SINDH MADRESSATUL ISLAM UNIVERSITY
Aiwan-e-Tijarat Rd, Seari Quarters, Karachi, Karachi City,
Sindh 74000, Pakistan.

What is Assembler?

An assembler is a program that takes basic computer instructions and converts them into a pattern of bits that the computer's processor can use to perform its basic operations. Some people call these instructions assembler language and others use the term assembly language.

What is the responsibility of Loader?

In computer systems a loader is the part of an operating system that is responsible for loading programs and libraries. It is one of the essential stages in the process of starting a program, as it places programs into memory and prepares them for execution.

What is the difference between language converter and language rewriter?

A program that translates between high-level languages is usually called a language translator, source to source translator, or language converter. A language rewriter is usually a program that translates the form of expressions without a change of language.

Why we use cross compilers?

A cross compiler is useful to compile code for multiple platforms from one development host. Direct compilation on the target platform might be infeasible, for example on embedded systems with limited computing resources. Cross compilers are distinct from source-to-source compilers.

In analysis-synthesis model of compilation what is the benefit of analysis?

Identification, measurement, and evaluation of the expected outcomes of available or proposed alternatives.

What we do in the second pass of assembler?

- Examines the operands for symbolic references to storage locations and resolves these symbolic references using information in the symbol table.
- Ensures that no instructions contain an invalid instruction form.
- Translates source statements into machine code and constants, thus filling the allocated space with object code.
- Produces a file containing error messages, if any have occurred.

Is it possible to use a common back end for different front ends?

Yes, it is possible.

What compiler construction tools?

1. Parser generators.
2. Scanner generators.
3. Syntax-directed translation engines.
4. Automatic code generators.
5. Data-flow analysis engines.
6. Compiler-construction toolkits.

What is the difference between syntax tree and parse tree?

A parse tree is a record of the rules (and tokens) used to match some input text whereas a syntax tree records the structure of the input and is insensitive to the grammar that produced it.

Name the layers associated with front end?

1. Lexical Analyzer.
2. Syntax Analyzer.
3. Sematic Analyzer.
4. Intermediate Code generation.
5. Code Optimization.
6. Code Generation.

What happens in semantic analysis?

Semantic analysis is the task of ensuring that the declarations and statements of a program are semantically correct, i.e, that their meaning is clear and consistent with the way in which control structures and data types are supposed to be used. Regular Expression or Finite Automata are used in which phase of compiler. Syntax analysis phase of compiler uses Regular Expressions and Finite Automata to check or validates syntax of the source code with respect to the source language.

What is the difference between Lexeme and pattern?

Pattern: A set of strings in the input for which the same token is produced as output. This set of strings is described by a rule called a pattern associated with the token.

Lexeme: A lexeme is a sequence of characters in the source program that is matched by the pattern for a token.

What is meant by ambiguous grammar?

An ambiguous grammar is a context-free grammar for which there exists a string that can have more than one leftmost derivation or parse tree, while an unambiguous grammar is a context-free grammar for which every valid string has a unique leftmost derivation or parse tree.

What is the importance of syntax analysis phase in compilation process? Why CFGs (Context Free Grammars) are used for defining syntax rules?

Syntax analysis or parsing is the second phase of a compiler. It takes tokens as input and generates a parse tree as output. In this phase compiler checks the syntax of the source code with respect to the source code language whether the input program has correct syntax or not. From the view of programmer, identifying and notifying syntax errors are the main visible feature or property of a compiler, which helps in debugging. On the other hand if the compiler does not do syntax analysis phase which directly means it won't checks syntax of the source code and it will convert it to machine code even if it have syntax error, which leads to critical issues that the program won't be able to work as the programmer wants it to do.

A lexical analyzer can identify tokens with the help of regular expressions and pattern rules. But a lexical analyzer cannot check the syntax of a given sentence due to the limitations of the regular expressions. Regular expressions cannot check balancing tokens, such as parenthesis. Therefore, this phase uses context-free grammar (CFG), which is recognized by push-down automata.

The rules of programming can be entirely represented in some few productions. Using these productions, we can represent what the program actually is. The input has to be checked whether it is in the desired format or not.

What is code optimization? Briefly describe the principle sources of code optimization.

Optimization is a program transformation technique, which tries to improve the code by making it consume less resources (i.e. CPU, Memory) and deliver high speed. In optimization, high-level general programming constructs are replaced by very efficient low-level programming codes. A code optimizing process must follow the three rules given below:

- The output code must not, in any way, change the meaning of the program.
- Optimization should increase the speed of the program and if possible, the program should demand a smaller number of resources.
- Optimization should itself be fast and should not delay the overall compiling process.