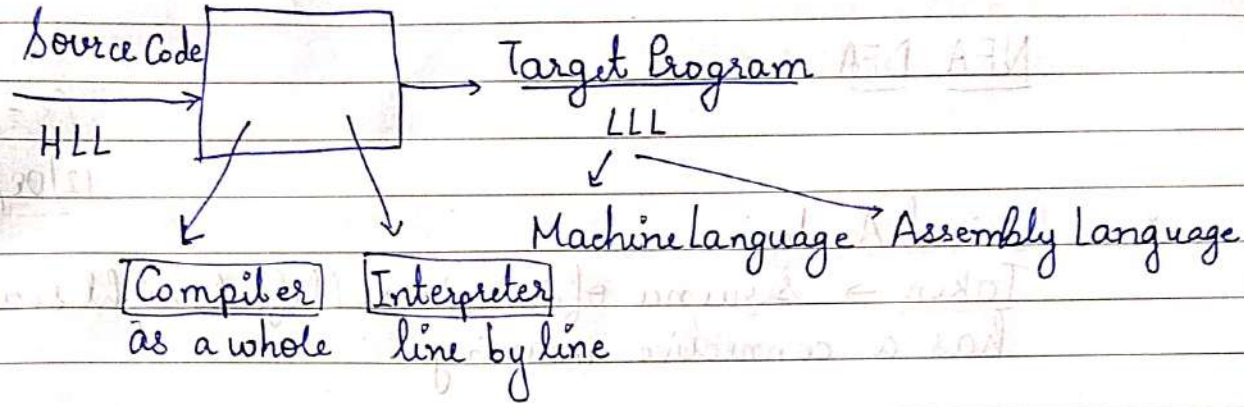
Date __/__/____

Saathi
11/07/2022

Compiler Design

Interpreter



Advantage: Debugging is easier in Interpreter.
Disadvantage: Time consuming → Interpreter.

Assembly Language: converts assembly language to machine level language. (Low level → Low level)

We will learn compiling here (not execution).

position = initial + rate x 60 → HLL

↓ → Source program

Phase I — Lexical Analyzer
output | Token → keywords, operator, identifier (out)

Phase II — Syntax " → Parser
Parse Tree (out)

Phase III — Semantic "

Target Program

Intermediate Code Generator → Three address code

Code Optimization

Code Generator

Saathi

Compiler Design
Aho, Ullman, Sethi

Compiler
Sridha

NFA DFA

(Q, Σ, δ, F)

12/08/2022

* Lexical Analysis

Token → Sequence of characters (left to right scan) that has a connective meaning.

Inputs: Total = num + num1 * 50

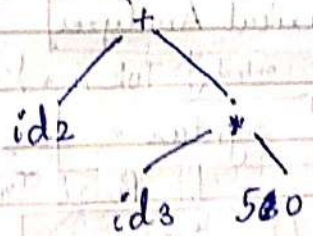
id1 id2 op1 id3 op2 50

Outputs: id1 = id2 + id3 * 50 (presented in tokens)
(Stored in symbol table)

* Syntax Analysis

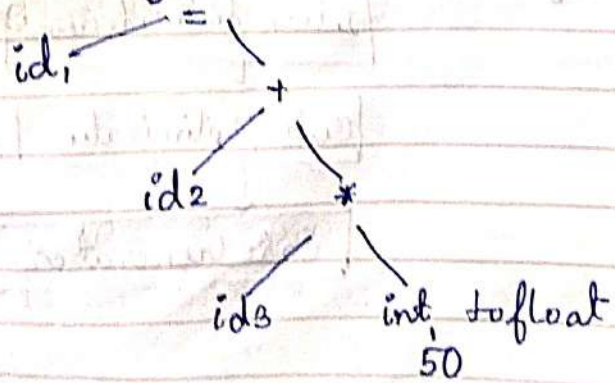
Input:- $id_1 =$

Parse tree



* Semantic Analysis

Input: Modified (Type change)



Saathi

Date ___/___/_____

* Intermediate Code Generator

Using temporary variable store $t_1 = (\text{int to float})\ 50.$

Another temporary variable store, $t_2 = id3 * t_1$

$t_3 = id_2 + t_2$

$id_1 = t_3$

~~MUL~~ ~~id3~~, t_1 ~~t_2~~

ADD ~~id_2~~, t_2

```
MUL t2, id3, t1
STORE   t2
ADD  t3 id2, t2
STORE   t3
MOV  id1, t3
```

* Code optimization

$t3 = id3 * 50.0$

$id_1 = id_2 + t_3$

Low LL → Binary, Assembly.

```
MOVF R2, #50
MULF R2, id3
MOVF R1, R2
ADDF R1, id2
```

Q) Position = Initial + rate * 60

1. Lexical Analysis

Input: Position = Initial + rate * 60

↓ ↓ ↓ ↓ ↓ ↓

id1 op1 id2 op2 id3 60

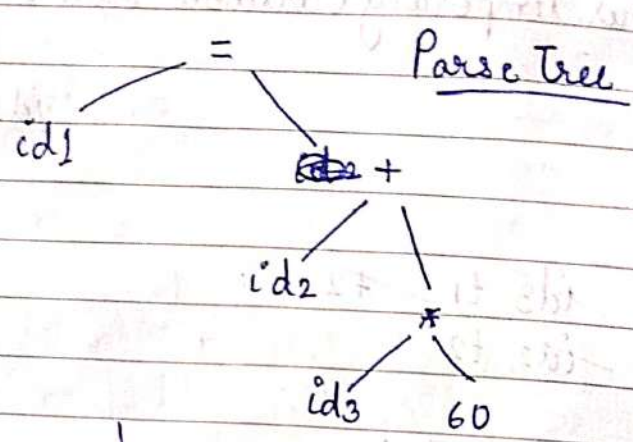
Saathi

Output: id1 = id2 + id3 * 60 (sent to Syntax Analyzer)

↓

* Syntax Analyzer

Input:

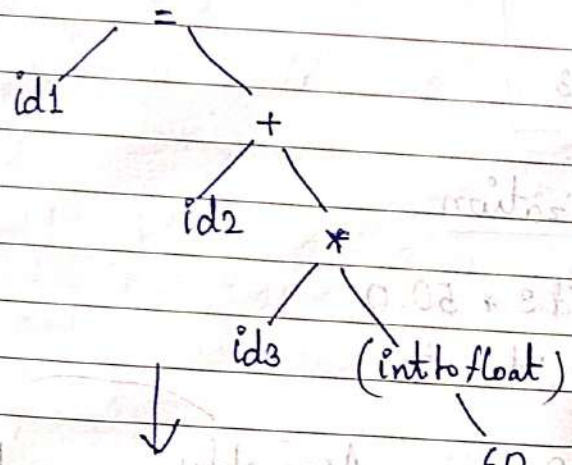


Parse Tree

↓

* Semantic Analyzer

Input:



↓

* Intermediate Code Generator

```
t1 = (int to float) 60
t2 = id3 * t1
t3 = id2 + t2
id1 = id2
```

↓

* Code Optimization

```
t3 = id3 * 60.0
id1 = id2 + t3
```

Date ___ / ___ / _____

Low LL

```
MOVF  R1  #60
MULF  R2, id3
MOVF  R1, R2
ADDF  R1, id2
```