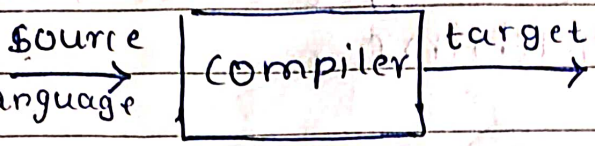
Ch1 Introduction.

Page No.
Date

★ compiler is a translator which translates HLL into LLL or MLL



compiler compiles the whole program whereas interpreter translates /interpret each statement

★ One-pass compiler or single-pass compiler is a compiler that passes through the source code of each compilation unit only once.

★ Two-pass compiler is a compiler which goes through assembly language twice and generate object code.

★ Single-Pass (C)	Multi-Pass (C++)
(i) passes through source code of each compilation unit once	(i) passes multiple times.
(ii) also called narrow compiler	(ii) also called wide compiler
(iii) less efficient code optimization and code generation	(iii) better code optimization and code generat.
(iv) large memory for compilation	(iv) small memory for compilation
(v) faster	(v) slower

Page No.
Date

★ cross compiler
compiler that run on one machine and produce target code for another machine

★ aspects of compilation
(i) compiler generated code which implements meaning of source program in an execution domain.
(ii) compilation process diagnosis the wrong semantics of source program.

★ Bootstrapping is a process in which simple program is used to translate complicated program which in turn used to translate more complicated program.

Ch2 Lexical Analysis

Page No.
Date

★ Lexical analysis is the process of converting a sequence of characters from source program into a sequence of tokens.

★ Token is a sequence of characters that represents a unit of information in source program.

★ Pattern is a rule that describes the character that can be grouped into tokens.

★ Lexeme is a sequence of characters in the source program that matches the pattern for a token.

★ Sentinels is a special characters that cannot be part of source program and natural choice is the character eof

★ Lex is a computer program that generates Lexical analyzer.

★ 3 sections of Lex
(i) declaration section
(ii) rule section
(iii) procedure section.

Date

★ Lex Library functions

(I) yylex() → start or resume scanning

(II) yytext() → Whenever lexer matches token, the text of the token is stored in null terminated string yytext

(III) yywrap() → returns 1 or 0 when yylex() reaches end of the file. → by default returns 1

(IV) yyerror() → reports error to user.