COMPILER DESIGN

HOMEWORK-1

1.Difference between compiler and interpreter.

Ans.

BASIS OF COMPARISON	***COMPILER***	***INTERPRETER***
Function	A compiler coverts high-level language program code into machine language and then executes it.	Interpreter converts source code into the intermediate form and then converts that intermediate code into machine language.
Scanning	Compiler scans the entire program first before translating into machine code.	Interpreter scans and translates the program line by line to equivalent machine code.
Working	Compiler takes entire program as input.	Interpreter takes single instruction as input.
Code Generation	Intermediate object code is generated in case of compiler.	In case of interpreter, no intermediate object code is generated.
Execution Time	Compiler takes less execution time when compared to interpreter.	Interpreter takes more execution time when compared to compiler.
Memory Requirement	Compiler requires more memory than interpreter.	Interpreter needs less memory when compared to compiler.
Modification	If you happen to make any modification in program you have to recompile entire program i.e. scan the whole program every time after modification.	If you make any modification and if that line has not been scanned then no need to recompile entire program.
Speed	Faster as compared to interpreter.	Slower when compared to compiler.

At Execution	There is usually no need to compile program every time(if not modified) at execution time.	Every time program is scanned and translated at execution time.
Error Detection	Compiler gives you the list of all error after compilation of whole program.	Interpreter stops the translation at the error generation and will continue when error gets solved.
Debugging	Compiler is slow for debugging because errors are displayed after entire program has been checked.	Interpreter is good for fast debugging.
Machine Code	Compiler converts the entire program to machine code when all errors are removed execution takes place.	Each time the program is executed; every line is checked for error and then converted into equivalent machine code.
Example	C, COBOL, C#, C++,etc.	Python, Perl, VB, etc.

2.Comparison between compiler, interpreter and assembler.

Ans.

BASIS OF COMPARISON	***COMPILER***	***INTERPRETER***	***ASSEMBLER***
Definition	Software that converts program written in a high level language into machine level language.	Software that translates a high level language program into machine language.	Software that converts programs written in assembly language into machine language.
Functionality	Compiler converts the whole high level language program to	Interpreter converts high level language program to	In contrast, assembler converts assembly language program to machine language.

	machine language at a time.	machine language line by line.	
Written For	Written for particular language.	Written for particular language.	Written for particular hardware.
Translation of instructions	One instruction translates to many instructions(one to many).	One instruction translates to many instructions(one to many).	One instruction translates to one instruction(one to one).
Translation of program	Translates entire program before running.	Translates program instructions by instruction until an either completed or error detected.	Translates entire program before running
Language	Language such as C, C++ use compilers.	Languages such as Ruby, Perl, Python uses interpreter.	Assembly languages uses an assembler.
Working phases	Compiler makes works in six phases over source code.	Interpreter makes works in four phases over source code.	Assembler makes works in two phases over the input.
Debugging	Debugging is easy.	Debugging is easy.	Debugging is difficult.