16/03/16

Module - 5

Intermediate

1) Postfix notation
2) Abstract syntax tree
3) Three Address code
   - Quadruple
   - Triple
   - Indirect

→ Postfix notation

a+s + p * q / d

((a+s) + ((p*q)/d))

A) as + pq * d/+

Advantages:

It eliminate the need of parathesis

Easy to Evaluate and analyse.

Disadvantages:

It is not suitable for code Optimisation.

Abstract Syntax Tree:

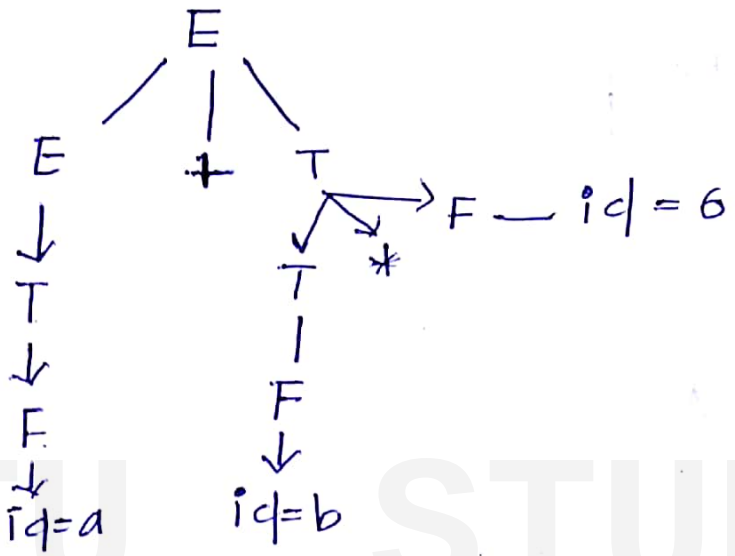
a+b*c

E → E+T / T

T → T*F / F

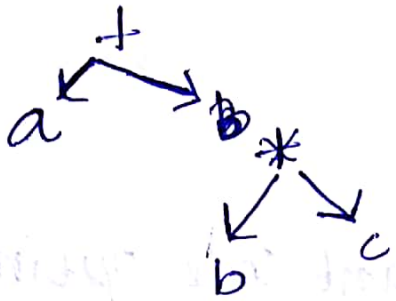
F → id

The Parse tree form the input stream a+b*c



Abstract Syntax Tree: It is the graphical representation of intermediate code. Hiding implementation details from high level

AST



→ Representation of AT in memory :-

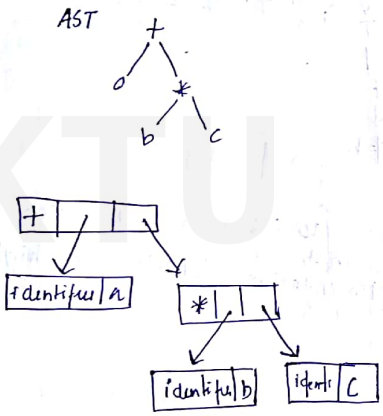
Operator node

oper	Pointer to oprnd 1	Pointer to oprnd 2

Operand node.

Type	data.

AST



Easy to create

It supports m/c independant code optimisation.

Three Address Code TAC

Sequence of stmt of the form:

C = A op B

where, A, B and C are program defined names

compiler generated temprary name

OP- stands for operator.

Statement.	TAC
count = index + 20.	count = index + 20
Pnew = P + (P * n * r)/100	t_1 = P * n t_2 = t_1 * r t_3 = t_2 / 100 t_4 = P + t_3 Pnew = t_4.

Pnew = P + (P * n * r) / 100

A = B * C / D.

A = ((B * C) / D)

- t_1 = B * C
  t_2 = t_1 / D.
  Anew = t_1 / D