MINOR PROJECT 1

SYNOPSIS

ON

Compiler Construction

Submitted By

Name	Sap ID	Roll No.	Branch
Krishan Rupani	500067535	R171218057	Btech-CSE DevOps
Neha Singh	500069028	R171218067	Btech-CSE DevOps
Payal Singla	500069630	R171218118	Btech-CSE DevOps

Under the guidance of

Dr. Hitesh Kumar Sharma
Assistance Professor
Department of Cybernetics,
School of Computer Science

UPES
UNIVERSITY WITH A PURPOSE

Department of Cybernetics,
School of Computer Science
UNIVERSITY OF PETROLEUM AND ENERGY STUDIES

School of Computer Science
University of Petroleum & Energy Studies, Dehradun

Project Proposal Approval Form (2019-2020)

Minor | I |

Project Title: Compiler Construction

Abstract: (Half to 2/3 of a page) (Mention Keyword at the Bottom)

Compiler is a familiar word to anyone who has ever written any computer program. Basically a compiler is a computer program that converts programming languages (high level language) into Machine understandable language (low level language). So a programmer writes the code in the compiler and compiler produces output according to the code or it produces error if the code is incorrect in anyway. We have used many compilers like Codeblocks, Dev C++, Jetbrains IDEs but all these compilers are software, you have to download them separately and if your computer doesn't have great computational power than it gets really overwhelming and a tedious task to manage so many compilers. What we are building is a web based compiler where you don't have to download compliers separately, just compile on Web Application. In this project we are developing compiler for subset of C which includes basic program structure, stdin and stdout and Control Statement. More will be developed in future projects.

Keywords: Compiler, Programming, IDE, Codeblocks, Dev C++, Jetbrains

Table of Content

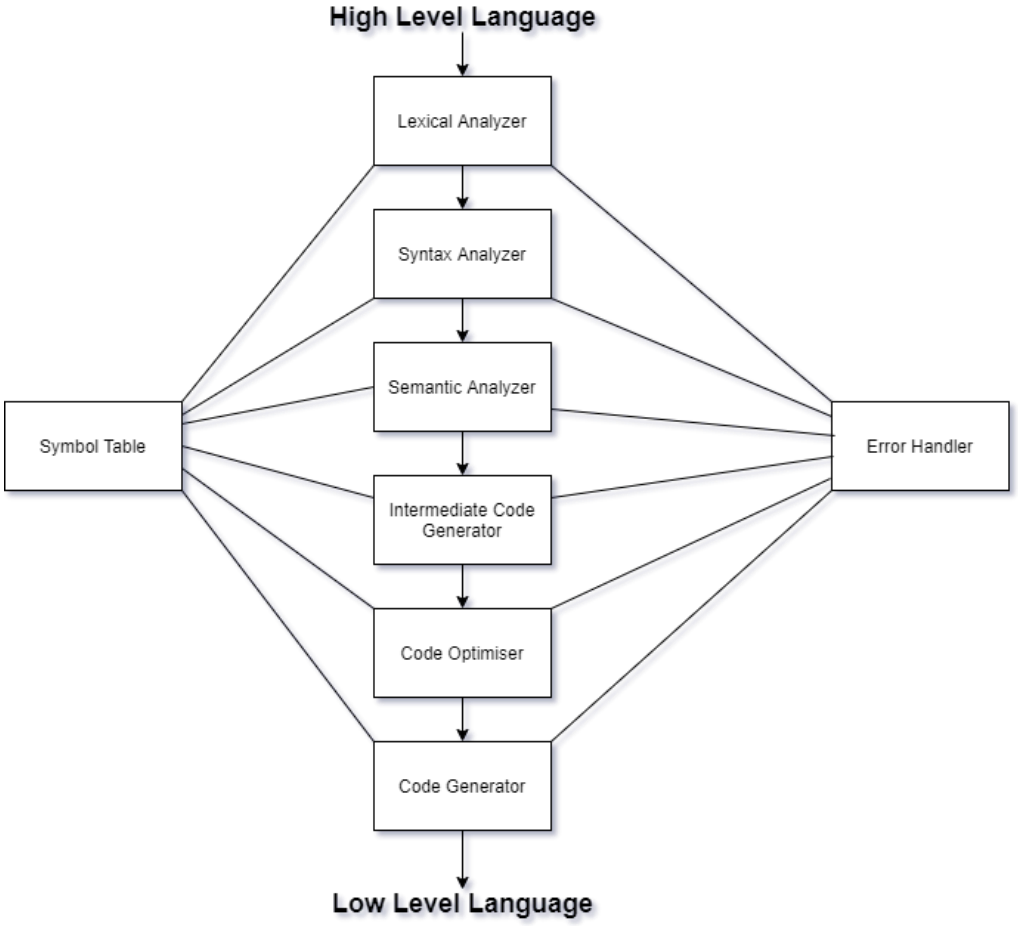
Introduction:

Inspiration to this project came from my own suffering, I have a PC with good computational power but because I do a lot stuff managing so many different compilers for every language made my life really tough. This problem is faced by many students, one more problem I have noticed is while learning programming its preferred that we practice as we are learning so for that purpose our compiler can be used by ed-tech startups who are providing online learning in programming and their students can practice and learn simultaneously. Main focus in this project will be on lexical analysis and syntax analysis, they both work closely in order to do analysis of the code.

How compilers work?

Compiler is a special program which converts a particular programming language to machine understandable code or language. Basically it converts high level language to low level language. If the source language (high level language) is incorrect in terms of rules for that particular language, then compiler also returns errors or warning.
Compilation process: is a sequence of various phases. Each phase takes input from its previous phase and has its own representation of source program and gives it's output to the next phase of the compiler.

Compiler Construction is divided in following stages:



Lexical Analysis: Works as a text scanner, scans as stream of characters and converts it into meaningful lexemes represented as tokens.
Syntax Analysis: Takes token as input from lexical analysis phase and generates a parse tree. Parser checks if the expression made by tokens is syntactically correct. BNF and concept of trees will be used here.
Semantic Analysis: Checks if parse tree constructed follows the rules of languages, it keeps track of identifiers, their types and expressions. It produces an annotated syntax tree as output.
Intermediate Code generator: Represents source code for some abstract machine, it is between high level and machine language. We will use x86 assembly language for this.
Code Optimization: Removes unnecessary line of code
Code Generator: maps optimized code to target machine language and translates IC into a sequence of relocatable machine code.
Symbol Table: Maintained throughout all phases of compiler. Used for quicker search of identifiers.
Data Structures (Trees particularly), File handling and Automata will be used in this project.

Market Analysis

There are a lot of companies producing compilers of desktop, Jetbrains a software company from Czech who produces some of the best IDEs we know for desktop like Intellij Idea for Java, Pycharm for python etc. has a annual revenue of $270M. Like this we have CodeBlocks, Dev C++, Netbeans and many other compilers available for different programming languages but most of them are for desktop and require some sort of specification to run properly. There are online compilers as well like online GDB which is a fascinating product and our inspiration as well but one thing that pinches me about this web application is its UI & UX. We will be developing with a better UI & UX.
Our compiler will be launched as an online compiler of users but there is one more way we will use our compiler. Many edtech startups want a compiler imbedded in their websites we will we helping them as well. Infact, the compiler first will be used in our own edtech startup.

Problem Statement:

Managing different compilers for different language can be overwhelming for programmer and its computer as well. There are many compilers online but most of these compliers are language specific for a programmer it would be difficult to open different compilers rather than just choosing the language they want to compile in, like we intent to build ultimately. Also in terms business, if a web application is providing a single language compiler on one domain then user behaviors tells us that users tend to forget the company after they have used the product.

Objectives:

To create a web based compiler for subset of C language.

Literature Review:

There are lots of compilers already in market like:
- Codeblocks
- Dev C++
- Jetbrains IDE
- GDB online Compiler

Some other not so famous compiler in market as projects are:
- SubC by Nils M Holm
- Swieros C Compiler by Robert Swierczek
- fbcc
- tcc
- Small C

There are many article online on Compiler design from where we are gaining our knowledge
- https://www.codeproject.com/Articles/30353/Designing-a-Compiler
This is the first piece of content we read on compiler, it taught us about phases of compiler in depth.

- https://www.sigbus.info/how-i-wrote-a-self-hosting-c-compiler-in-40-days#:~:text=Here%20are%20some%20examples%20of,It%20also%20supports%20function%20calls.
This is a self-hosting compiler that supports C11 language feature written by **Rui Ueyama** who is a software engineering at Google, he wrote the compiler in 40 days' sprint which is very inspiring. Here we learnt a lot about programming aspect of Compiler Design, study still in progress.

- https://norasandler.com/2017/11/29/Write-a-Compiler.html
This Compiler is by Nora Sandler, we got to know about practical side of compiler design, also viewed some of her code.

- https://www.thecrazyprogrammer.com/2017/02/lexical-analyzer-in-c.html
This article is exclusively on Lexical analysis, we got to know a lot about this phase through this article.

While we will be building our own compiler these resource are from where we are strengthening our concepts.

Methodology:

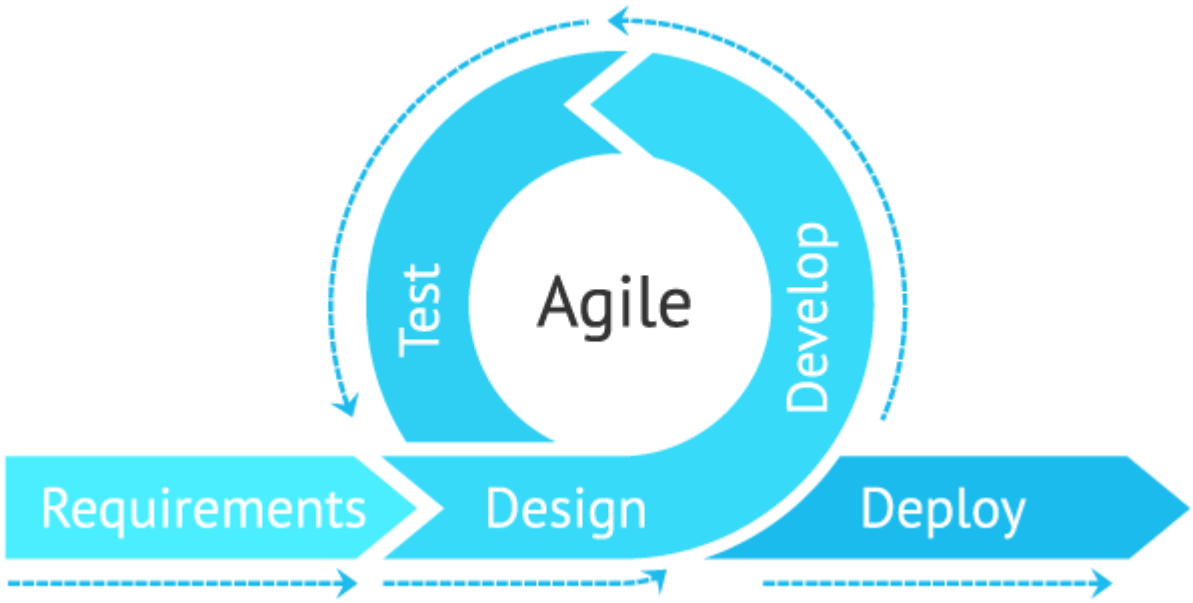
We will make our project using agile methodologies following agile values and principals.
Agile project management is a methodology that is commonly used to deliver complex projects due to its adaptiveness. It emphasizes collaboration, flexibility, continuous improvement, and high quality results.
The Agile Manifesto is comprised of four foundational values and 12 supporting principles which lead the Agile approach to software development.

Agile Values:

1. Individuals and Interactions Over Processes and Tools
2. Working Software Over Comprehensive Documentation
3. Customer Collaboration Over Contract Negotiation
4. Responding to Change Over Following a Plan

Agile Principals:

1. Customer satisfaction through early and continuous software delivery
2. Accommodate changing requirements throughout the development process
3. Frequent delivery of working software
4. Collaboration between the business stakeholders and developers throughout the project
5. Support, trust, and motivate the people involved
6. Enable face-to-face interactions
7. Working software is the primary measure of progress
8. Agile processes to support a consistent development pace
9. Attention to technical detail and design enhances agility
10. Simplicity
11. Self-organizing teams encourage great architectures, requirements, and designs
12. Regular reflections on how to become more effective



System Requirements: (Software/Hardware)

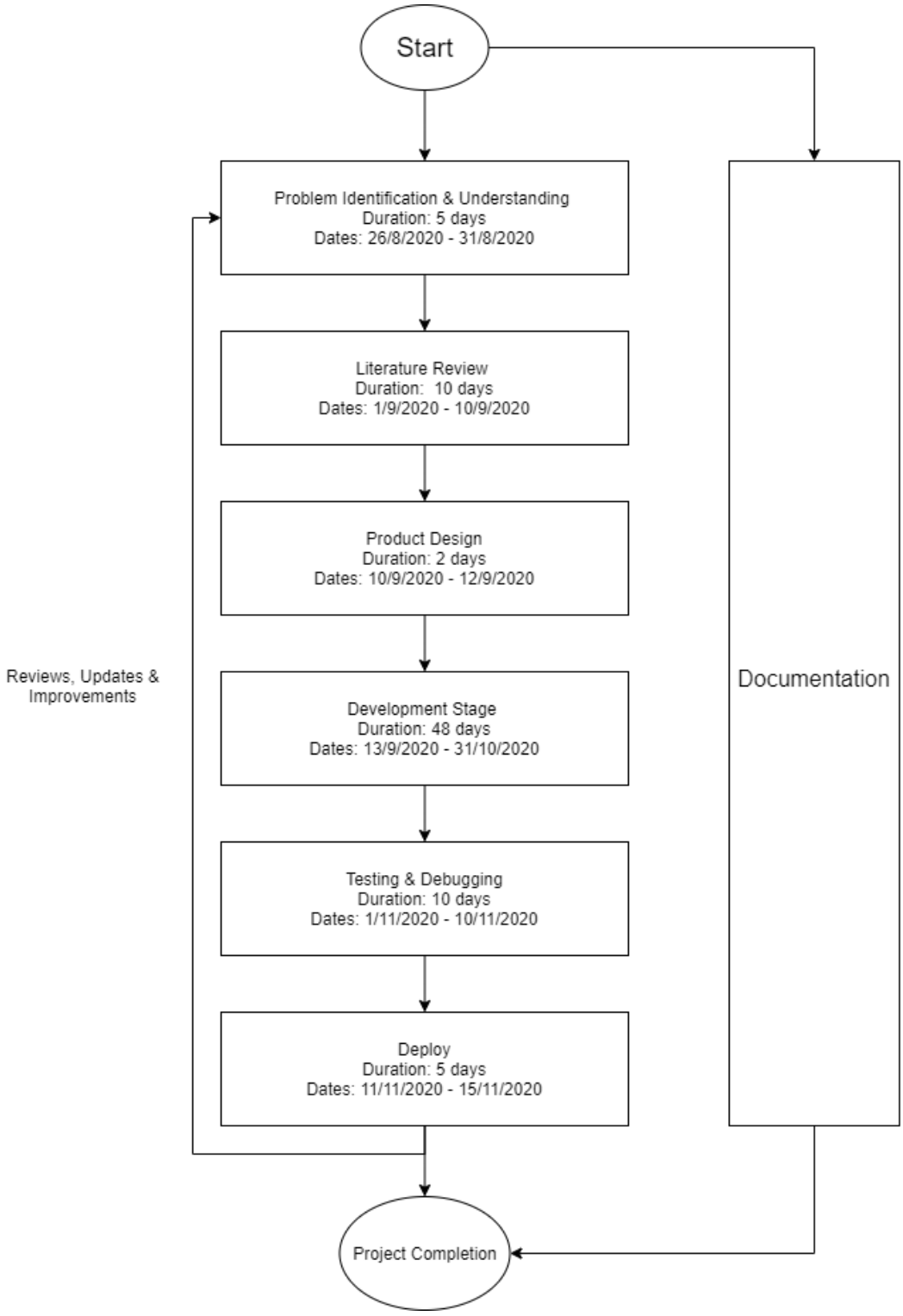
Software:

=> C programming language
=> HTML & CSS
=> Windows (7 and above)/ Linux Operating System
=> x86 assembly language
=> Codeblocks compiler

Hardware:

=> 2 GB RAM
=> 100 GB externally hard disk
=> 1 gigahertz (GHz) or faster 32-bit (x86) or 64-bit (x64) processor

Schedule: (PERT Chart)



References:

1. Compilers: Principles, Techniques and Tools by Alfred V. Aho, Ravi Sethi, Jeffery D. Ullman
2. Compiler Design in C By Allen I. Holub
3. Linux manual pages for LEX and Yacc

* Whole Documents should not be more than 7 pages excluding Front Page
* The Front should contain Project Name, Partial Submission for Minor, Students name, Enrollment No, SAP Id no, Mentor Name

Approved By

Dr. Hitesh Kumar Sharma

Assistance Professor
Department of Cybernetics,
School of Computer Science

Project Guide

Dr. Monit Kapoor

Head of Department
Department of Cybernetics,
School of Computer Science

Head of Department