COMPILER CAT 2 ANSWERS

Q.1 a) | What do you mean by SDTS. Explain with example.

In syntax directed translation, along with the grammar we associate some informal notations and these notations are called as semantic rules.

So we can say that

1. Grammar + semantic rule = SDT (syntax directed translation)

- In syntax directed translation, every non-terminal can get one or more than one attribute or sometimes 0 attribute depending on the type of the attribute. The value of these attributes is evaluated by the semantic rules associated with the production rule.
- In the semantic rule, attribute is VAL and an attribute may hold anything like a string, a number, a memory location and a complex record
- In Syntax directed translation, whenever a construct encounters in the programming language then it is translated according to the semantic rules define in that particular programming language.

Example

Production	Semantic Rules
E → E + T	E.val := E.val + T.val
E → T	E.val := T.val
T → T * F	T.val := T.val + F.val
T → F	T.val := F.val
F → (F)	F.val := F.val
F → num	F.val := num.lexval

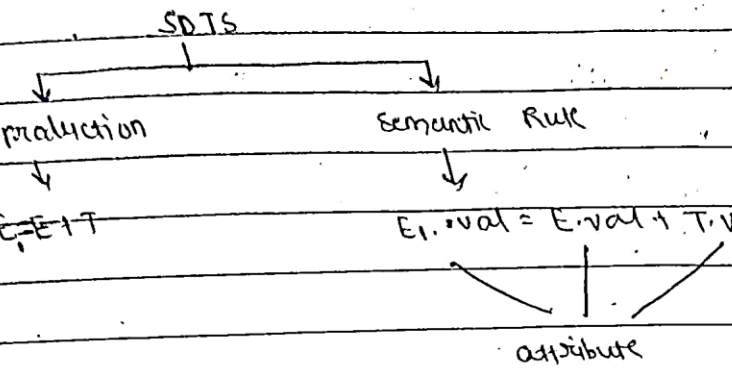
E.val is one of the attributes of E.

num.lexval is the attribute returned by the lexical analyzer.

* Syntax Directed Translation Scheme :—>

↳ SDTS

- production with grammar symbol
- semantic rule with attribute value.



Syntax directed translation Scheme :

We know that every programming language contains a construct (a language construct is a syntatically allowable part of program that may be from one or more lexical tokens in accordance with the rule of programming language), so it is very imp. to know the rules of translation of programming construct whenever a construct encounter and in programming language, it is evaluate, translatated a/c to the semantic rules defined in that particular programming programming language. The translation may be generation one of the intermediate code, object code or adding info. in the simple table, about the construct type

Syntax structure of programming language specify the property of construct. The context free grammar is used

Specify the syntactical structure of programming language, for that we add attribute with the grammar symbol and semantic rules with the production to make the translation of construct easy.

what were the actions

The Syntax analyzer directs the whole process during the parsing of source code

(1) call the lexical analyzer whenever the syntax analyzer want another token

(2) perform the action of semantic analyzer.

(3) perform the action of intermediate code generator.

Syntax Directed Definition is a generalization of a contex free grammar (CFG). It is called attribute grammar. Syntax Directed Translation is an extension of CFGs. It helps compiler designer to translate language constructs directly by attaching semantic actions or subroutines Along with each production of grammar, we attach a semantic action. (translation rule, semantic rule, action). The grammar together with semantic action is called syntax-directed translation (SDT)

$A \rightarrow \alpha \quad \{ \text{Action} \} \Leftarrow (SDT)$

Semantic actions are attached with grammar rules to perform following tasks:

1) To store retrieve type information in symbol table.
2) To perform type checking & parameter checking.
3) To issue error messages.
4) To build syntax tree
5) To generate intermediate code.

Q.1 b) | Define Attribute. Explain different types of attributes.

* Attribute :

Attributes are associated information with the language construct by attaching them to the grammar symbol representing that construct

SDTS

Production	Semantic Rule
$E \rightarrow E_1 + T$	$E.val = E_1.val + T.val$
$E \rightarrow T$	$E.val = T.val$
$T \rightarrow T_1 * F$	$T.val = T_1.val * F.val$
$T \rightarrow F$	$T.val = F.val$

Types of attributes :-

(1) Synthesized Attribute

(2) Inherited Attribute

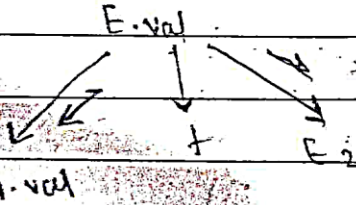
① Synthesized Attributes :-

} parent node

} child node

An attribute at a node is synthesized if its value is computed from the attributed values of the children of that node in the parse tree.

② Inherited Attribute :-



} parent node

} child node

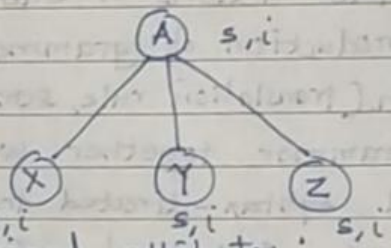
An attribute at a node is inherited if its value is computed from the attribute value of the sibling / parent of that node in the parse tree.

Attribute for Grammar Symbol

In SDT, each grammar symbol has an associated set of attributes. Grammar symbol has n-number of attributes. Attributes may be type, value, address, pointer, string.

The attributes associated with Grammar symbols are classified into two types

$A \to XYZ$



① Synthesized attributes:

The value of a synthesized attribute at a node is computed from the value of attributes at the children of that node in the parse tree.

$\therefore A.s = f(X.s + Y.s + Z.s)$

$A \to XYZ \quad \{A.s = X.s + Y.s + Z.s\}$

s is synthesized attribute

② Inherited attribute:

The value of an inherited attribute is computed from the values of attributes at the siblings and parent of that node.

$\therefore Y.i = f(X.i + Z.i + A.i)$

$\therefore A \to XYZ \quad \{Y.i = A.i + X.i + Z.i\}$

i is an inherited attribute.

Q.2 a) | Translate the expression
A: = -B * (C + D)/E

Q.2 b)	Translate given expression into TAC if x < y then a= b + c else p= q+r

0) if (x < y) goto 2

1) goto 5

2) $t_1 = b + c$

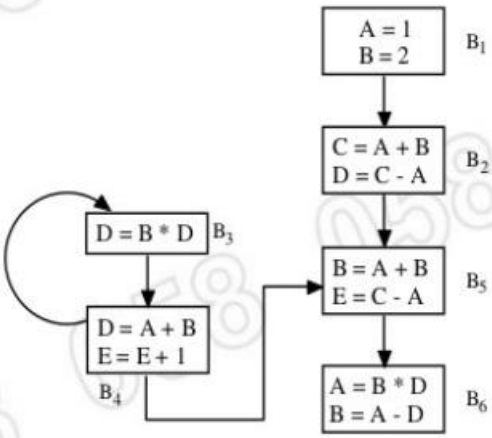
3) $a = t_1$

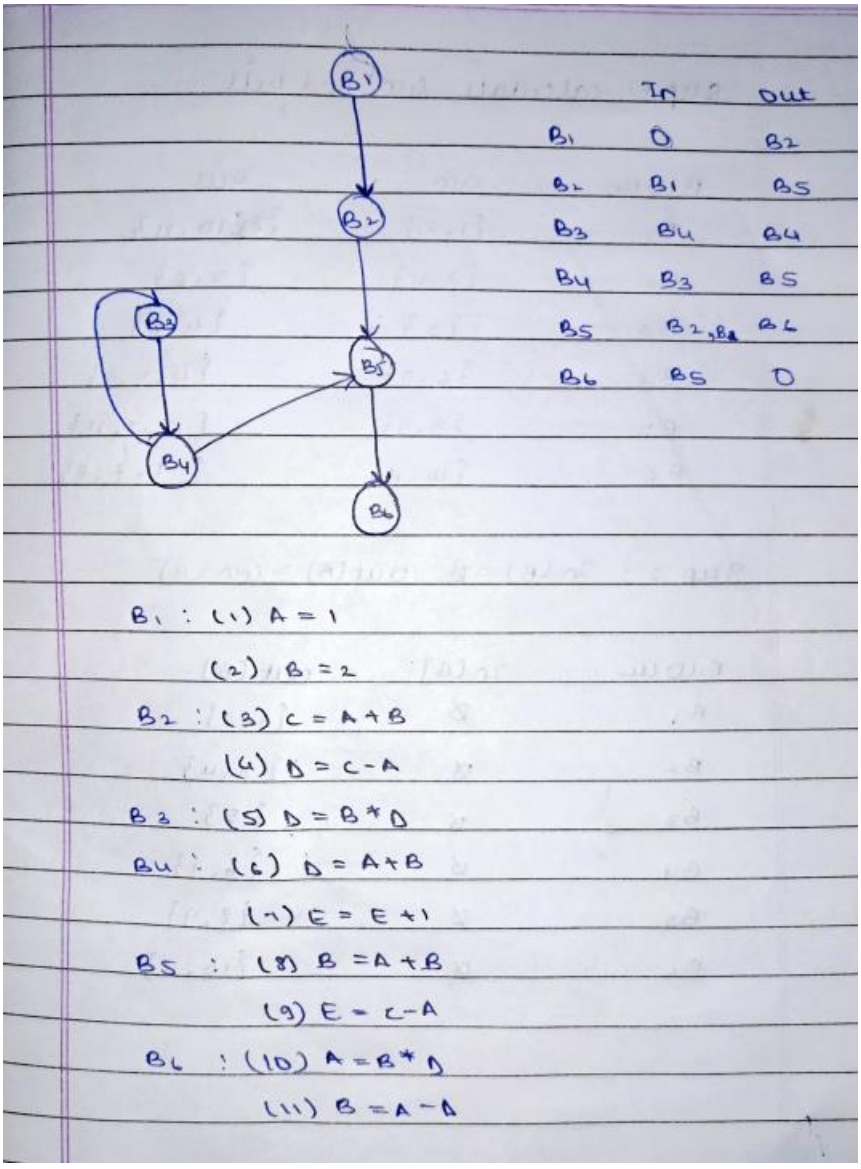
4) goto S. next

5) $t_2 = q + r$

6) $p = t_2$

Q.4 a) Find IN and OUT for every blocks for the following graph





Step 1: calculate Gen And kill

BLOCK	Gen	kill
B_1	{1, 2}	{8, 10, 11}
B_2	{3, 4}	{5, 6}
B_3	{5}	{4, 6}
B_4	{6, 7}	{4, 5, 9}
B_5	{8, 9}	{2, 7, 11}
B_6	{10, 11}	{1, 2, 8}

Step 2 : In(B) = ∅ Out(B) = Gen(B)

BLOCK	In[B]	Out[B]
B_1	∅	{1, 2}
B_2	∅	{3, 4}
B_3	∅	{5}
B_4	∅	{6, 7}
B_5	∅	{8, 9}
B_6	∅	{10, 11}

Step 3:

BLOCK	In[B]	Out[B]
B_1	$\emptyset$	$\{1,2\}$
B_2	$\{1,2\}$	$\{1,2,3,4\}$
B_3	$\{6,7\}$	$\{5,7\}$
B_4	$\{5\}$	$\{6,7\}$
B_5	$\{3,4,6,7\}$	$\{3,4,6,8,9\}$
B_6	$\{8,9\}$	$\{9,10,11\}$

$Out(B_1) = O - \{8,10,11\} \cup \{1,2\}$

$= \{1,2\}$

$Out(B_2) = \{1,2\} - \{5,6\} \cup \{3,4\}$

$= \{1,2,3,4\}$

$Out(B_3) = \{6,7\} - \{4,6\} \cup \{5\}$

$= \{5,7\}$

$Out(B_4) = \{5\} - \{4,5,9\} \cup \{6,7\}$

$= \{6,6,7\}$

$Out(B_5) = \{3,4,6,7\} - \{5,7,11\} \cup \{8,9\}$

$= \{3,4,6,8,9\}$

Out $(B_6) = \{8,9\} - \{1,2,8\} \cup \{10,11\}$

$= \{9,10,11\}$

Step 4:

Block	In[B]	Out (B)
B_1	$\emptyset$	$\{1,2\}$
B_2	$\{1,2\}$	$\{1,2,3,4\}$
B_3	$\{6,7\}$	$\{5,7\}$
B_4	$\{5,7\}$	$\{6,7\}$
B_5	$\{1,2,3,4,6,7\}$	$\{1,3,4,6,8,9\}$
B_6	$\{3,4,6,8,9\}$	$\{3,4,6,9,10,11\}$

Out $(B_1) = \emptyset - \{8,10,11\} \cup \{1,2\}$

$= \{1,2\}$

Out $(B_2) = \{1,2\} - \{5,6\} \cup \{3,4\}$

$= \{1,2,3,4\}$

Out $(B_3) = \{6,7\} - \{4,6\} \cup \{5\}$

$= \{5,7\}$

Out $(B_4) = \{5,7\} - \{4,5,9\} - \{6,7\}$

$= \{6,7\}$

Out (B5) = {1,2,3,4,6,7} - {2,7,11} ∪ {8,9}
= {1,8,3,4,6,8,9}

Out (B6) = {3,4,6,8,9} - {1,2,8} ∪ {10,11}
= {3,4,6,9,10,11}

Step 5 :

Block	In [B]	Out [B]
B_1	0	{1,2}
B_2	{1,2}	{1,2,3,4}
B_3	{6,7}	{5,7}
B_4	{5,7}	{6,7}
B_5	{1,2,3,4,6,7}	{1,3,4,6,8,9}
B_6	{1,3,4,6,8,9}	{3,4,6,9,10,11}

Out (B1) = 0 - {8,10,11} ∪ {1,2}
= {1,2}

Out (B2) = {1,2} - {5,6} ∪ {3,4}
= {1,2,3,4}

Out (B3) = {6,7} - {4,6} ∪ {5}
= {5,7}

Out (B4) = {5,7} - {4,5,9} ∪ {6,7}
= {6,7}

Out (B5) = {1,2,3,4,6,7} - {2,7,11} ∪ {8,9}
= {1,3,4,6,8,9}

Out (B6) = {1,3,4,6,8,9} - {1,2,8} ∪ {10,11}
= {3,4,6,9,10,11}