Time/Project management

Throughout this coursework, organization and regular progress were essential to maintain a good pace with the compiler's coursework. Although our start seemed very optimistic, we encountered some heavy challenges mainly along the third week resulting in a slow down in our progress and original planned deadlines.

In general, Khayle's main focus was centered around Code Generation and Yuna's around the Lexer/Parser (ASTs). When working on implementations and debugging features, both would complement each other when looking as the problem as each was fairly quick at identifying the error in the part they have been working on.

The github commits are not very representative of our progress. After experiencing various random automatic overwriting issues with github desktop resulting in loss of days of work, we exchanged a lot of our progress via other platforms.

Timetable:

Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26: DEADLINE		

Days worked on the coursework for 3 to 4 hours

Days worked on the coursework for 7 hours +

Big issue

TIMELINE

1st of March: Start of the Project, looking at the derivable and understanding what is expected of us

- Thinking on how to organize our time
- Getting informed on how to get a lexer running
- Ideal plan for the four weeks:

Week 1	Get parser/lexer fully working for all basics & start implementing Codegen
Week 2	Memory management & all basic features fully working[lexer,parser,codegen] + some intermediates
Week 3	Automate testing to see how many tests we pass, all intermediates and already working on advanced features
Week 4	Finish up all advanced features and review that everything works fine

-WEEK 1-

GOALS for week 1:

(1) Get a parser/lexer working by the end of the week [both Khayle & Yuna]

3rd of March: Implemented first version of the Lexer

3rd - 7th of March: Trouble understanding where to start with parser and how it links with the lexer. Required more research.
[commit 5c533d71664fec31cdc0715895501268f3fd2500]

7th of March: Thinking about Code generation [Codegen] and good understanding of where to head with the parser, experiencing some bugs with github

REVIEW of week 1:

(1) Goal achieved, good understanding of parser/lexer for both of us.

   a. Understanding format of AST : what they are, how they should be implemented

(2) Some research done for Code Gen

GOALS for next week2:

(1) Get Code Generation working for all the Basics [Khayle]

(2) Implement AST for basics & AST/Parser for intermediate functions [Yuna]

-WEEK 2 -

9th of March: First Version of the parser with all basic implementation but missing all ASTs
Git
commit: 2a88b4fd4c1c7bc26f7dc468ce4db52bbb04128f

11th of March: ASTs for all basic functions and random test cases parsing properly

13th of March: Implementation of Code Generation + memory management

14th of march: Due to merging issue we lost all progress of implementation of memory management and Codegen implemented

REVIEW of week 2:

(1) Reviewed the whole parser and decided to conform to ANSI C lexer/parser
(2) Got all ASTs for basic features => Parser working properly
(3) Big loss on memory management and Codegen => made us behind on ideal weekly progress

GOALS for next week3:

(1) Implement Code Generation for Intermediates
(2) Get automated testing working
(3) Parse the advanced features

-WEEK 3-

15th - 16th of March: Reworking on implementation of Code Generation +memory management + Implementing ASTs for intermediate features

18th of March: All basics fully work : lexer+parser+code gen => move on to intermediate

19th - 20th of March : Khayle works on Codegen and Yuna on automated testing.

REVIEW of week 3:

(1) Got memory management working
(2) All basic features are fully working and multiple intermediate features working as well
(3) Automated testing working too : pass 47 test cases

GOALS for next week4:

(1) Realistically will not get to implement all advanced features but focus on passing all test cases for intermediates and basics and implement as much advanced features as time allows
(2) Aim for a pass of 70 test cases for submission

-WEEK 4-

23rd -26th of March: Implementing as many advanced features as time allows

REVIEW of week 4:

(1) All basic and intermediate features are fully working and some advance features are implement

(2) Not implementing structs, pointers but all other features should be working

Overall Feeling on the Coursework, what could have been done better:

- Should have conformed to the ANSI C format earlier so more time would have been spent on memory management and Codegen.
- Underestimated the amount of work Codegen actually was.
- Wish there were more explanatory specification so we wouldn't have had a slow start figuring out how parser and lexer work together and have a better idea on the hierarchy of priorities we should have set.