**COMPILE CONSTRUCTION LAB FINAL**

**SUBMITTED BY: DUA NADEEM**

**SUBMITTED TO: SIR. BILAL HAIDER**

**REGISTRATION N0: FA21-BCS-016**

**Question#4**

**Explain the function that is performing semantic analysis in your mini compiler.**

**Semantic Analysis - What the Semantic.cs function does**

1. **Main function Validate(AstNode ast) :** This function receives Abstract Syntax Tree generated previously in Parser class (AST is received). It validates logical code based language defined rule by methods implementation and data validations to every type present from tree to guarantees correctness.
2. **Recursive Processing (Walker based)** Uses "visit or walk" mechanisms or a similar type that allows to transform data structure recursively , based class hierarchy validations, performing operation according each step to obtain type definitions and relations/declarations checking using Symbol Table also during this transformations by rules of defined logic:
   * It "visits" each node in the Abstract syntax Tree by subroutines dispatch definitions, using recursive definition approaches.
   * It performs specific semantic validation per node type, usually calling "local methods validation steps implementations". All validation data information is obtained or present using an interface Symbol table system + Abstract syntax-tree, this combination is validated with this system that implements an internal approach with "recursive structure and code block and symbol-type scope definition", over that data set.
   * Using "Stack/data" mechanisms implementation to provide current block /scope definition on that moment and data, in validations. (the most complex aspect in this sub routines in recursive call system process ).
3. **Scope & Type Checks** performs specific action checks based types using code definitions over variables, parameters on code. The compiler ensures program declarations at different blocks are correct during data transformation to instructions (checks existance of variable in Symbol-table for current sub structure, and performs "type compatibility checks" on declarations, expression and if and while parameters are met on every call routines definitions).
4. **Type Checks with code/data blocks by tree**: validates "operations" to know result using methods type definitions: guarantees a "+" or "\*" use types as integer number by check at this methods and reports errors, using recursions calls inside expressions in different tree location structures for validations on type operations on AST. If used with correct type, process will pass, and is possible low level output generation step can start on target transformation in another step (after all validations where performed correctly).
5. **Error Reporting** If errors/ problems exists at type definitions, variables not defined, the system must display text error at code with the context /info on validation failed by methods routines execution when visiting the AST.