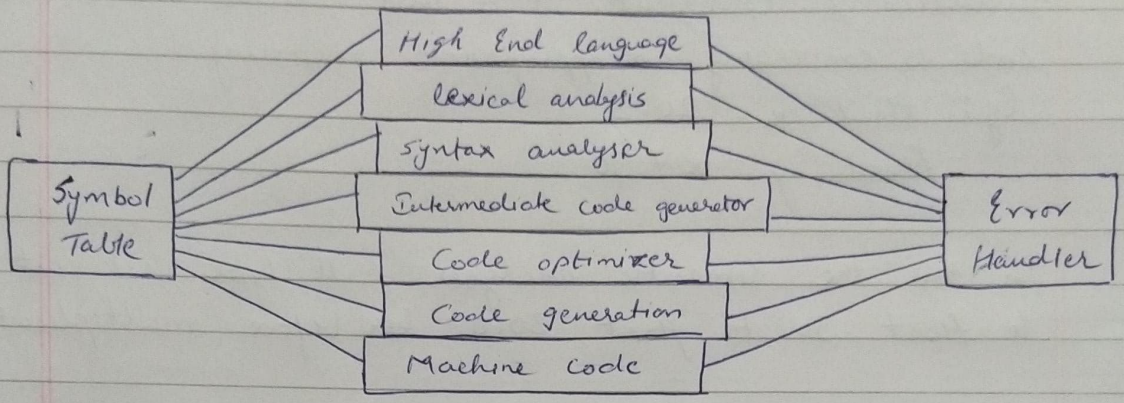
Name - Vasu Kalariya
Roll - PE29
Sub - SSC

Batch-2

Theory Assignment

Q 1 Explain the phase of compiler with suitable example



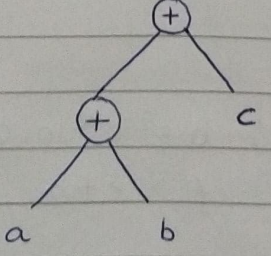
a] Lexical Analyzer: It is the first phase when compiler scans the source code. This process be left to right by character & group these character into tokens.

Eg: x = y+10

Tokens: x → identifier , Y → identifier
'=' → Assignment operator
'+' → Addition operator
10 → Number.

b] Syntax Analysis: It is all about discovering structure in code it determines whether or not text follow an expected format. The main aim is to make sure that the source code written by programmer is correct or not.

Eg (a+b) * c → Parse tree



c] Semantic analysis: checks the semantic consistency of the code. It uses the syntax tree of the previous phase along with the symbol table to verify that the given source code is semantically consistent. It is also checks whether the code is converging an appropriate meaning.

Eg: float x = 20.2
float y = x + 30

In this the semantic analyser will type cast int 30 to float 30 to float 30.0 ~~bef~~ before multiplication

d] Intermediate code generation: It is b/w the high level & machine level language. This intermediate code needs to be generated in such a manner that makes it easy to translate it into the target M.C.

Eg: total = count + rate * 5

Intermediate code with the help of address code method is:

t1 = int_to float (5)
t2 = rate * t1
t3 = count + t2
total = t3

e] Code optimization: This phase removes unnecessary code line & arrange the sequence of statement to speed up the execution of program w/o wasting resource

Eg. a = into float (10)
b = c * a
d = e + b
f = d

→ can become: b = c * 10.0
f = e + b

f) Code generation: It gets input from the code optimization phase & produces the pages code or object code as a result. The objective of this phase is to allocate storage & generate relocatable machine code

Eg: a = b + 60.0

would possibly be translated to registers.

```
MOVF   a, R1
MVLF   #60.0, R2
ADDF   R1, R2
```

Q 2 Define & give examples: Token, Lexeme, Pattern.

Token: It is a sequence of characters that can be treated as a single logical entity typical tokens areas.
(1) Identifier (2) keywords (3) operators (4) special symbols (5) constants

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

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

Example:

Token	Lexeme	Pattern
const. if	const. if	const if
relation	<, <=, <>, =, >	< or < = or = or <> or = >
i	pi	any char b/w "and" & "concept"
num	3.14	
literal	"core"	Pattern

Q 3 What is the purpose of Symbol table?

Symbol table is an important data structure created & maintained by compilers in order store info about the occurance of variable by entities such as variable names, function, names objects interface etc

Q 4 Write a LEX program to count the number of vowels constants in given string.

```
%{
    int vow_count=0;
    int const_count=0;
%}
%%
[aeiouAEIOU] {vow_count++;}
[a-zA-Z] {const_count++;}
%%
int yywrap() { }
int main()
{ printf("Enter the string of vowels & constants:");
  yylex();
  printf("No of vowels are: %d \n", vow_count);
  printf("No. of constants are: %d \n", const_count);
  return 0;
}
```

Output:

Enter the string of vowels & constants: Good Morning.
No. of vowels : 4
No of constants : 7