CSCI E-95, Spring 2025: Compiler Design and Implementation

Prof. James L. Frankel
Harvard University

Version of 5:00 PM 1-Apr-2025

First Class Meeting on 1/28/2025

First Class Meeting Agenda

- Class website & Canvas, Zoom links
- Staff introductions
- Polls
- Student introductions
- Class website & class information
- Student actions: Order books, Say Hello!, Student Locations, HarvardKey, Cisco AnyConnect VPN, cscie95.dce.harvard.edu instance
- Course problem set overview
- Problem Sets 0 & 1
- Overview of Compiler & Review of the C Programming Language

Class Website, Canvas, Zoom Links

- Our **class website** is located at URL: **https://cscie95.dce.harvard.edu/spring2025/**

- Please participate in the live stream and ask questions verbally using **Zoom** available in **Canvas** (**https://canvas.harvard.edu/courses/150161**) under the **Zoom** menu
- In addition, questions may be asked textually using **Zoom's Chat facility**

Zoom

- You are encouraged to turn on your video feed
  - This allows the course staff to better determine if students seem puzzled and/or if they have questions

- Class and section meetings are recorded
  - Students who are unable to attend a meeting for any reason are able to view recordings later
  - It's still better to participate in the live session so that questions can be asked and answered
  - Many students find that reviewing material later to fully appreciate the details presented during class – even if they participated in the live class – is very helpful

Staff Introductions

- Professor
  - James “Jamie” Frankel
- Teaching Assistants
  - Mark Ford
  - Stephen Benjamin

Quick Polls

- Goals and Interest for Class Enrollment (Multiple Choice)
- Class Participation
- Section Participation
- Class Expectations (Multiple Choice)

- You can choose to answer the polls anonymously

Student Introductions

- Please tell us a little about yourself
  - Where you're located
  - What you do when you're not at Harvard
  - Your technical background
  - Your out-of-work/school hobbies

Tour of Class Website

- At the top there are alerts in red
- Quick Links
- Links for streaming and videos
- Info about midterm exam, prerequisites, learning objectives, overview, bibliography, instructors and section, Ed Discussion wiki/forum, Say Hello!, your location, git & GitHub, grading, accessibility, plagiarizing, use of generative AI, publishing/distributing course materials, MIPS & SPIM, outline/approximate schedule, agenda for the upcoming class, slides used in class, questionnaire & problem sets, assorted links, example programs used in class, grammar for our C Programming Language, link to the section home page

Meeting Times

- Section meets on Tuesdays in Room L01, 53 Church Street, Harvard Square, Cambridge, Massachusetts from 6:45 PM to 7:45 PM Eastern Time (ET) and in Zoom using the **Section: HELIX Classroom** room
  - This is immediately before class meets

- Class meets on Tuesdays in Room L01, 53 Church Street, Harvard Square, Cambridge, Massachusetts from 8:00 PM to 10:15 PM Eastern Time (ET) and in Zoom using the **Class: HELIX Classroom** room
  - Elongated class meeting time

- I will attempt to include a break during the class meetings *(but no guarantee because of scope of material to be presented)*

Section

- Required part of class

- Very important
  - Discusses concepts & issues that are not covered in class
  - Often gives a sketch of algorithms and approaches to be used in solving the problem sets
  - Adds enrichment on topics discussed in class/lecture

  - Great forum for a more interactive dialog

  - Is live streamed and also recorded

Class Website Review

- Questions?
  - Questions are always welcomed
    - Any questions now?
  - If there is limited time to answer a questions, I'll let you know

- Review of Website
  - Midterm exam
  - Prerequisites
  - Overview
  - Required and optional books
  - The daily agenda (these slides) and all slides used in class
  - …

- Order books, if you have not already done so
  - **Compilers: Principles, Techniques, and Tools**, **2nd Ed.** by Alfred V. Aho, Monica S. Lam, Ravi Sethi, Jeffrey D. Ullman
  - **C: A Reference Manual**, **5th Ed.** by Samuel P. Harbison and Guy L. Steele, Jr.

- Somewhat limited online access is available to all of our books through our Library Reserves link in Canvas

Required Readings

- Refer to the Approximate Schedule section of the course website for required readings to be completed before each class meeting

Say Hello!, Student Locations, Harvard Key, Using cscie95.dce.harvard.edu

- Submit a video in Canvas under **Discussions** as a reply to my “Say Hello!” topic
- Please post your primary location using **Student Locations** facility in Canvas
- Ensure that your **Harvard Key** is established
- Ensure that you are able to VPN into Harvard using vpn.harvard.edu and Cisco Secure Client/AnyConnect VPN
- Ensure that you have an account on our cscie95.dce.harvard.edu AWS instance
  - **Once your VPN connection is established**, login to cscie95.dce.harvard.edu using SSH/SFTP (SecureCRT & SecureFX) with your HarvardKey NetID as your login name and your HarvardKey password as your password
  - If you are unable to login to cscie95.dce.harvard.edu, you may need to synchronize your HarvardKey password by using a browser to visit https://key.harvard.edu/manage-account and then clicking on “Synchronize Password >” and following the instructions on the next screen

g.harvard.edu e-mail Address

- If you're interested, you can get a g.harvard.edu account that will give you a *logname*@g.harvard.edu e-mail address and access to Google Apps for Harvard
  - Get started at http://g.harvard.edu/
  - Note: Please be aware that when you claim your g.harvard account that g.harvard will become your primary Harvard e-mail address. All official communication from Harvard will be sent to your new g.harvard address and your g.harvard account will become your HarvardKey login name.

Class Discussion Group: Ed Discussion

- Ask all non-personal questions in Ed so the whole class can benefit from the answers
  - Ed can be found in Canvas by following the Ed Discussion link (https://canvas.harvard.edu/courses/150161/external_tools/109187?display=borderless)
  - Students are welcome to answer questions there, too
  - Personal questions should be sent to the course staff via e-mail
    - If appropriate, include all three course staff members in e-mails to allow the fastest reply
  - All registered students should already be in our Ed group

- To **receive immediate notifications about new threads:** in our Ed group, in the upper right, click on the **Account** icon, then…
  - Then, in the pulldown menu, Select **Settings**
  - Click on **Notifications**
  - Change the frequency to receive e-mails about new threads to **Instant**
  - Also, ensure that all other **Notification Emails** are active

Midterm Exam

- Our **midterm exam** will be available starting at **8:00 PM ET on March 25, 2025**
  - The exam must be started within 24 hours of the date & time above
  - The exam is three hours in length
  - The exam will be administered under Proctorio
- There will be **no class meeting on March 25th**, but **section will still be held on March 25th**
  - No topics relevant to the midterm exam will be discussed in that section meeting

Problem Set Overview

- Problem Set 0: the course questionnaire, fix-this-program & word-count
- Problem Set 1: lexer for our C Programming Language subset
- Problem Set 2: parser for our compiler
- Problem Set 3: symbol table management system
- Problem Set 4: semantic analysis
- Problem Set 5: IR generation
- Problem Set 6: MIPS assembly language code generation
- Final Project: optimizations

Five Free Late Days

- Please don’t use any of your five free late days early in the class
- Because the later problem sets are built upon earlier problem sets, the free late days are more valuable later in the semester
- Also, the larger problem sets are worth more points and take much more time to complete

Problem Set 0

- Complete Problem Set 0
  - Establish a GitHub account
  - Install git as described on the section website (https://cscie95.dce.harvard.edu/spring2025/section/index.html)
  - Modify the course questionnaire with your personal answers
  - Fix warnings and errors in fix-this-program on the cscie95 instance
  - Write the word count program
  - Create a branch named "problem-set-0", create a merge request, add the appropriate comment
- Due this coming Sunday night, February 2, 2025 at midnight ET

Problem Set 1

- Present **Problem Set 1**
  - Due at midnight ET on Sunday night, February 9th, 2025

- The code that you write for your compiler can take advantage of ***all of the syntax and features*** of the latest ISO/ANSI C Programming Language

Lying to Students

- I will lie to you this semester

Lying to Students

- I will lie to you this semester
  - There are too many details to give the whole truth
  - That is the only way we can make reasonable progress through the material

- By the end of the semester, all lies will be fully corrected

Non-academic Class Activities

- Encourage a student community

- If interested, students are welcome to gather with us after each class for dinner in Cambridge – including tonight!
  - Opportunity for students to socialize in an informal setting outside of class
  - Discussion/conversation/sharing after class
  - Not relevant to class

- Other non-class activities
  - Class ski trip in January
  - Sailing trip during the summer

Class Break

- Let’s take a 10-minute break
- You’re welcome informally interact during the break

Today's New Material

- Cover **Overview of Compiler** slides
- Cover **Review of the C Programming Language** slides through **Language: Operators** slide **#5**

Second Class Meeting on 2/4/2025

- Questions?
  - Problem Set 0
  - Problem Set 1
  - Section or lecture
  - Anything else

Problem Set 1

- Keep code in src/compiler directory
- Accept the same command line interface as does the skeleton code in the class repository
- You are not required to keep the same structure that is used by the skeleton compiler, but it is probably the correct approach unless you have a good reason to deviate from its structure
- The link to Manual for Lexical Analysis with Flex, Flex 2.6.2 on the class website under **Open software and documentation** has a great deal of useful information

- Reminder that **Problem Set 1 is due at midnight ET on Sunday night, February 9, 2025**

Problem Set 2

- Present **Problem Set 2**
  - Due at midnight ET on Sunday night, February 23rd, 2025
- Show the grammar for our subset of the C Programming Language on the class web site

- **Problem Set 2 is the most time-consuming problem set in the course**
  - Start early
  - Ask questions

- We'll present information on yacc/bison next week

New Material for this Week (1 of 3)

- Cover new slides
  - **Lexical Analysis**
- Example
  - Construct an NFA from a regular expression using the McNaughton-Yamada-Thompson algorithm ***exactly***
  - Construct a DFA from an NFA using the Subset Construction
  - Cover new slides
    - **Example of Regex, NFA, DFA**

New Material for this Week (2 of 3)

- Cover new slides
  - **Using Lex**
- Example
  - Show **lexer-standalone.lex** & **lexer.c**
  - Run **make standalone**
  - Demonstrate **./lexer**

New Material for this Week (3 of 3)

- Cover more new slides
  - **Review of the C Programming Language**
    - Continue with the **Language: Operators** slide **#5** with the **Observation** through the **Logical not !** operator in **Unary Prefix Operators** slide **#11**

Third Class Meeting on 2/11/2025

Third Class Meeting Agenda

- Questions and Comments
- Administrivia
  - Start Work on Problem Sets Early
  - Section
  - Lectures & Section Meetings
  - Office Hours
  - Overall Compiler Directions
- Aho, Lam, Sethi, Ullman Compilers Textbook
- Review of Current Problem Set Status
- Syntax Analysis
  - Show **recursiveDescentParser.c**
- Using YACC/Bison
  - Show **lexer.lex** & **parser.y**
- Parse Tree, AST, and Type Tree
- Review of the C Programming Language (continued)

Questions and Comments

- Section
- Last week's class or any previous classes
- Problem Sets 0 through 2
- Using the class cscie95.dce.harvard.edu instance
- Ed Discussion threads
- Readings
  - Harbison & Steele
  - The Dragon book
- Anything else

Start Work on Problem Sets Early

- In order to ensure that your questions are answered in a timely manner, start work on problem sets early
- The course staff are often quite prompt in answering Ed questions, but there is no guarantee of immediate responses

Section

- Required part of class
- Section meets immediately before class on Tuesdays from 6:45 PM to 7:45 PM Eastern Time (ET) and in Zoom using the **Section: HELIX Classroom** room

- Very important
  - Discusses concepts & issues that are not covered in class
  - Often gives a sketch of algorithms and approaches to be used in solving the problem sets
  - Adds enrichment on topics discussed in class/lecture

  - Great forum for a more interactive dialog

  - Is live streamed and also recorded

Lectures & Section Meetings

- Watch both the lecture and section videos
- All lecture and section videos are recorded
- Many questions are answered in lecture and section

Office Hours

- Although attendance at office hours is not required, there are always interesting questions asked and detailed answers furnished

- Many students attend office hours without asking any questions so that they hear other students' questions and answers from the course staff
  - That is to say, "lurking" is always encouraged in office hours

Overall Compiler Directions

- Base your compiler on the skeleton compiler in the class repository
- Be sure to support the required command line interface
- The Problem Set descriptions are very detailed
  - They try to answer all questions and fully-specify the assignment
  - The relevant sections from Harbison & Steele are also needed to complete the problem sets
  - Many students find it helpful to restate the PS descriptions as an implementation checklist
- Follow all posts and replies on Ed
  - They will answer questions about class and problem sets

Aho, Lam, Sethi, Ullman Compilers Textbook

- Read the assigned textbook readings
- Do the practice problems in the textbook
- Even though we are not assigning textbook problem sets, you are responsible for the readings
  - Useful for the programming problem sets
  - Necessary for the midterm exam

Problem Set 1

- **Problem Set 1**
  - Was due at midnight Eastern Time on Sunday, February 9th, 2025

- I have made some minor clarifications to Problem Set 1 in order to answer some questions asked in Ed or in Office Hours

- Everyone should have already completed PS1

Problem Set 2 (1 of 2)

- **Problem Set 2**
  - Due at midnight Eastern Time on Sunday, February 23rd, 2025

- Problem Set 2 is the most time-consuming problem set in the course
  - Start early
  - Ask questions on Ed
  - Attend section
  - Attend office hours, if possible
  - Don’t postpone asking questions

Problem Set 2 (2 of 2)

- Follow the grammar on the class website
- Every node in the AST should have a node type
  - Which operator/statement?
  - Which/how many children?
  - etc.
- Don't create unnecessary nodes
  - No node for parenthesized expression
  - No node for a single non-terminal
  - No node for a choice of non-terminals (such as, a or b or c)
  - No node for terminals (in most cases)
  - etc.
- Create identifier nodes to store identifiers as strings in the AST
- Create literal nodes to store literals as values in the AST
  - Integer literals as integral values with associated type
    - Each type is represented by an enumeration constant from a single enumerated type (for now)
  - String literals as strings (with a string length, if the string has each escape character translated into its actual character)

New Material for this Week (1 of 4)

- Cover new slides
  - **Syntax Analysis**
- Example
  - Show **recursiveDescentParser.c**
  - Run **make recursive**
  - Demonstrate **./recursiveDescentParser**

New Material for this Week (2 of 4)

- Cover new slides
  - **Using YACC or Bison**
- Example
  - Show **lexer.lex** & **parser.y**
  - Run **make parser**
  - Demonstrate **./parser**

New Material for this Week (3 of 4)

- Cover new slides
  - **Parse Tree, AST, and Type Tree**
  - Just look at the Parse Tree and AST portions of these slides (*i.e.,* ignore the Type Tree slide for now)

New Material for this Week (4 of 4)

- Cover more new slides
  - **Review of the C Programming Language**
    - Continue from after the **Logical not !** operator in **Unary Prefix Operators** slide **#11** through the **Precedence(§7.2.1)** slide **#19**

Fourth Class Meeting on 2/18/2025

Fourth Class Meeting Agenda

- Questions and Comments
- Precedence Examples
- Current Problem Set Status
  - PS2
  - Present PS3
- Spring Break
- Midterm Exam
- Symbol Table Management
- Parse Tree, AST, and Type Tree
  - Present the Type Tree
- Type Checking
- Review of the C Programming Language (continued)

Questions and Comments

- Section
- Last week's class or any previous classes
- Problem Sets 0 through 2
- Lexer
- Parser & AST construction
- Our grammar
- Review of the C Programming Language
- The class cscie95.dce.harvard.edu/spring2025 instance
- Ed Discussion threads
- Readings
  - Harbison & Steele
  - The Dragon book
- Anything else

Precedence Examples

- Two C code examples
  - k = i++, j++;
  - k = (l = i++), (i + j++);

- The precedence of the comma (sequential evaluation) operator is lower than the precedence of the assignment operators – in fact, the comma operator has the lowest precedence of all C operators (see H&S Page 205, Table 7-3)
  - Therefore, the value assigned to k is the result of evaluating the i++ expression and is not affected by the right operand of the comma operator
  - The value of the entire comma expression is the result of evaluating the right operand of the comma expression, but that result is not used; therefore, the second example is parenthesized as:
    - (k = (l = (i++))), (i + (j++));

- This expression assigns to m the result of evaluating the right operand of the comma operator:

  m = (k = (l = i++), (i + j++));

Problem Set 2 (1 of 3)

- **Problem Set 2**
  - Due at midnight Eastern Time on this coming Sunday, February 23rd, 2025
  - You should all have already made significant progress working on PS2
  - Problem Set 3 is based on the AST from PS2

- Problem Set 2 is the most time-consuming problem set in the course
  - Ask questions on Ed
  - Attend section
  - Attend office hours, if possible
  - Don’t postpone asking questions

  - Ask questions now

Problem Set 2 (2 of 3)

- Follow the grammar on the class website
- Every node in the AST should have a node type
  - Which operator/statement?
  - Which/how many children?
  - etc.
- Don't create unnecessary nodes
  - No node for parenthesized expression
  - No node for a single non-terminal
  - No node for a choice of non-terminals (such as, a or b or c)
  - No node for terminals (in most cases)
  - etc.
- Create identifier nodes to store identifiers as strings in the AST
- Create literal nodes to store literals as values in the AST
  - Integer literals as integral values with associated type
    - Each type is represented by an enumeration constant from a single enumerated type (for now)
  - String literals as strings (with a string length, if the string has each escape character translated into its actual character)

Problem Set 2 (3 of 3)

- Show (Parse Tree and) AST starting from **decl** and derived from
  - int a, b, c;

- Refer to the grammar on the class website

Problem Set 3

- Present **Problem Set 3**
  - Due Sunday, March 9th, 2025 at midnight Eastern Time

- Problem Set 3 involves building symbol tables & type data structures

Spring Break

- Spring Break is Monday, March 17th, 2025 through Sunday, March 23rd, 2025

Midterm Exam

- Our Midterm Exam begins on Tuesday, March 25th, 2025 at 8 PM ET
- It's three hours in duration and can be started anytime from March 25th, 2025 at 8 PM ET to March 26th, 2025 at 7:59 PM ET

New Material for this Week (1 of 4)

- Cover new slides
  - **Symbol Table Management**
  - Go over symbolTables.c

New Material for this Week (2 of 4)

- Cover new slides
  - **Parse Tree, AST, and Type Tree**
  - Let’s examine the Type Tree slide
  - See how the Type Tree is derived from the AST for a decl

New Material for this Week (3 of 4)

- Cover new slides
  - **Type Checking**
  - Cover through the **Minimum Integer Precision and Range (§5.1.1, p. 125 & Table 5-2, p. 127)** slide **#15**

New Material for this Week (4 of 4)

- Continue with the C Programming Language slides
  - **Review of the C Programming Language**
    - Continue from the **Overloading (§4.2.4)** slide **#20** through the **Language: Declarations (§4)** slide **#29**

Fifth Class Meeting on 2/25/2025

Fifth Class Meeting Agenda

- Questions and Comments
- Current Problem Set Status
  - PS2
  - PS3
- Spring Break and Midterm Exam
- Symbol Table Management
- Table-Driven Top-Down Parsing
- Type Checking
- Review of the C Programming Language (continued)

Questions and Comments

- Section
- Last week's class or any previous classes
- Problem Sets 0 through 3
- Lexer
- Parser & AST construction
- AST, Type Tree, Symbol Table
- Our grammar
- Review of the C Programming Language
- Ed Discussion threads
- Readings
- Anything else

Problem Set 2

- **Problem Set 2**
  - Was due at midnight Eastern Time on last Sunday, February 23rd, 2025

- How is everyone doing with PS2?
- How much of our C language can your compiler accept and emit the resulting AST using your pretty-printer?
  - When passed through your compiler again, does all of your pretty-printed code produce the exact same code?

Problem Set 3

- **Problem Set 3**
  - Due Sunday, March 9th, 2025 at midnight Eastern Time

- Problem Set 3 involves building symbol tables & type data structures
  - Symbol tables
    - Other names
      - Identifier resolution to entry in correct symbol table
      - Replace identifier in AST with pointer to entry in correct symbol table
    - Statement labels
    - String table
  - Type tree
    - Each other names symbol table entry must have an appropriate type tree attached to it
  - Create all needed pointers to allow traversal both inward and outward among symbol tables

Spring Break

- Spring Break is Monday, March 17th, 2025 through Sunday, March 23rd, 2025

Midterm Exam

- Midterm Exam
  - Four weeks from today

  - Earliest start time is 8:00 PM Eastern Time on Tuesday, March 25th, 2025
  - Must be started by 7:59 PM Eastern Time on Wednesday, March 26th, 2025
  - Maximum exam duration is three hours

  - Administered by Proctorio/Canvas
    - No books, notes, computers, etc.
    - Canvas needed for access to slides

  - May ***not*** be taken in class
    - Consistency for all students
    - Controlled access to slides

Inward/Downward Edges among **Symbol Tables**

- Inward/Downward edges are useful for printing all symbol tables and for performing other operations
  - For example, we will need to traverse all symbol tables within each procedure/function to:
    - Determine the size of the stack frame required
    - Determine the stack offset for each automatic (stack-based) variable

Outward/Upward Edges among **Symbol Tables**

- Outward/Upward edges are useful for searching through symbol tables for the appropriate scope in which an identifier is declared
  - For example, when encountering a reference to an identifier, a search must be conducted starting with the current block to outermore blocks, to the procedure scope, and finally to the file scope to find the declaration for the identifier

Outward/Upward Edges in **AST**s

- Upward/Outward edges are useful in the AST when performing some operations in your compiler
  - For example, searching for the innermost matching language construct for **break** and **continue** statements

New Material for this Week (1 of 3)

- **Table-Driven Top-Down Parsing**

New Material for this Week (2 of 3)

- Type checking slides
  - **Type Checking**
  - Continue with the **Some Additional Type Names** slide **#18**

  - Skip over the **Numeric Encodings** slides
  - Skip over the **Character Encodings** slides

New Material for this Week (3 of 3)

- Continue with the C Programming Language slides
  - **Review of the C Programming Language**
    - Continue from the **Scope (§4.2.1)** slide **#30** through the **Extent (or Lifetime or Storage Duration) (§4.2.7)** slide **#35**

Sixth Class Meeting on 3/4/2025

Sixth Class Meeting Agenda

- Questions and Comments
- Daylight Saving Time
- Harbison & Steele Readings
- Current Problem Set Status
  - PS2
  - PS3
  - PS4
- Spring Break
- Upcoming Midterm Exam including Proctorio
- Upcoming Dates
- Type Checking
  - Review **Type Checking** slides from last week
  - Signed and unsigned types in a comparison operator
- Complete the **Review of the C Programming Language** slides (continued)

Questions and Comments

- Section
- Last week's class or any previous classes
- Problem Sets 0 through 4
- Lexer
- Parser & AST construction
- AST, Type Tree, Symbol Table
- Our grammar
- Review of the C Programming Language
- Ed Discussion threads
- Readings
- Anything else

Daylight Saving Time

- **Daylight Saving Time begins at 2 AM on this coming Sunday, March 9th in Massachusetts**
- Clocks will be set forward one hour
- If you are a living in a different time zone, please adjust your schedule as needed and set your time zone in Canvas

Harbison & Steele (1 of 2)

- Even though no specific readings from **Harbison & Steele** are assigned in the syllabus, everyone should be reading and referring to H&S on a regular basis
  - Harbison & Steele is the only reliable source for specifications on type checking for Problem Set 4

- Familiarity with Chapters 1, 2, and 4 through 9 is required and assumed

Harbison & Steele (2 of 2)

- The grammar is not sufficient to determine if a program is valid
- There are many examples where the grammar accepts invalid programs
- Each section of H&S contains constraints that must be checked to ensure that a valid program exists

Problem Set 2

- **Problem Set 2**
  - Was due at midnight ET on Sunday, February 23rd, 2025
- Problem Set 2 involves building the AST
- If you don’t yet have PS2 completed, please send e-mail to all course staff and let us know what’s up
  - We can help resolve your issues

Problem Set 3

- **Problem Set 3**
  - Due on this coming Sunday, March 9th, 2025 at midnight ET

- Problem Set 3 involves building symbol tables & type data structures
  - Symbol tables
    - Other names
      - Identifier resolution to entry in correct symbol table
      - Replace identifier in AST with pointer to entry in correct symbol table
    - Statement labels
    - String table
  - Type tree
    - Each other names symbol table entry must have an appropriate type tree attached to it
  - Create all needed pointers to allow traversal both inward and outward among symbol tables

Problem Set 4

- **Problem Set 4**
  - Will be due Sunday, March 30th, 2025 at midnight ET

- Problem Set 4 involves semantic analysis/type checking

- Present **Problem Set 4**

Spring Break

- Spring Break is Monday, March 17th, 2025 through Sunday, March 23rd, 2025

Midterm Exam

- Our **Midterm Exam begins in three weeks** on March 25th, 2025 at 8:00 PM ET
  - There will be **no class meeting on March 25th**, but **section will still be held on March 25th**
    - No topics relevant to the midterm exam will be discussed in that section meeting

- Ask any questions relevant to the midterm exam in section, in class, in office hours, or on Ed before noon ET on the day of the exam

- May ***not*** be taken in class
  - Consistency for all students
  - Controlled access to slides

Midterm Exam

- Exam is **three hours in duration**
  - The exam is graded with a **maximum score of 120**
  - The problems are scored in points where **a point is weighted to be approximately one minute** of exam answer duration
  - But, the exam is time-consuming – we assume that you will take all three hours
- Earliest starting time is 8:00 PM ET on Tuesday, March 25th, 2025 – **three weeks from today**
- Latest starting time is 7:59 PM ET on Wednesday, March 26th, 2025
- The exam will be available through **Canvas** under the **Quizzes** tab
  - The exam will be administered under **Proctorio**
  - **Either Google Chrome, Microsoft Edge, Brave, or Opera must be used as the browser** for the exam – only these browsers are acceptable to Proctorio
  - Proctorio requires that a **Proctorio Extension** be installed into your browser
- No books may be used during the exam, but all slides from class will be available through the **Files** menu item in Canvas (a link to the **Files** page of Canvas is in the exam Instructions)
  - Nothing else can be used during the exam: no notes, no electronics other than the computer being used for the exam, no cell phones, no talking, no communication
  - But, nothing else is needed!

The Proctorio Setup Quiz

- *As soon as possible,* take the **Proctorio Readiness Quiz** that is available in **Canvas** under **Quizzes**
- Really **carefully read all of the instructions**
  - **The instructions are the same as for the real Midterm Exam**
  - **We are not able to deal with issues caused by not following the instructions!**
- Before taking the Midterm Exam, contact the course staff *as soon as possible* if you have any problems taking the **Proctorio Readiness Quiz**

Midterm Exam Format

- Different question formats
  - Essay
    - Text box in which answer can be typed
  - File Upload
    - Create your answer using: (1) a drawing application on your computer, (2) a drawing app from a web site, or (3) a piece of paper whose scan or photo is submitted
    - In any of these cases, you will need to ***upload your answer file***. You must ensure that you have uploaded any answer files **before the termination of the exam time**. **No files can be uploaded after time has expired.**
  - Multiple Choice
    - Choose the correct answer
  - Multiple Answers
    - Select all correct answers

Material on Midterm Exam

- Book readings
  - Aho, Lam, Sethi & Ullman, Chapters 1-6
  - Harbison & Steele
    - Familiarity with Chapters 1, 2, and 4 through 9 is required and assumed
- Material covered in class through – and including – next week's class meeting except for Intermediate Representation (if we get that far)
- Material covered in Problem Sets 1-3
- Discussions on Ed
- Material covered in section is beneficial, and is required

Topics for Midterm Exam (1 of 3)

- C Programming Language
- Flex/Lex
- Regex's
- Context-Free Grammars
- Ambiguity
- Left Recursion
- Left Factoring
- Bison/YACC
- The Grammar for Our Subset-C Language
- NFA, DFA
  - Creating an NFA from a Regex using the McNaughton-Yamada-Thompson Algorithm
  - Conversion of NFA to DFA using Subset Construction

Topics for Midterm Exam (2 of 3)

- Top-Down Parsing
- Recursive Descent Parsing
- Symbol Table Management
- Types in the C Programming Language
- Representing Types in Our Compiler
- Parse Tree, AST
- Name Spaces
- FIRST, FOLLOW, LL(1) Grammars, Predictive Parsing Table M, Table-Driven Predictive Parsing
- Number Representations

Topics for Midterm Exam (3 of 3)

- Type Checking
- C Standard Conversions
- Bottom-Up Parsing
- Shift-Reduce Parsing

Midterm Directions

- Read each question very carefully
  - The question should include all necessary information
- The point values in the exam are equal to the number of minutes that *a student who would score highly* might take to complete the question
- The exam in lengthy – don't be concerned if you don't complete everything
- **Initially, spend approximately the number of minutes on each question that the question is worth, then return to the question later to complete them as available time permits**

Reviewing Upcoming Dates

- PS3 Due: Sunday, March 9th, 2025 at midnight ET
- We have just one more class meeting (on Tuesday, March 11th, 2025) before the Midterm Exam due to Spring Break
- We have two more section meetings (on Tuesday, March 11th, 2025 and on Tuesday, March 25th, 2025) (but on March 25th, there will be no Midterm Exam material or answers to Midterm Exam questions)
- Midterm Exam
  - Earliest start time is 8:00 PM ET on Tuesday, March 25th, 2025
  - Must be started by 7:59 PM ET on Wednesday, March 26th, 2025
  - Must be completed within 3 hours of the time the student starts the exam
- PS4 Due: Sunday, March 30th, 2025 at midnight ET

New Material for this Week (1 of 3)

- Type checking slides
  - **Type Checking**
  - Review slides **#19 through #24**

New Material for this Week (2 of 3)

- Type checking
  - Discuss what happens with:

```
int i = -5;
unsigned int ui = 6;

if(i < ui)
  printf("i < ui\n");
else
 printf("i >= ui\n");
```

New Material for this Week (3 of 3)

- Continue with the C Programming Language slides
  - **Review of the C Programming Language**
    - Continue from the **Extent (or Lifetime or Storage Duration) (§4.2.7)** slide **#35** through **Static and Local Storage Duration** slide **#36**

Seventh Class Meeting on 3/11/2025

Seventh Class Meeting Agenda

- Questions and Comments
- Daylight Saving Time
- Harbison & Steele Readings
- Current Problem Set Status
  - PS2
  - PS3
  - PS4
- Spring Break
- Upcoming Midterm Exam including Proctorio
- Upcoming Dates
- Issues with Cross-Compiling
- Complete the **Review of the C Programming Language** slides
- Cover the **Shift-Reduce Parsing** slides
- Cover the **Numeric Encodings** slides

Questions and Comments

- Section
- Last week's class or any previous classes
- Problem Sets 0 through 4
- Midterm Exam
- Lexer
- Parser & AST construction
- AST, Type Tree, Symbol Table
- Our grammar
- Review of the C Programming Language
- Ed Discussion threads
- Readings
- Anything else

Daylight Saving Time

- **Daylight Saving Time began at 2 AM on this past Sunday, March 9th in Massachusetts**
- Clocks were set forward one hour
  - Local time is now _____ PM
- If you are a living in a different time zone, please adjust your schedule as needed and set your time zone in Canvas

Harbison & Steele (1 of 2)

- Even though no specific readings from **Harbison & Steele** are assigned in the syllabus, everyone should be reading and referring to H&S on a regular basis
  - Harbison & Steele is the only reliable source for specifications on type checking for Problem Set 4

- Familiarity with Chapters 1, 2, and 4 through 9 is required and assumed

Harbison & Steele (2 of 2)

- The grammar is not sufficient to determine if a program is valid
- There are many examples where the grammar accepts invalid programs
- Each section of H&S contains constraints that must be checked to ensure that a valid program exists

Problem Set 2

- **Problem Set 2**
  - Was due at midnight ET on Sunday, February 23rd, 2025
- Problem Set 2 involves building the AST
- If you don’t yet have PS2 completed, please send e-mail to all course staff and let us know what’s up
  - We can help resolve your issues

Problem Set 3

- **Problem Set 3**
  - Was due at midnight ET on this past Sunday, March 9th, 2025

- Problem Set 3 involves building symbol tables & type data structures
  - Symbol tables
    - Other names
      - Identifier resolution to entry in correct symbol table
      - Replace identifier in AST with pointer to entry in correct symbol table
    - Statement labels
    - String table
  - Type tree
    - Each other names symbol table entry must have an appropriate type tree attached to it
  - Create all needed pointers to allow traversal both inward and outward among symbol tables

Problem Set 4 (1 of 2)

- **Problem Set 4**
  - Will be due Sunday, March 30th, 2025 at midnight ET
  - This problem set is due on the Sunday after the midterm exam

- Problem Set 4 involves semantic analysis/type checking
  - Creation of type trees
  - Functions written to perform conversions (usual casting conversions, usual assignment conversions, usual unary conversions, usual binary conversions, etc.)
  - Implicit casts are made explicit by adding casts to the AST
  - Each AST node in an expression must be labelled with a correct type tree
  - Each AST node in an expression must also be labelled as either an lvalue/rvalue or an rvalue and whether that type is modifiable or not
  - Errors should be emitted for violations of semantic/type requirements

Problem Set 4 (2 of 2)

- Work first on creating type trees
- Then, starting with simple expressions and building to more complex expressions, add functions for conversions as needed and insert explicit casts and label all AST nodes in expressions with type trees, lvalue vs. rvalue, and modifiability
- As progressing, emit errors for invalid type checks
- Leave the most complex casts for last, *i.e.,* those for ++ and -- and compound assignment operators

Spring Break

- Spring Break is next week Monday, March 17th, 2025 through Sunday, March 23rd, 2025
- There will be **no class and no section meetings next week**
- However, **we will be holding office hours on Thursday next week, 7-8 PM ET**

Midterm Exam

- Our **Midterm Exam begins in two weeks** on March 25th, 2025 at 8:00 PM ET
  - There will be **no class meeting on March 25th**, but **section will still be held on March 25th**
    - No topics relevant to the midterm exam will be discussed in that section meeting

- Ask any questions relevant to the midterm exam in class today, in office hours, or on Ed before noon ET on the day of the exam

- May ***not*** be taken in class
  - Consistency for all students
  - Controlled access to slides

Midterm Exam

- Exam is **three hours in duration**
  - The exam is graded with a **maximum score of 120**
  - The problems are scored in points where **a point is weighted to be approximately one minute** of exam answer duration
  - But, the exam is time-consuming – we assume that you will take all three hours
- Earliest starting time is 8:00 PM ET on Tuesday, March 25th, 2025 – **two weeks from today**
- Latest starting time is 7:59 PM ET on Wednesday, March 26th, 2025
- The exam will be available through **Canvas** under the **Quizzes** tab
  - The exam will be administered under **Proctorio**
  - **Either Google Chrome, Microsoft Edge, Brave, or Opera must be used as the browser** for the exam – only these browsers are acceptable to Proctorio
  - Proctorio requires that a **Proctorio Extension** be installed into your browser
- No books may be used during the exam, but all slides from class will be available through the **Files** menu item in Canvas (a link to the **Files** page of Canvas is in the exam Instructions)
  - Nothing else can be used during the exam: no notes, no electronics other than the computer being used for the exam, no cell phones, no talking, no communication
  - But, nothing else is needed!

The Proctorio Setup Quiz

- *As soon as possible,* take the **Proctorio Readiness Quiz** that is available in **Canvas** under **Quizzes**
- Really **carefully read all of the instructions**
  - **The instructions are the same as for the real Midterm Exam**
  - **We are not able to deal with issues caused by not following the instructions!**
- Before taking the Midterm Exam, contact the course staff *as soon as possible* if you have any problems taking the **Proctorio Readiness Quiz**
- **Ensure that you are able to look at the class slides in Canvas under the Files tab while taking the Proctorio Setup Quiz**

Midterm Exam Format

- Different question formats
  - Essay
    - Text box in which answer can be typed
  - File Upload
    - Create your answer using: (1) a drawing application on your computer, (2) a drawing app from a web site, or (3) a piece of paper whose scan or photo is submitted
    - In any of these cases, you will need to ***upload your answer file***. You must ensure that you have uploaded any answer files **before the termination of the exam time**. **No files can be uploaded after time has expired.**
  - Multiple Choice
    - Choose the correct answer
  - Multiple Answers
    - Select all correct answers

Material on Midterm Exam

- Book readings
  - Aho, Lam, Sethi & Ullman, Chapters 1-6
  - Harbison & Steele
    - Familiarity with Chapters 1, 2, and 4 through 9 is required and assumed
- Material covered in class through – and including – next week's class meeting except for Intermediate Representation (if we get that far)
- Material covered in Problem Sets 1-3
- Discussions on Ed
- Material covered in section is beneficial, and is required

Topics for Midterm Exam (1 of 3)

- C Programming Language
- Flex/Lex
- Regex's
- Context-Free Grammars
- Ambiguity
- Left Recursion
- Left Factoring
- Bison/YACC
- The Grammar for Our Subset-C Language
- NFA, DFA
  - Creating an NFA from a Regex using the McNaughton-Yamada-Thompson Algorithm
  - Conversion of NFA to DFA using Subset Construction

Topics for Midterm Exam (2 of 3)

- Top-Down Parsing
- Recursive Descent Parsing
- Symbol Table Management
- Types in the C Programming Language
- Representing Types in Our Compiler
- Parse Tree, AST
- Name Spaces
- FIRST, FOLLOW, LL(1) Grammars, Predictive Parsing Table M, Table-Driven Predictive Parsing
- Number Representations

Topics for Midterm Exam (3 of 3)

- Type Checking
- C Standard Conversions
- Bottom-Up Parsing
- Shift-Reduce Parsing

Midterm Directions

- Read each question very carefully
  - The question should include all necessary information
- The point values in the exam are equal to the number of minutes that *a student who would score highly* might take to complete the question
- The exam in lengthy – don't be concerned if you don't complete everything
- **Initially, spend approximately the number of minutes on each question that the question is worth, then return to the question later to complete them as available time permits**

Reviewing Upcoming Dates

- PS3 Due: Was due on Sunday, March 9th, 2025 at midnight ET

- Today's class meeting is the last class before the Midterm Exam due to Spring Break
  - Next class meeting is on Tuesday, April 1, 2025
- We have one more section meeting (on Tuesday, March 25th, 2025) immediately before the Midterm Exam (but with no Midterm Exam material or answers to Midterm Exam questions)

- Midterm Exam
  - Earliest start time is 8:00 PM ET on Tuesday, March 25th, 2025
  - Must be started by 7:59 PM ET on Wednesday, March 26th, 2025
  - Must be completed within 3 hours of the time the student starts the exam

- PS4 Due: Sunday, March 30th, 2025 at midnight ET

- Our regular schedule resumes on Monday, March 31st, 2025

Issues with Cross-Compiling

- Our compiler runs on the host computer (our cscie95.dce.harvard.edu instance which is **X86_64**)
- Our compiler produces code for our target computer (**32-bit MIPS**)
- So, it's not correct to use information about the host computer to determine minimum and maximum values that are representable on our target computer
  - Therefore, don't use the <limits.h> header file for our target computer

- For example, on our **host computer**, a char has 1 byte, a short int has 2 bytes, an int has 4 bytes, and a **long int has 8 bytes**
- On our **target computer**, a char has 1 byte, a short int has 2 bytes, an int has 4 bytes, and a **long int has 4 bytes**

New Material for this Week (1 of 3)

- Complete the Review of the C Programming Language slides
  - **Review of the C Programming Language**
    - Continue from the **Storage Class Specifiers (§4.3)** slide **#37** through **the last** slide

New Material for this Week (2 of 3)

- Present the **Shift-Reduce Parsing** slides
  - Bottom-Up LR(*k*) Parsing
  - **Shift-Reduce Parsing**
  - Sufficient for all modern computer languages, including C
  - Our students don't need to be able to create an LR-parsing table – just be able to execute a parser using it

New Material for this Week (3 of 3)

- **Numeric Encodings** except for showing how to perform addition of floating-point values in IEEE 754 representation

Eighth Class Meeting on 4/1/2025

- Ninth class meeting if you count the midterm exam meeting

- Happy April Fools' Day!

Eighth Class Meeting Agenda

- Questions and Comments
- Sequential Nature of Problem Sets
- Current Problem Set Status
  - PS4
  - PS5
- Midterm Exam
- Errata from Harbison & Steele
- Students Not Yet Working on PS4
- Important Tables in Harbison & Steele
- Cover the **Intermediate Representation** slides
- Start to cover the **MIPS Instruction Set** slides
- [Cover **Numeric Encodings: How to perform addition of IEEE 754 floating-point values**]
- [Cover the **Character Encodings** slides]
- [System header files]

Questions and Comments

- Section
- Last week's class or any previous classes
- Problem Sets 0 through 4
- Midterm Exam
- Symbol Table
- Type Tree Creation
- Type Checking
- Labelling All AST *Expression* Nodes with Type Tree, Lvalue vs. Rvalue, and Modifiability
- Making Implicit Casts Now Be Explicit
- Our grammar
- Review of the C Programming Language
- Ed Discussion threads
- Readings
- Anything else

Sequential Nature of Problem Sets

- The Problem Sets are designed to not require modifications to an earlier problem set in order to implement the requirements of a later problem set
  - Problems need to be fixed
  - Required features that were bypassed need to be implemented
  - But, you should be able to take the output of PS*n* without modification and use it to implement PS*n+1*

Problem Set 4 (1 of 3)

- **Problem Set 4**
  - Was due this past Sunday, March 30th, 2025 at midnight Eastern Time

- Problem Set 4 involves semantic analysis/type checking
  - Creation of type trees
  - Functions written to perform conversions (usual casting conversions, usual assignment conversions, usual unary conversions, usual binary conversions, etc.)
  - Implicit casts are made explicit by adding casts to the AST
  - Each AST node in an expression must be labelled with a correct type tree
  - Each AST node in an expression must also be labelled as either an lvalue/rvalue or an rvalue and whether that type is modifiable or not
  - Errors should be emitted for violations of semantic/type requirements

Problem Set 4 (2 of 3)

- Work first on creating type trees
- Then, starting with simple expressions and building to more complex expressions, add functions for conversions as needed and insert explicit casts and label all AST nodes in expressions with type trees, lvalue vs. rvalue, and modifiability
- As progressing, emit errors for invalid type checks
- Leave the most complex casts for last, *i.e.,* those for ++ and -- and compound assignment operators

Problem Set 4 (3 of 3)

- Insertion of casts in PS4
  - Explicit casts are inserted into the AST in PS4 when the type of the result of an operator is not the same as the type when that result is used as an operand of another operator
  - It is possible that more than one cast would be inserted into the AST for a single implicit type conversion
    - For example, one cast might be added as a result of applying the **Usual Unary Conversions** and another added as a result of applying the **Usual Binary Conversions**
    - If this is the case, then multiple casts should be added to the AST

Problem Set 5 (1 of 2)

- Problem Set 5
  - Will be due on Sunday, April 13th, 2025 at midnight Eastern Time

- Present Problem Set 5

- Problem Set 5 involves generating intermediate code (IR)

Problem Set 5 (2 of 2)

- The most important design methodology is to remember to follow the approach where expressions are evaluated to an *lvalue* (when possible) and are evaluated to an *rvalue* only when necessary
  - Expressions are evaluated to an *rvalue* when:
    - The result of an operator is defined to never be an *lvalue*
    - The operand of an operator requires its operand to be an *rvalue*
- Tag each AST node in an expression with
  - The name of the temporary that holds the *lvalue* or *rvalue* of that subexpression
  - Whether that temporary is an *lvalue* or an *rvalue*

Midterm Exam

- We’re still reviewing and grading the midterm exams
- Next week, I hope to present midterm exam statistics and review the answers

Errata from Harbison & Steele

- Link to Harbison's errata is on the class web site
- There is also a link to our own errata from H&S

Students Not Yet Working on PS4

- Students who are not yet working on PS4 may choose the following path
  - If you have not yet started work on PS4, it is possible to delay work on PS4 or, perhaps, to never work on PS4
  - If you are certain that a type-correct program is given to your compiler, then PS4 would not change the AST in any way (except for inserting some casts in a few cases)
  - Therefore, in these cases, we suggest that you skip working on PS4 for now and implement PS5 *before* PS4
  - This gives you a shortcut to being able to run code

- If you haven't already done so, you will have to label every node in an expression in the AST with the type of that node
  - This would normally happen in PS4, but if PS4 is being skipped for now, it is still needed for PS5
  - This is required to know which IR instruction should be generated

Important Tables in Harbison & Steele

- **Table 7-1** on Page 204
  - Nonarray expressions that can be lvalues
  - As discussed in my errata, the sentence immediately before this table is *not* correct
- **Table 7-2** on Page 204
  - Operators requiring lvalue operands

Model for Intermediate Representation

- Two different storage areas in which data can be stored
  - Temporaries/registers
  - Memory

- Each location (byte) in memory has an address
- A temporary/register has a name, but not an address

- The load*Type* and store*Type* IR instructions are the only way to copy data between temporaries/registers and memory

New Material for this Week (1 of 2)

- Present the **Intermediate Representation** slides
  - Cover through the **unknown** slide **#unknown**

New Material for this Week (2 of 2)

- Present the **MIPS Instruction Set** slides
  - Cover through the **unknown** slide **#unknown**

New Material for this Week (x of y)

- **Numeric Encodings**: Show how to perform addition of floating-point values in IEEE 754 representation

New Material for this Week (x of y)

- **Character Encodings**

New Material for this Week (x of y)

- Look at some system header files on cscie95.dce.harvard.edu
  - They reside in directory /usr/include

  - /usr/include/stdint.h
  - /usr/include/limits.h