Success in CS-460: Programming Languages

Robert Bruce

What is Programming Languages all about?

- Defining and utilizing programming language grammar.
- Parsing through tokenization and later, recursive descent to form syntax trees.
- Lexical analysis for semantics (context) and syntax (conformity to grammatical rules).

On a more subtle level we will examine:

- Differences in programming language design and features.

How will we implement this?

You will write programs in C or C++ which implement:

1. Procedural-based Deterministic Finite State Machines.
2. Binary trees using Left-child, Right-sibling to store parse trees.
3. Linear linked lists to store symbol tables

The culminating final project in this class is developing an interpreter in a C-like programming language.

Is this really difficult?

- Yes, it *can* be difficult.
- I designed the in-class exercises to prepare you for success and overcome the difficulties. *Hint: a solid foundation built on tokenization will make lexical analysis and subsequent steps much easier when writing an interpreter.*
- Remember: you are not alone! You will be working on a team. Furthermore, I will be your coach and mentor.

Emphasis on short lectures

- Lectures will generally be short (approximately 30 minutes).
- Class-time primarily focused on in-class exercises in teams.
- I've designed the in-class exercises to help you understand and achieve course goals and objectives.

Success in CS-460: Programming Languages

- Attend classes and contribute to the in-class exercises.
- Manage your time well. Pace yourself!
- Utilize my weekly student hours (traditionally called “office hours”).
- Communicate! Communicate! Communicate!

Communicate! Communicate! Communicate!

- When issues arrive on a team and cannot be resolved, please let me know as-soon-as-possible.
- I have diplomatic ways to find common ground and get the team to work well together.
- I want your team to succeed!
- I have student hours (traditionally called “office hours”) to serve you! Student hours are a GREAT time talk. I really enjoy hearing form students.

Advice from former CS-460 students

- "To know the road ahead, ask those coming back." - Chinese proverb

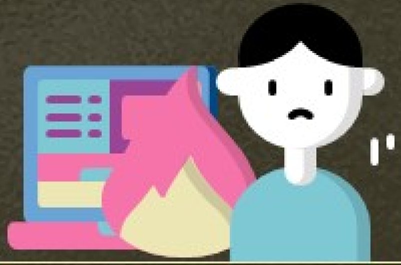
Challenges

Easiest

CST
AST
Symbol Table
Comments
Tokenization

Hardest

Descent Parser
Interpreter



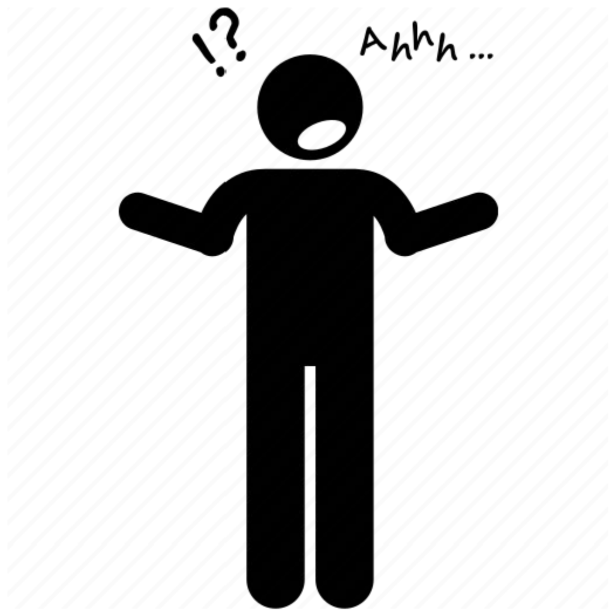
Advice for New Teams

- Start early
- Ask questions
- Don't take CS 470 at the same time
- Have more structure for tasks and deadlines within our team
- Split up the work where possible

Difficulties

- Time crunches
  - Didn't always leave enough time for bug fixing
  - Often leading to redundant and messy code
    - Leading to more time to fix
      - Leading to messier code
        - Making it take longer to fix
          - Leading to messier code
            - Leading to it taking longer
              - Leading to messier code

- Sharing the project over Github
  - Handful of nightmares with overwrites
  - Hard to work at the same time



Realizing the Importance of Time Management

- Starting projects early
  - Plan early on
  - Break apart tasks
  - Make a schedule to meet
- Less stress for everyone
  - More sleep
  - More time for other projects
- More time to debug and write better code
  - Rushing to finish

Easiest/Hardest parts of the project

- Easier
  - Ignoring comments
  - Tokenization
- Harder
  - Abstract syntax tree
  - Putting all the pieces together for the interpreter

Different Approaches We Would Have Now:

- Starting a Trello board to assign tasks
- Starting the projects earlier
- Having more full-team meetings to work through issues
- Better communication

Advice To A New Team

- Start the projects early
- Communicate with one another
- Don’t be afraid to ask for help!
- Utilize your team members strengths

Recommendations

- Set up and work from a repository from the beginning
- Keep it simple
- Establish clear cut team member roles early on
- Avoid the sunken cost fallacy
  - Sam's symbol table entry classes to a struct
  - 3 -5 files completely retrofitted
- Make sure everyone is on the same page
- Brainstorm together with the rest of the group
- "Communicate, communicate, communicate" – Robert Bruce

Challenges

Using Replit - coding repository for teams

- *Delegating tasks to team members*
- COMMUNICATION
- *Organization*
- *Bringing it all together*

- *Balancing all other classwork*

Improvements on Development

Manage time more efficiently

Have clear tasks assigned

Use trello or a task manager

What were the easiest and most difficult aspects to writing an interpreter?

Easiest:

- Removing comments
- Building off of the DFA with other functions

Difficult:

- Concrete syntax tree
- Making our execute main function

If you started this project all over again, how would your approach differ or would it remain the same? Explain.

- Start planning things sooner
- Make more functions when the same thing is written multiple times
- Spend time to refactor the code
- Document the code so the next person can easily understand it and to make it easier to understand what old code does
- Before starting the next assignment, we should ensure that the functionality of our previous assignment works with the new inputs.

What experience and advice would you share to a new team about to embark on writing an interpreter for CS-460?

Take into account what the interpreter should or shouldn't allow (ex. Variables with numbers in the name)

Take into account edge cases that aren't in the graded test cases.

Understand the DFA you need to make before doing anything else, as that will form the basis for the whole project

Plan out what you are going to code before you start coding

Our advice for future CS 460 students:

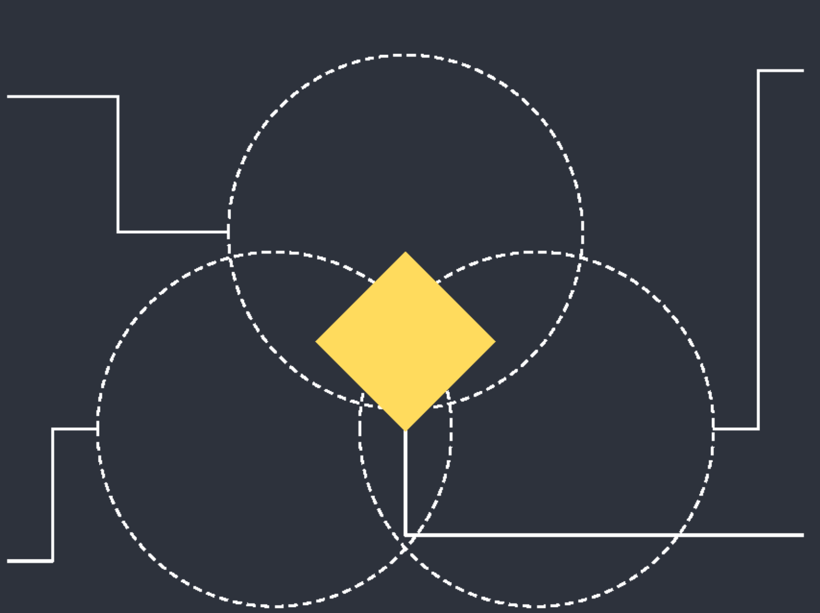
ORGANIZATION!!!

Difficult vs. Easy Aspects

- It's easier to account for what the computer should do if the given code is 100% correct

- BUT there's so many different ways for the code to be incorrect

Our takeaways from this class



Understanding Interpreters

Coding collaboratively

Importance of code organization

Teamwork

The following is an exact quote from the speaker notes of one CS-460 team:

Advice for future CS460 students

This is why our BIGGEST advice to future CS 460 students would be to make sure you organize your code from the start. And I think that if we were starting this project from the beginning again we might benefit from spending a little time learning a collaborative coding tool like github. While our way of doing it on our own computers worked well enough for us it would probably save us from a lot of zipping, uploading, and re-downloading files and not procrastinating, but we'll work on learning how to do that later (hahaha get it??)

Interpreter: Difficult and Easy

It was generally easy getting started on each segment of the Interpreter as we had a good foundation of what we wanted to accomplish and how it was going to be done, however working on and fine tuning the details of each section was difficult and thus took the longest amount of time.

Establishing proper version control also initially proved difficult however we setup a Github repository and worked off of that, and also maintained communication about what we were specifically working on in our Discord server, which helped significantly.

If We Restarted (git reset --hard)

- Some rounds of testing required additional modifications to previous aspects of the interpreter. Thus always looking ahead and keeping the bigger picture in mind more often would have helped during implementation.
- Group communication method and code sharing setup, using a Discord server and Github, would be kept the same as it worked out well to stay updated on the current iteration of the code.
- Directly pre planning scheduling for each segment could have helped limit any time crunching leading up to due dates.

Project Significance in Industry

- Working on an interpreter in a group project showcases your collaboration and communication skills, essential for software engineering roles that often involve teamwork and coordination with other developers.
- Discussing your experience with coding an interpreter in a group project allows you to showcase your learning process, adaptability to new technologies, and the ability to quickly grasp and contribute to unfamiliar codebases.
- Demonstrating we can encounter challenges and find innovative solutions while working on the project.

Summary

- I’m here to help you and guide you.
- We work as a team.
- Your commitment to this class is paramount to your success.
- Make me proud of your work!

Thank You