**COMPILER CONSTRUCTION**

**LAB TERMINAL**

**Student Name:** Hafza Nawaz

**Registration no:** FA21-BCS-018

**Teacher Name:** Sir Bilal Ahmad

**Explain 2 core Function of mini compiler**

**Parser.parse\_statement() (from Parser.py):**

Purpose: This function is responsible for parsing a single statement in the source code based on the current token.

**Functionality:**

It checks the current token. If the token is keyword like int which starts a declaration, then it will call parse\_declaration(). Similarly if the token is if keyword it will call parse\_if\_statement() etc.

It creates AST nodes according to the given statement.

The parsed statement will be an AST node object.

It handles different types of statements, including variable declarations, assignments, conditional (if), loops (while), function calls, print statements.

Importance: This is a key part of the parser and provides high-level control flow to create AST Nodes.

**CodeGenerator.visit(node) (from CodeGenerator.py):**

Purpose: This function generates the assembly code for different types of AST nodes (visitor pattern implementation).

**Functionality:**

It uses a visitor pattern to traverse AST nodes.

Based on the type of the AST node it is visiting, it generates the corresponding assembly instructions.

For BinaryExpr, it loads the operands into registers, performs the given operations and stores the result into the register.

For Assignment, it loads the value into register and store it in a memory location based on the variable information.