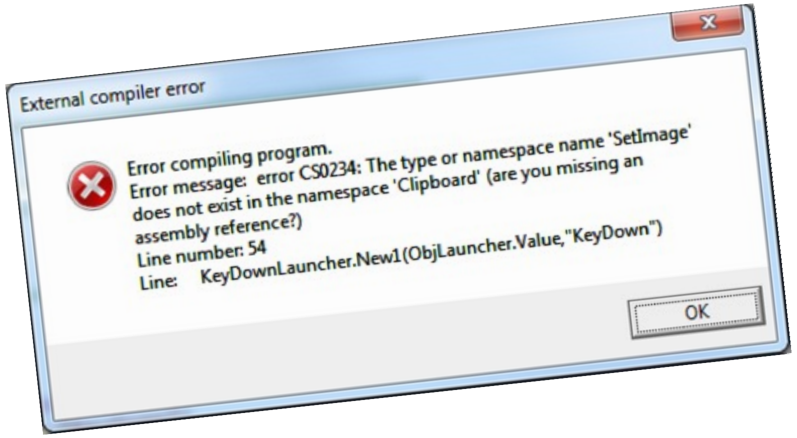
- **Topic**: Course Introduction

- **Questions**:
  - *What is a compiler?*
  - *What are some of your favorite compilers?*
  - *Have you ever built a compiler?*





```
<expr> ::= <term> "+" <expr>
         | <term>
<term> ::= <factor> "*" <term>
         | <factor>
<factor> ::= "(" <expr> ")"
           | <const>
<const> ::= integer
```



```
External compiler error

Error compiling program.
Error message: error CS0234: The type or namespace name 'SetImage'
does not exist in the namespace 'Clipboard' (are you missing an
assembly reference?)
Line number: 54
Line:    KeyDownLauncher.New1(ObjLauncher.Value,"KeyDown")
```

COMPILER INFRASTRUCTURE

Class Format

- We will use Piazza for discussions
  - I will check and try to reply within 24 hours to discussion questions
  - Not guaranteed to reply in off-hours or weekends
  - Please start class-wide discussions, link to interesting articles, etc. etc.
  - Please DO NOT publicly share homework solutions before the due date or ask for detailed help in the class-wide channels. Ask privately.
  - I will moderate.

Attendance

- Small class means lots of discussion! Please try your best to attend
- Please participate! In-person, Piazza discussions, interacting with your classmates, etc.
- In the past I have required attendance. This class is getting large enough that it is becoming difficult.
  - Please try to attend class. If attendance regularly drops below 50% then we might go back to requiring it.

Office Hours

- I will find 2 hours per week for office hours (I’ll have it set by Monday)
  - Remote or in-person
  - I will create a google doc sign up sheet with a zoom link
  - Room is E2 233

- Post clarifying questions as discussion topics on Piazza so that everyone can benefit!

Homeworks

One assignment for each module

- 2 weeks for each homework, posted midway through module.
- Each assignment has a suggested language. If you want to use a different language, please approve with me
  - Final solution must run in the docker and have exactly the same interface
  - Using a different language may require more documentation
  - But it can be a fun exercise (especially if you use a functional language!)
- Ask questions on Piazza, discuss concepts, visit office hours, etc.
- Start early!

Homeworks

To help with scaling the class:

- We will be doing homeworks in pairs
  - Each assignment, please find a partner.
  - Post on the discussion board if you’re not able to find one
  - You will need a different partner for each assignment
    - Come to class and get to know your classmates!

  - Please try to do some pair programming:
    - One person on keyboard and one typing

Tests

- 1 Midterm and 1 Final
  - Midterm is given on Oct. 30
  - Final is Tuesday, Dec. 12 from 8 AM to 11 AM

- 3 pages of notes allowed (not required!)

Paper Assignment

- Read 2 academic papers over the quarter. One during the first half and one during the second half.
- Approve your paper with me at least 1 week before it is due.
  - But you can work on it whenever (start early!)
- Write a 4 page double spaced review of each paper
- List of suggested papers:
  - https://docs.google.com/spreadsheets/d/1C5zMXC_RC94EmgmJVOliwYLylEyDUmK9KCO8oixaC3k
  - Happy for suggestions to expand it
  - Still needs to be approved
- If you are a grad student, find a paper related to your project/thesis!

Final Project

You can choose to do a final project instead of a final exam

- Final projects must be approved by me no later than 2 weeks before the end of the semester
- 6 page double spaced report on the project (due at the date of final)
- 10 minute presentation to the class (due day of class)
- If you are a grad student, I suggest thinking about a project you can incorporate into your thesis/project!

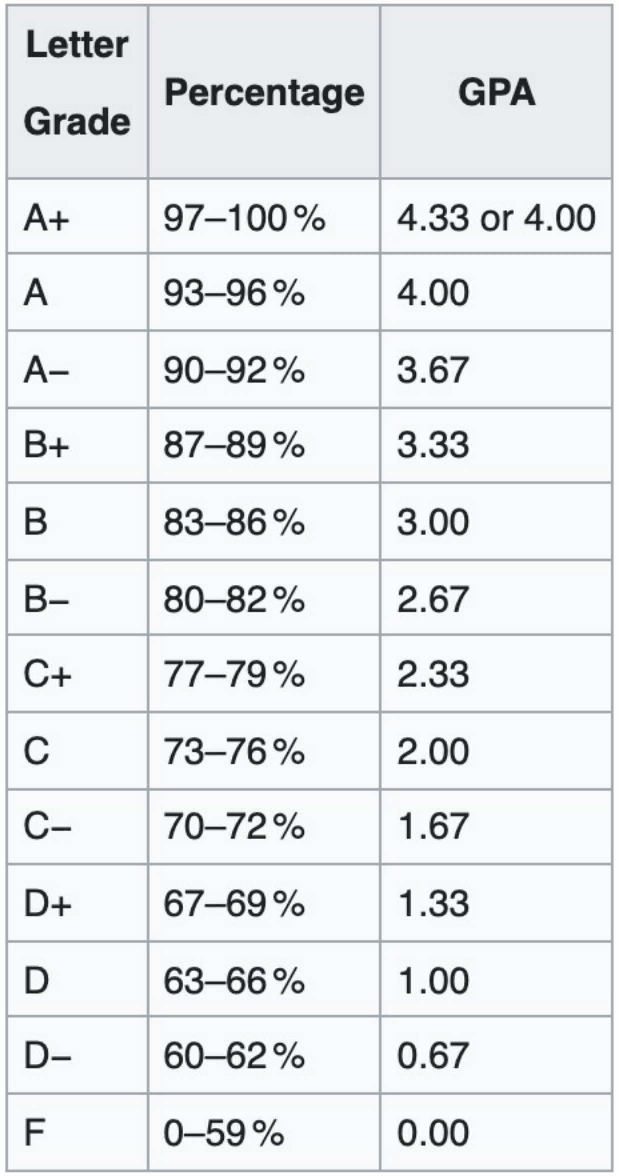
Final Project

Option to publish final project online:

https://sorensenucsc.github.io/CSE211-fa2022/projects.html

Rubric

- 40% Homework (10% each)
- 20% paper assignment (10% each)
- 10% Midterm
- 30% Final



Letter Grade	Percentage	GPA
A+	97–100 %	4.33 or 4.00
A	93–96 %	4.00
A–	90–92 %	3.67
B+	87–89 %	3.33
B	83–86 %	3.00
B–	80–82 %	2.67
C+	77–79 %	2.33
C	73–76 %	2.00
C–	70–72 %	1.67
D+	67–69 %	1.33
D	63–66 %	1.00
D–	60–62 %	0.67
F	0–59 %	0.00

From: https://en.wikipedia.org/wiki/Academic_grading_in_the_United_States

Resources

- Slides and lectures:
  - I will post the slides for each lecture before class starts

- Recommended textbooks:
  - **Engineering: A Compiler 2nd Edition**: unlimited online availability from the library. Links in Resources tab of webpage.

  - **Compilers: Principles, Techniques, and Tools 2nd Edition (Dragon Book)**: Limited availability from the library. Ask me for links, or to borrow physical/kindle editions

- Academic papers/blogs:
  - I will link to them as needed on the Resources/schedule site

TA

- Rithik Sharma
  - A PhD student working with me
  - Has a lot of experience with compilers!
- He will be hosting office hours as well
  - Will be determined soon!
- Please get to know him!

Walls of text incoming…

Academic Integrity

- One of the joys of university life is socializing and working with your classmates. I want you to make friends with each other and discuss the material. This is an advanced topics course; there is a high chance that your classmate will be your colleague throughout your career!

- **That said, I expect all assignments (homeworks, tests, paper reviews, presentations, final projects) to be your own original work (or your joint work in pairs for homework).**

- If you work with another classmate on an assignment, please mention this, e.g. in the comments of your code. If you use a figure you didn't create in a presentation, then it needs a citation. If you use code from an online source, it needs a citation, even if it is small! I won't dock points for cited code. Please review the universities policy on plagiarism

- This class has a zero tolerance policy on cheating. Please don't do it. I would much rather get a hundred emails asking for help than have to refer anyone for academic misconduct.

Academic Integrity in the world of ChatGPT

- LLMs are an amazing technology and we should learn how to utilize them to increase our productivity
- Because this is an advanced topics class, I doubt ChatGPT will be much help on homeworks.
  - If you use it, please discuss your usage in the report
  - For example, was it able to get the right answer? How close was it?
- Please don’t use it for writing
  - You can use it to refine your writing, but don’t copy and paste chunks of text from it.
  - In many cases, professors can tell these days.
- Very interested to discuss people’s experiences with this technology. There are interesting final projects ideas on LLMs (and maybe even a project about how to use it with compiler homework!)

Privacy

We will use the campus technology to record lectures

- Things you do or say will be recorded. I doubt that this will be an issue, but if you want me to remove any part of the recording, please just let me know.
- Canvas is secure™: other communications are less so
- Let’s do our best to protect our privacy and respect the privacy of others.

Disability Accommodation

UC Santa Cruz is committed to creating an academic environment that supports its diverse student body. If you are a student with a disability who requires accommodations to achieve equal access in this course, please submit your Accommodation Authorization Letter from the Disability Resource Center (DRC) to me by email, preferably within the first two weeks of the quarter. I would also like us to discuss ways we can ensure your full participation in the course. I encourage all students who may benefit from learning more about DRC services to contact DRC by phone at 831-459-2089 or by email at drc@ucsc.edu.

COVID Policy

- Please be courteous of others

- Notify me directly if you need to quarantine (or take time off for other covid-related issues) so we can plan

- We will work together and have a great quarter!

COVID Policy

- COVID continues to effect us in many ways. This quarter will surely have challenges related to this. I imagine people will continue to get sick, whether it is us, or people that we love. I hope we can all approach our interactions with empathy and understanding. I will do my best to accommodate the various individual challenges that may arise. Please communicate with me early and often!

- Currently, this is designated as an *in-person* class. It is **not** a hybrid class. As explained in the attendance section, I do expect you to plan on primarily attending in person. If you cannot attend in person for reasons related to COVID (e.g. if you or someone close to you gets sick), let me ASAP to discuss accommodations. I will do my best to provide lecture materials to you.

- If I am unable to attend (e.g. if I get COVID), then I will aim to teach remotely as long as I am feeling well enough.

Now back to something a little lighter…

LSD Seminar

- CSE-280O-01: LSD Seminar (co-organized with Lindsey Kuper)
  - Friday’s at Noon
  - (mostly) student speakers!
  - https://lsd-ucsc.github.io/lsd-seminar/2023fa/
- Topics will be highly relevant to this class; please consider joining!

Website tour

- Class website
- Canvas
- Other useful websites:
  - Godbolt

Made it through the syllabus…

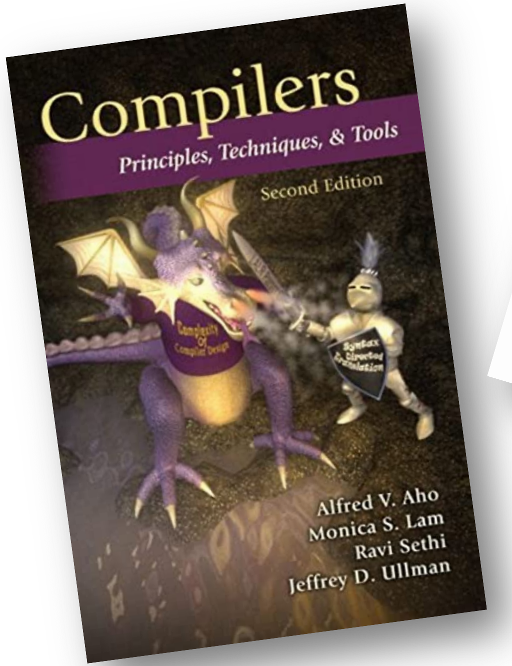
The slides should start to get more interesting now

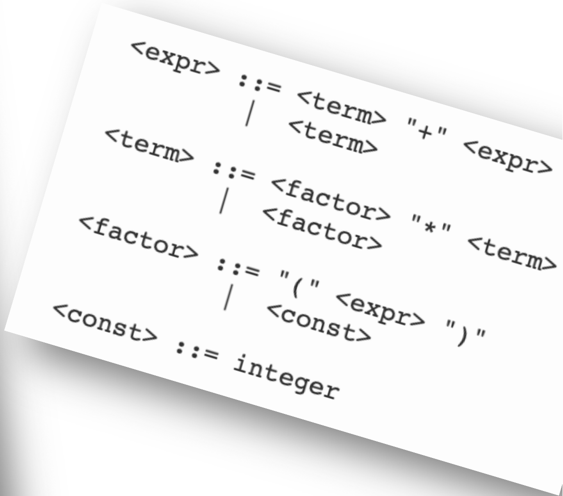
CSE211: Compiler Design

Sept. 22, 2022

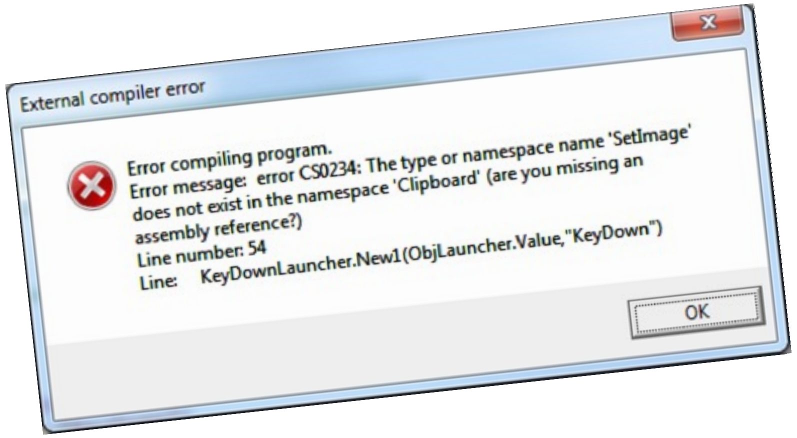
- **Topic**: Course Introduction

- **Questions**:
  - *What is a compiler?*
  - *What are some of your favorite compilers?*
  - *Have you ever built a compiler?*





```
<expr>   ::= <term> "+" <expr>
           | <term>
<term>   ::= <factor> "*" <term>
           | <factor>
<factor> ::= "(" <expr> ")"
           | <const>
<const>  ::= integer
```



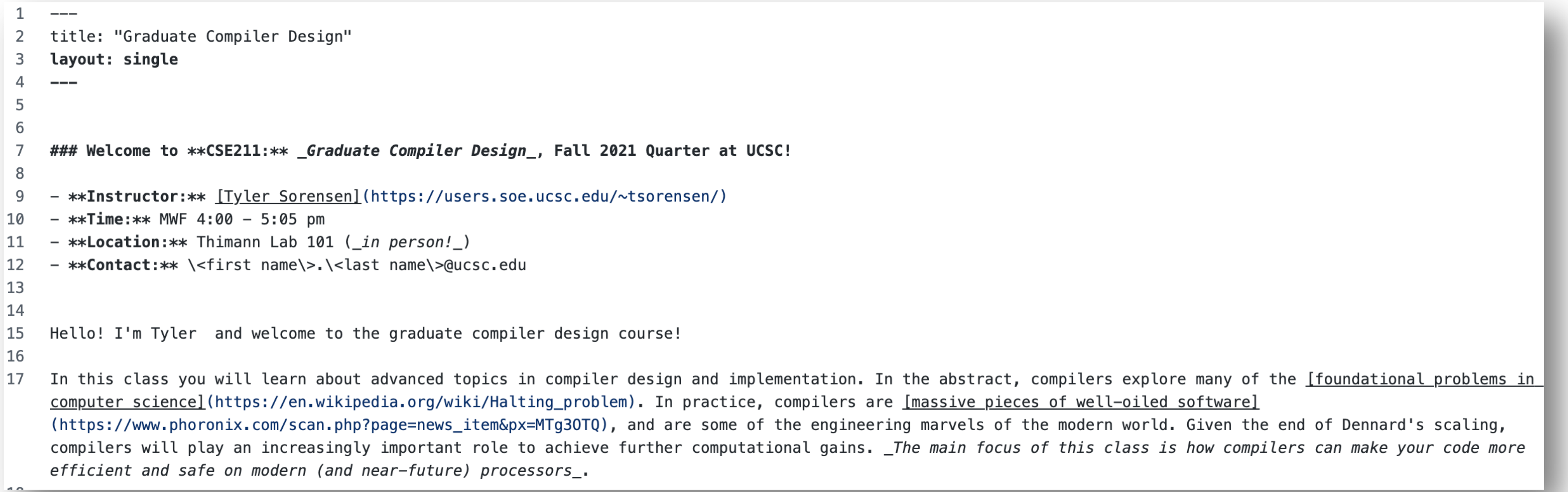
External compiler error

Error compiling program.
Error message: error CS0234: The type or namespace name 'SetImage' does not exist in the namespace 'Clipboard' (are you missing an assembly reference?)
Line number: 54
Line: KeyDownLauncher.New1(ObjLauncher.Value,"KeyDown")

LLVM
COMPILER INFRASTRUCTURE

What is a compiler?

What are some of your favorite compilers



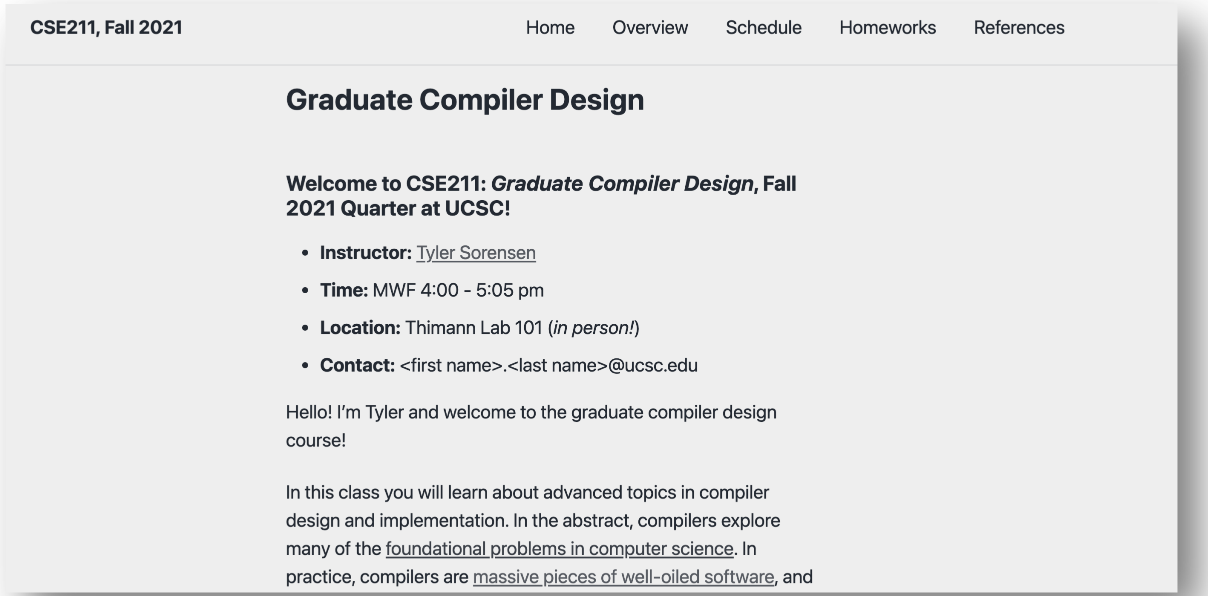
```
---
title: "Graduate Compiler Design"
layout: single
---


### Welcome to **CSE211:** _Graduate Compiler Design_, Fall 2021 Quarter at UCSC!

- **Instructor:** [Tyler Sorensen](https://users.soe.ucsc.edu/~tsorensen/)
- **Time:** MWF 4:00 - 5:05 pm
- **Location:** Thimann Lab 101 (_in person!_)
- **Contact:** \<first name\>.\<last name\>@ucsc.edu


Hello! I'm Tyler  and welcome to the graduate compiler design course!

In this class you will learn about advanced topics in compiler design and implementation. In the abstract, compilers explore many of the [foundational problems in computer science](https://en.wikipedia.org/wiki/Halting_problem). In practice, compilers are [massive pieces of well-oiled software](https://www.phoronix.com/scan.php?page=news_item&px=MTg3OTQ), and are some of the engineering marvels of the modern world. Given the end of Dennard's scaling, compilers will play an increasingly important role to achieve further computational gains. _The main focus of this class is how compilers can make your code more efficient and safe on modern (and near-future) processors_.
```



Graduate Compiler Design

Welcome to CSE211: *Graduate Compiler Design*, Fall 2021 Quarter at UCSC!

- **Instructor:** Tyler Sorensen
- **Time:** MWF 4:00 - 5:05 pm
- **Location:** Thimann Lab 101 (*in person!*)
- **Contact:** <first name>.<last name>@ucsc.edu

Hello! I'm Tyler and welcome to the graduate compiler design course!

In this class you will learn about advanced topics in compiler design and implementation. In the abstract, compilers explore many of the foundational problems in computer science. In practice, compilers are massive pieces of well-oiled software, and

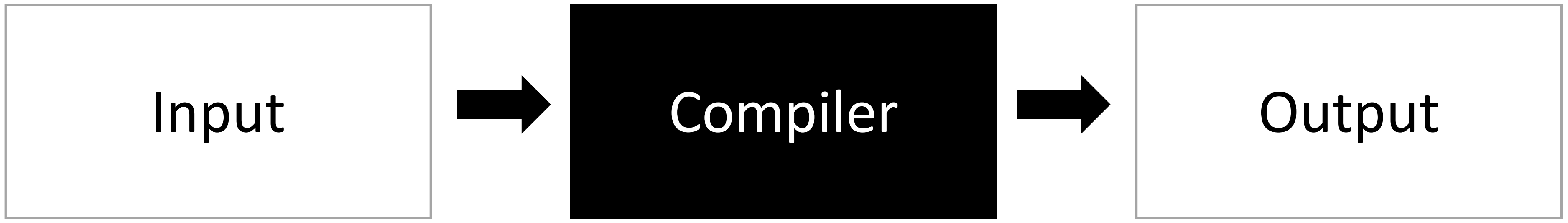
Building this website started with:

- Markdown to describe the page
- compiled with Jekyll to a static webpage
- static webpage is in HTML and javascript

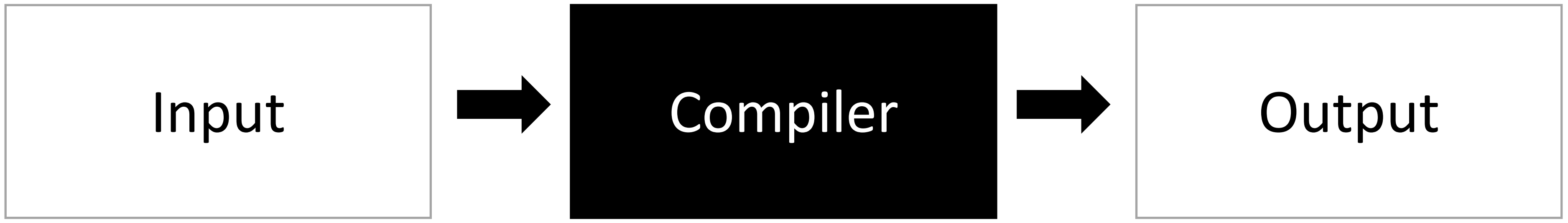
Canon

Have you ever built a compiler?

What is a compiler?



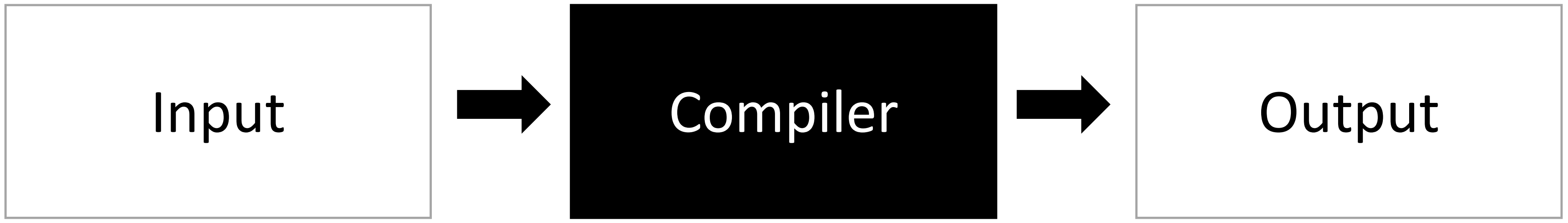
What is a compiler?



Strings belonging to language L

Strings belonging to language L’

What is a compiler?



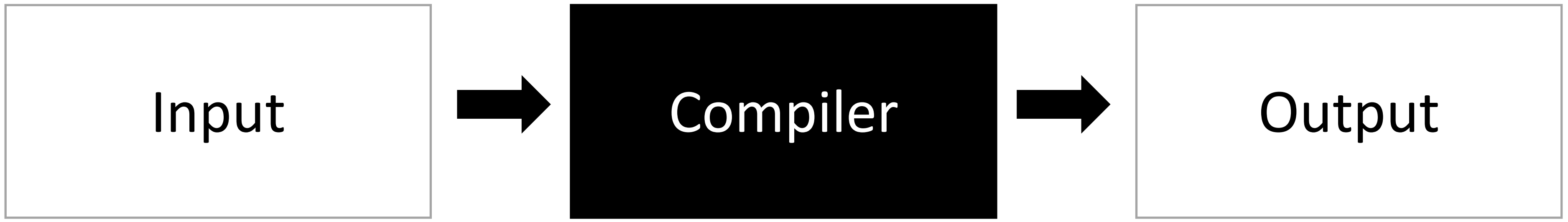
Strings belonging to language L

A series of statements in programming language L

Strings belonging to language L’

An executable binary file in an ISA language

What is a compiler?



Strings belonging to language L

Strings belonging to language L’

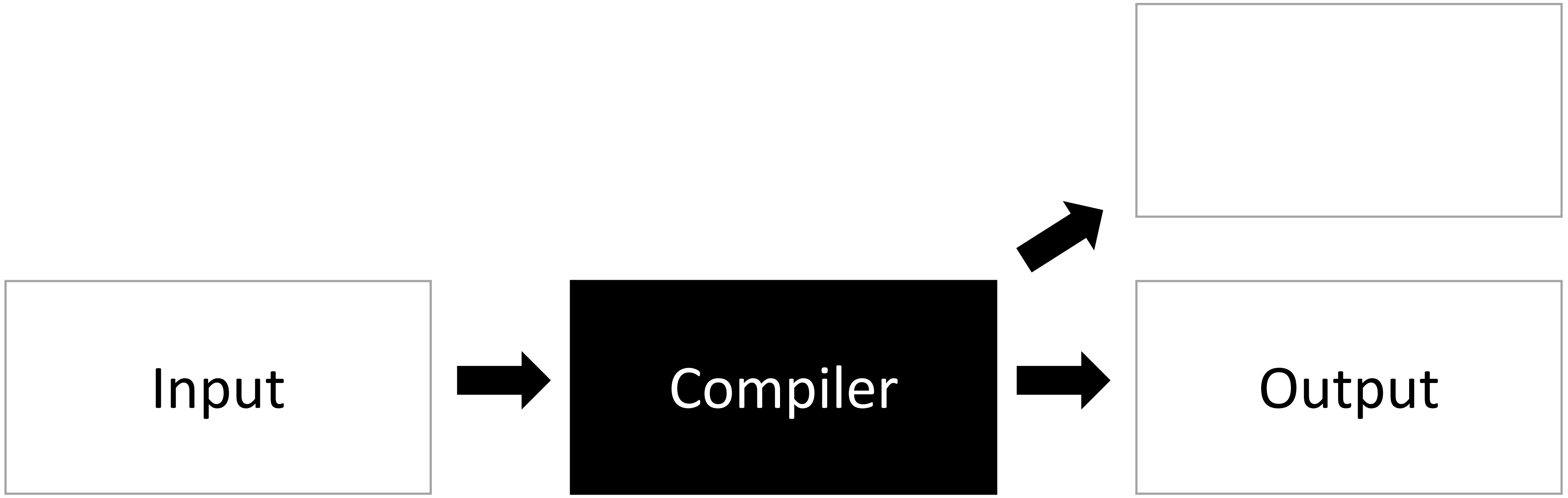
A series of statements in programming language L

An executable binary file in an ISA language

An x86 Binary executable

A program written in C++

What is a compiler?



Strings belonging to language L

A series of statements in programming language L

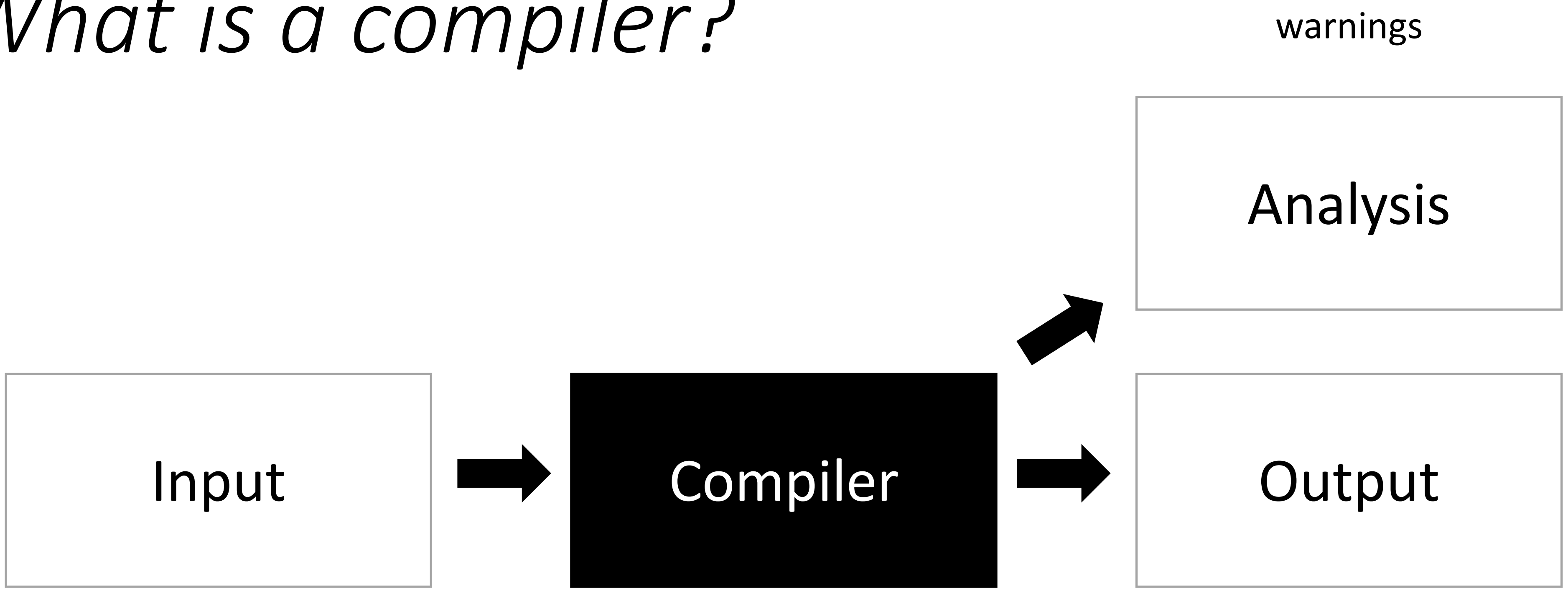
A program written in C++

Strings belonging to language L’

An executable binary file in an ISA language

An x86 Binary executable

What is a compiler?



Strings belonging to language L

A series of statements in programming language L

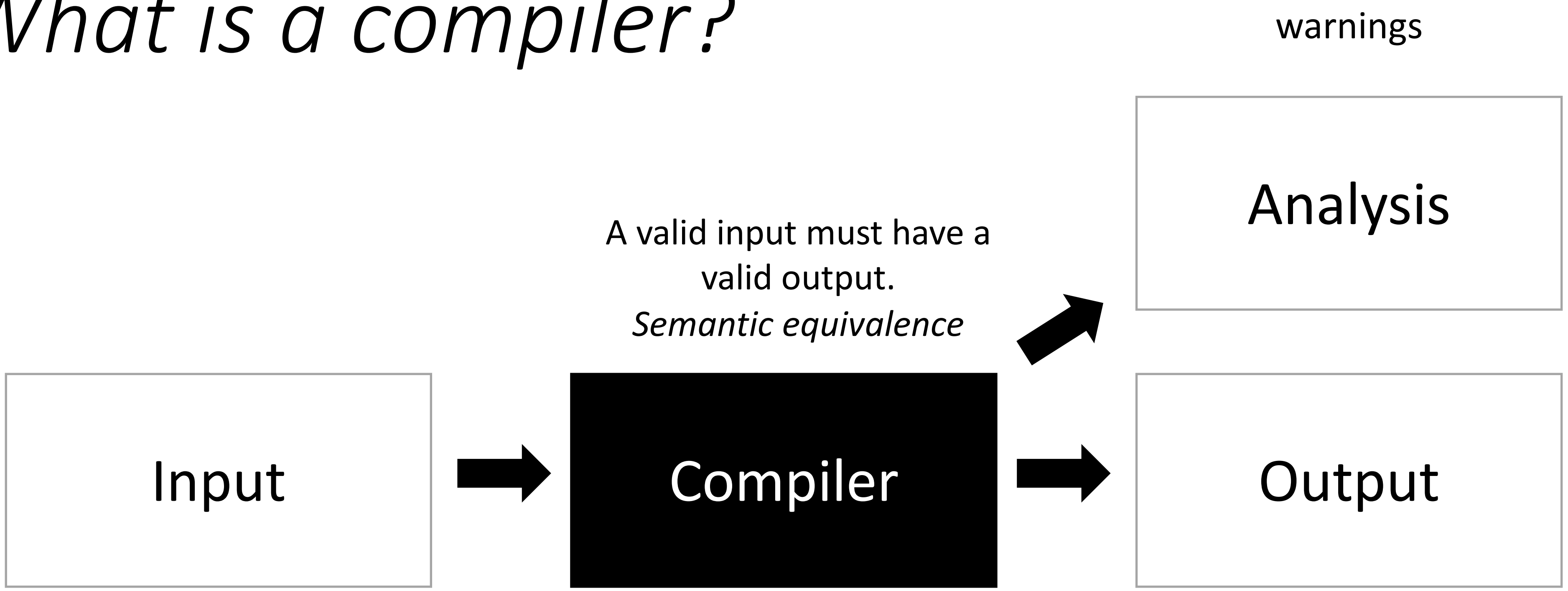
A program written in C++

Strings belonging to language L’

An executable binary file in an ISA language

An x86 Binary executable

What is a compiler?



A valid input must have a valid output.
Semantic equivalence

Strings belonging to language L

A series of statements in programming language L

A program written in C++

Strings belonging to language L’

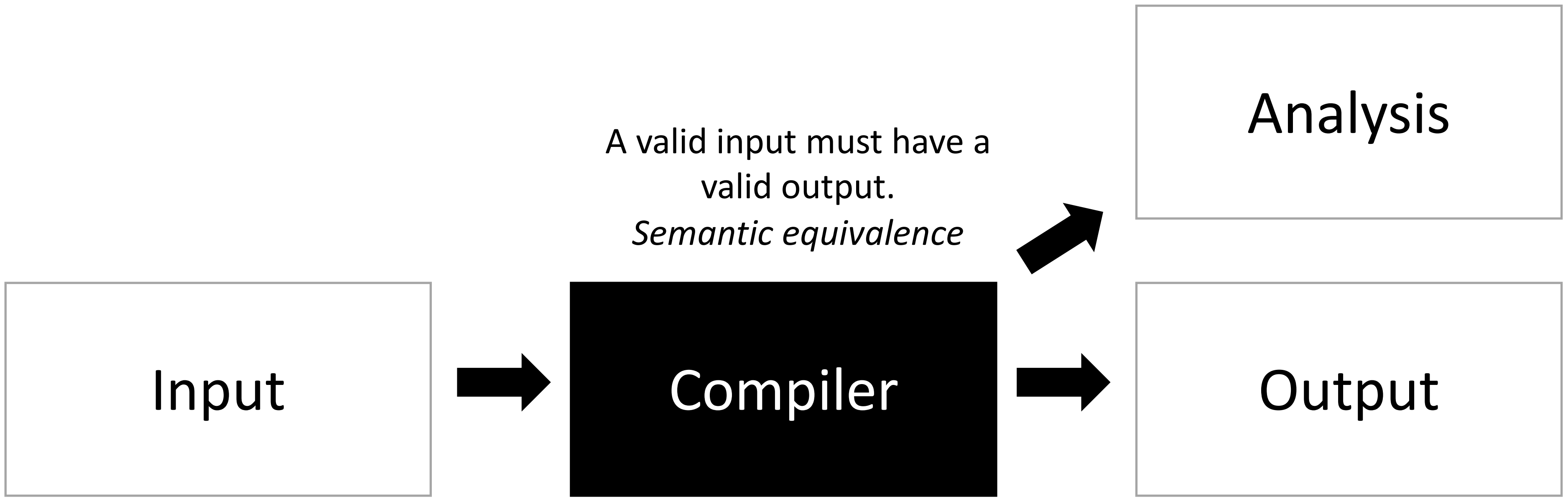
An executable binary file in an ISA language

An x86 Binary executable

What is a compiler?

warnings

A valid input must have a valid output.
Semantic equivalence



Strings belonging to language L

A series of statements in programming language L

A program written in C++

What can happen when the Input isn’t valid?

Strings belonging to language L’

An executable binary file in an ISA language

An x86 Binary executable

What can happen when the Input isn’t valid?

```
int my_var = 5;
my_var = my_car + 5;
```

Try running this through clang; you will get an error and a suggestion!

What can happen when the Input isn’t valid?

```
int my_var = 5;
my_var = my_car + 5;
```

```
int foo() {
  int *x = malloc(100*sizeof(int))
  return x[100];
}
```

What about this one?

Next class

- Tokenizing with regular expressions