Compiler

1. what do you understand by longest pattern matching rule in LEX tool?

2. define a token

3. what do you understand by backtracking of a compiler. discuss in context with left factoring

4. what is left factoring and how left factoring is eliminated from the grammar?

5. what are the different conflicts that occur during shift-reduce parsing

6. give example of l-atributed definition

7. describe dependency graph in SDT

8. what are the various methods of implementing three address statements ?

9. what are the triplets prrferred over quadraples as intermediate code?

10. what are activation records?

PART-B

2 a) write short notes on i)buffer pairs ii)sentinentals

or

b) i)give an example of lexical error. how lexical errors are handled

ii)briefly explain about compiler construction tools

3 a)i)consider the grammar:

S-->Rajaaja

R-->AB

A-->AR|AT|B

T-->TB|A

find whether grammar is left or right recursive? find appropriate solution to convert it to non-recursive grammar.

a)what do you men by an ambigious grammar?consider the following grammar.

E-->E+E|E*E|1.

Check whether the grammar is ambigious for the input string:1+1*1.

Give an appropriate to rewrite the grammar to remove ambiguity.

OR

b)(i)Consider the context free grammar.

S-->SS+|SS*|a

and the string aa+a*.

a)Give a leftmost derivation for the string.

b)give the rightmost deivation for the string.

c)give the parse tree for the string.

d)is the grammar ambigious or unambigious ?justify.

(ii)Define a context free grammar with the help of an example.

Q4. a)consider the grammar with the folllowing translation rules and E as the start symbol.

E-->E*T (TE value =E value *T value).

IF (Evalue =t value).

T-->T&F (T value=T value+F.value).

IF (T value=F value).

F-->num(F.value=num.value).

Compute E.value for the root of the parse tree for expression:2#3&5#6&4.

b)(i)For the given syntax directed definitions draw the annotated parse trees for the expression 1*2*3*(4+5)n

PRODUCTIONS SEMANTIC RULES.

L->En L val=E.val

E->E1+T E.val=E1.val++T.val

E->T E.val=T.val

T->T1*F T.val=T1.val*F.val.

T->F T.val=F.val>

F->(E) F.val=E.val

F->digit F.val=digit*lexval.

(ii) for the given syntax directed definition draw the annotated parse trees for the expression : float w,x,y

PRODUCTIONS	SEMANTIC RULES
D->TL	L.inh=T.type
T->int	T.type=integer
T->float	t.type=float
L->L1,id	L1.inh=L.inh;addtype(id.entry,L.inh)
L->id	addtype(id.entry,L.inh)

5 a)write three adress code,quadraple , triples, syntax tree and DAG for the the following expressions:(consider operator procedure as per ANSIC)

i. a=b*-c+b*-c ii. -(a+b)*(c+d)+(a+b+c)

or

b)(i) write the three adress code for the following statement:

```
int a[10],b[10],d,i;

d=0;

for(i=0;i<10;i++)d+=a[i]*b[i];
```

ii)write the three adress code for the following code:

```
int fact(int n)

{

if n==0) return1;

else return (n*fact(n-1));
```

}

6 a)consider quicksort example pseudocoe,show downward-growing stack of activation record and activation tree for quicksort example

```
 quicksort(A,p,r){
 if(p<r){
q<-partition(A,p,r)
quick sort(A,p,r)
quick sort (A,q+1,r)
}
}
partition(A,p,r){
a<-A[p]
i<-p-1
j<- r+1
while(true){
repeat
j<- j-1
until (a[j]<=x)
repeat
i<-i+1
until (A[i]>=x)
if(i<j)
swap(A[i],A[j])
else
return(j)
```

}

}

or

b i)consider following code:

int fib(n){

if(n==0) return 1;

if (n==1) return 1;

return fib(n-1)+fib(n-2);

}

draw the activation tree for n=5

ii) write short notes on following

i)activation trees ii)activation record