# Finculate-not-Speculate


Group 6

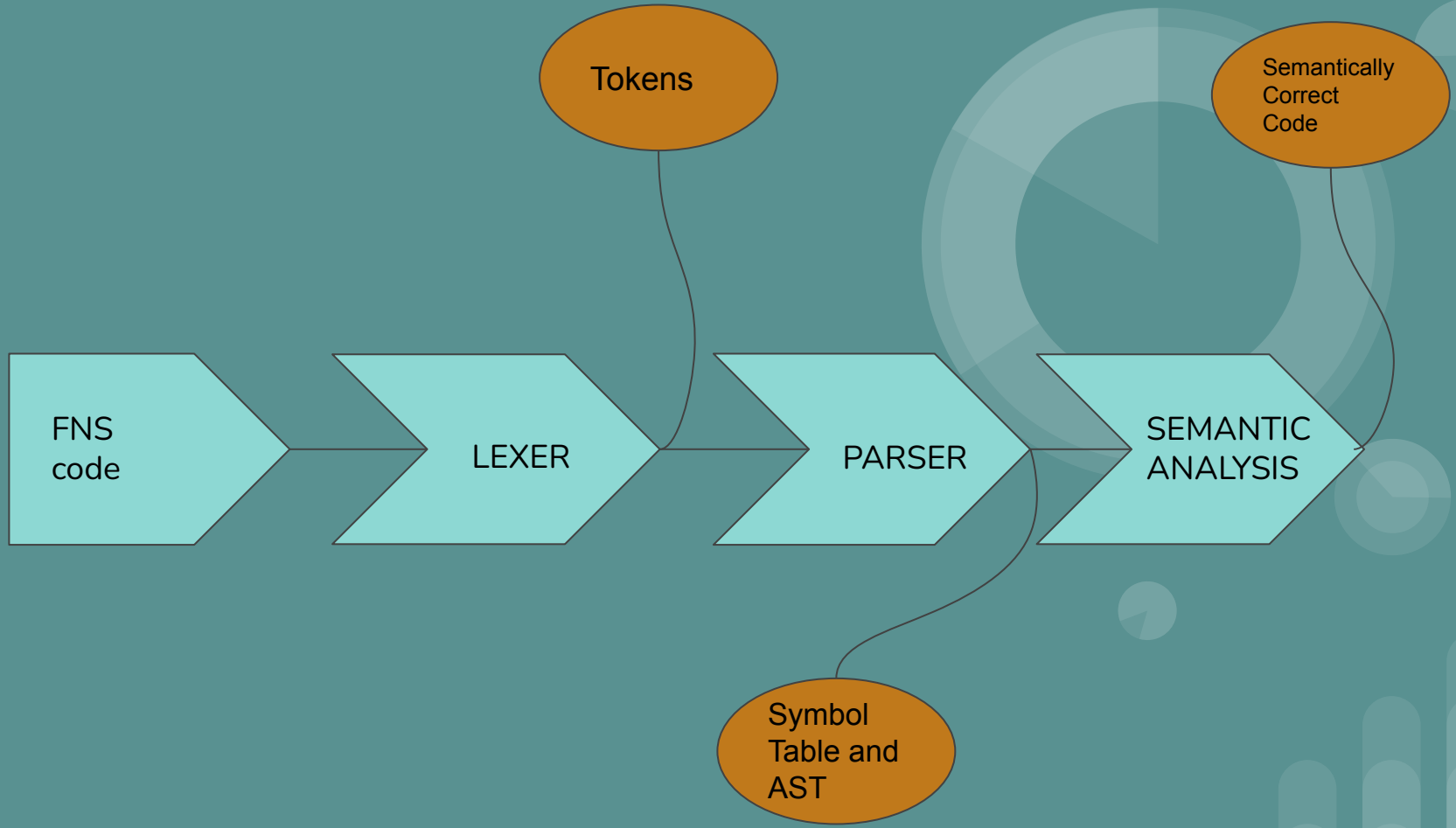
Diya Goyal
Dontha Aarthi
Nyalapogula Manaswini
Sushma
Namita Kumari
Kharadi Monika
Md Adil Salfi

# Introduction

**IDEA :** FNS - A procedural programming language to ease financial calculations. It has inbuilt functions to calculate simple interest, compound interest, SIP and step-up SIP maturity returns.

The goal is to do complicated calculations in the backend so that the user has an exact value for the amount of money involved in the picture.

All calculations involving bank loans, returns and investments have the formula of compound interest at its core. Doing mental calculations is non-intuitive because compound interest has an exponential graph, hence making estimates is very difficult. Our language can be a useful tool to automate calculations for the end-user so that they can make better decisions that affect their financial health.


Tokens
Semantically Correct Code
FNS code
LEXER
PARSER
SEMANTIC ANALYSIS
Symbol Table and AST

PHASE 1:

# LEXICAL ANALYSIS

# LEXICAL ANALYSIS

- A single element of the programming language is considered as a token.
- It consists of contents, identifiers, keywords, operators and separators.
- Lexical analysis is used to tokenize the source code.It returns all the valid tokens and checks for the invalid ones.
- It also removes the comments and white spaces.
- LEXER is used for lexical analysis. We have used flex for the same purpose.

# LEXER

- “Input.fns” is the source code taken as an input stream for the lexer.
- The generated tokens are then pushed to the parser to make the ast, symbol table and to check with the grammar rules.
- The tokens are defined in the lexer.l file. Using flex the input file is tokenised and then printed.
- Incase of invalid tokens it shows an error message “Invalid character : Line no ”.

PHASE 2:

# SYNTAX ANALYSIS

# SYNTAX ANALYSIS

- PARSER is used for checking/verifying the grammar of our source language.
- Parsing functions take a lexical analyze and a lexer buffer as arguments, which is a function from lexer buffers to tokens and return the semantic attribute of the corresponding entry point.
- Our parser is built using BISON.
- GRAMMAR: A program which is generally represented as a sequence of ASCII characters is changed into a syntax tree using grammar rules, which describes the ordering of symbols in the language.

- At the start of the code we arrange the tokens according to the preferences, %left and %right are used for associative precedences.
- It usually checks all data provided to ensure it is sufficient to build a data structure in the form of a parse tree.
- It checks data types,variable types to see whether declared properly or not.
- It checks whether the statement written in example are declared properly or not.
- The grammar is define in the parsar.y file. Using bison, parser.tab.c and parser.tab.h .

PHASE 3:

# SEMANTIC ANALYSIS

# SEMANTIC ANALYSIS

- Semantic Analysis is the process of drawing meaning from a text i.e, it interprets sentences so that computers can understand them and helps us in maintaining the semantic correctness of the program.
- Functions of Semantic Analysis are Type Checking, Label Checking and Flow Control Check
- It occurs during compile time(static semantics) and run time(dynamic semantics).
- Syntax tree of the parser phase and the symbol table are used to check the consistency of the program in accordance with the language definition.
- It stores the gathered information in either syntax tree or symbol table which would be used by compiler during the intermediate-code generation.
- Errors recognized by semantic analyzer are Type mismatch, Undeclared Variables and Reserved identifier misuse.

- The AST (Abstract Syntax Tree) is a hierarchical tree that represents the source code and the nodes of the tree represent construct of the code.
- The compiler generates symbol table and AST during semantic analysis.
- Ast explanation:
  - information preserved in ast are like variable type which contains int,float and binop which consist of arithmetic operations like add,sub,mult.
  - order and definition of executables statements present in ast with name type stmt,type decl.
- The semantic checks that we have implemented include checking the data types, redeclarations, undeclared variables and functions.

The src folder contains the following files :

lexer.l
parser.y
symbolTable.h
ast.h
semantics.h
functions.h
functions.c
Makefile

- “make all” command is used to make all the files.

# THANK YOU