ACE

Q. Generation of intermediate code based on an abstract machine model is useful in compilers because **(GATE - 94)**

(a) It makes implementation of lexical analysis and syntax analysis easier

(b) Syntax-directed translations can be written for intermediate code generation

(c) It enhances the portability of the front end of the compiler

(d) It is not possible to generate code for real machines directly from high level language programs.

ACE

Q. In the context of compilers, which of the following is/are NOT an intermediate representation of the source program? **(GATE - 21-Set2)**

[MSQ]

(a) Abstract Syntax Tree (AST)

(b) Three address code

(c) Control Flow Graph (CFG)

(d) Symbol table

06. Which of the following is in the quadruple form for the expression a or b and c or d (Assume 'and' has higher precedence than 'or'). Assume each quadruple is numbered from 1.

(a) (1) (or, a, b, T_1)
(b) (2) (and, b, c, T_1)
(c) (2) (or, a, T_1, T_2)
(d) None

(1) and b c T_1
(2) or a T_1 T_2
(3) or T_2 d T_3

07. The triples of a*–(b + c) is

(a) 1. (+, b, c)
2. (NEG, (1),)
3. (*, a, (2))

(b) 1. (+ , b, c)
2. (*, a, (1))
3. (NEG, (2), –)

(c) 1. (*, a, b)
2. (+, b, c)
3. (NEG, (1), (2))

(d) 1. (+, b, c, T1)
2. (*, a, T1, T2)
3. (NEG, T2, T3, -)

08. Consider the following code segment.
x = u – t;

code?

(a) $p_1 = a - b$
$q_1 = p_1 * c$
$p_1 = u * v$
$q_1 = p_1 + q_1$

(b) $p_3 = a - b$
$q_4 = p_3 * c$
$p_4 = u * v$
$q_5 = p_4 + q_4$

(c) $p_1 = a - b$
$q_1 = p_2 * c$
$p_3 = u * v$
$q_2 = p_4 + q_3$

(d) $p_1 = a - b$
$q_1 = p * c$
$p_2 = u * v$
$q_2 = p + q$

10. Choose the incorrect statement
(a) DAG representation of a basic block enables elimination of common subexpression.
(b) DAG representation of a basic block enables dead code elimination.
(c) DAG representation allows optimal register

07. The triples of a*–(b + c) is

(a) 1. (+, b, c)
2. (NEG, (1),)
3. (*, a, (2))

1.	+ b c
2	NEG (1) –
3	* a (2)

(b) 1. (+ , b, c)
2. (*, a, (1))
3. (NEG, (2), –)

(c) 1. (*, a, b)
2. (+, b, c)
3. (NEG, (1), (2))

(d) 1. (+, b, c, T1)
2. (*, a, T1, T2)
3. (NEG, T2, T3, -)

08. Consider the following code segment.
x = u – t;
y = x * v;
x = y + w;
y = t – z;
y = x * y;
The minimum number of total variables required to convert the above code segment to static single assignment form is ______.

$q_1 = p_2 * c$ $q_1 = p * c$
$p_3 = u * v$ $p_2 = u * v$
$q_2 = p_4 + q_3$ $q_2 = p + q$

10. Choose the incorrect statement
(a) DAG representation of a basic block enables elimination of common subexpression.
(b) DAG representation of a basic block enables dead code elimination.
(c) DAG representation allows optimal register usage by specifying & controlling the lifetime of variables in registers.
(d) none

11. The DAG for a*b*b is
(a) *
(b) *

t = t − z
u = u * t
min variables = 5
u, t, v, w, z

y = x * y;

The minimum number of total variables required to convert the above code segment to static single assignment form is 5+5 = 10

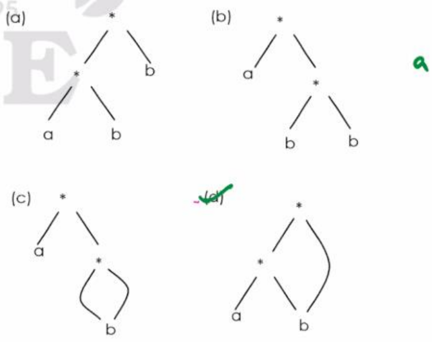
Gate-17

09. Consider the following intermediate program in three address code

p = a − b
q = p * c
p = u * v
q = p + q

Which one of the following corresponds to a static single assignment form of the above

11. The DAG for a*b*b is



Zoom Webinar · OneNote for Windows 10 · ramesh masuna · Home Insert Draw View Help · Share

(c) (2) (or, a, T_1, T_2)

(d) None

(1) and b c T1
(2) or a T1 T2
(3) or T2 d T3

07. The triples of a*–(b + c) is

(a) 1. (+, b, c)
2. (NEG, (1),)
3. (*, a, (2))

(b) 1. (+ , b, c)
2. (*, a, (1))
3. (NEG, (2), –)

(c) 1. (*, a, b)
2. (+, b, c)
3. (NEG, (1), (2))

(d) 1. (+, b, c, T1)
2. (*, a, T1, T2)
3. (NEG, T2, T3, -)

1	+ b c	
2	NEG (1)	–
3	* a (2)	

Gate-15

08. Consider the following code segment.

x = u − t;
y = x * v;
x = y + w;
y = t − z;
y = x * y;

y = x*y
= (y+w) * (t−z)
= ((x*v)+w) * (t−z)
= (((u−t)*v)+w) * (t−z)

TAC
u = u − t
u = u * v
u = u + w
t = t − z
u = u * t
min variables = 5

The minimum number of total variables

(c) p_1 = a − b
q_1 = p_2 * c
p_3 = u * v
q_2 = p_4 + q_3

(d) p_1 = a − b
q_1 = p_1 * c
p_2 = u * v
q_2 = p + q

10. Choose the incorrect statement

(a) DAG representation of a basic block enables elimination of common subexpression.

(b) DAG representation of a basic block enables dead code elimination.

(c) DAG representation allows optimal register usage by specifying & controlling the lifetime of variables in registers.

(d) none

11. The DAG for a*b*b is

1.	+	b	c
2	NEG (1)	–	
3	*	a	(2)

2. (NEG, (1),)
3. (*, a, (2))

2. (*, a, (1))
3. (NEG, (2), –)

~~(c)~~ 1. (*, a, b)
2. (+, b, c)
3. (NEG, (1), (2))

~~(d)~~ 1. (+, b, c, T1)
2. (*, a, T1, T2)
3. (NEG, T2, T3, -)

Gate-15
08. Consider the following code segment.

```
x = u – t;
y = x * v;
x = y + w;
y = t – z;
y = x * y;
```

$Y = x*y$
$= (y+w) * (t-z)$
$= ((x*v)+w) * (t-z)$
$= (((u-t)*v)+w) * (t-z)$

TAC
u = u–t
u = u*v
u = u+w
t = t–z
u = u*t

min variables = 5
u, t, v, w, z

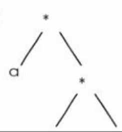
The minimum number of total variables required to convert the above code segment to static single assignment form is 5+5 = 10

Gate-17
09. Consider the following intermediate program in three address code

10. Choose the incorrect statement
(a) DAG representation of a basic block enables elimination of common subexpression.
(b) DAG representation of a basic block enables dead code elimination.
(c) DAG representation allows optimal register usage by specifying & controlling the lifetime of variables in registers.
(d) none

11. The DAG for a*b*b is

(a)


(b)


$q_2 = p_4 + q_3$ $q_2 = p + q$

1.	+	b	c
2	NEG (1)	–	
3	*	a	(2)

2. (NEG, (1),)
3. (*, a, (2))

2. (*, a, (1))
3. (NEG, (2), –)

~~(c)~~ 1. (*, a, b)
2. (+, b, c)
3. (NEG, (1), (2))

~~(d)~~ 1. (+, b, c, T1)
2. (*, a, T1, T2)
3. (NEG, T2, T3, -)

Gate-15
08. Consider the following code segment.

```
x = u – t;
y = x * v;
x = y + w;
y = t – z;
y = x * y;
```

$Y = x*y$
$= (y+w) * (t-z)$
$= ((x*v)+w) * (t-z)$
$= (((u-t)*v)+w) * (t-z)$

TAC
u = u–t
u = u*v
u = u+w
t = t–z
u = u*t

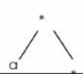
min variables = 5
u, t, v, w, z

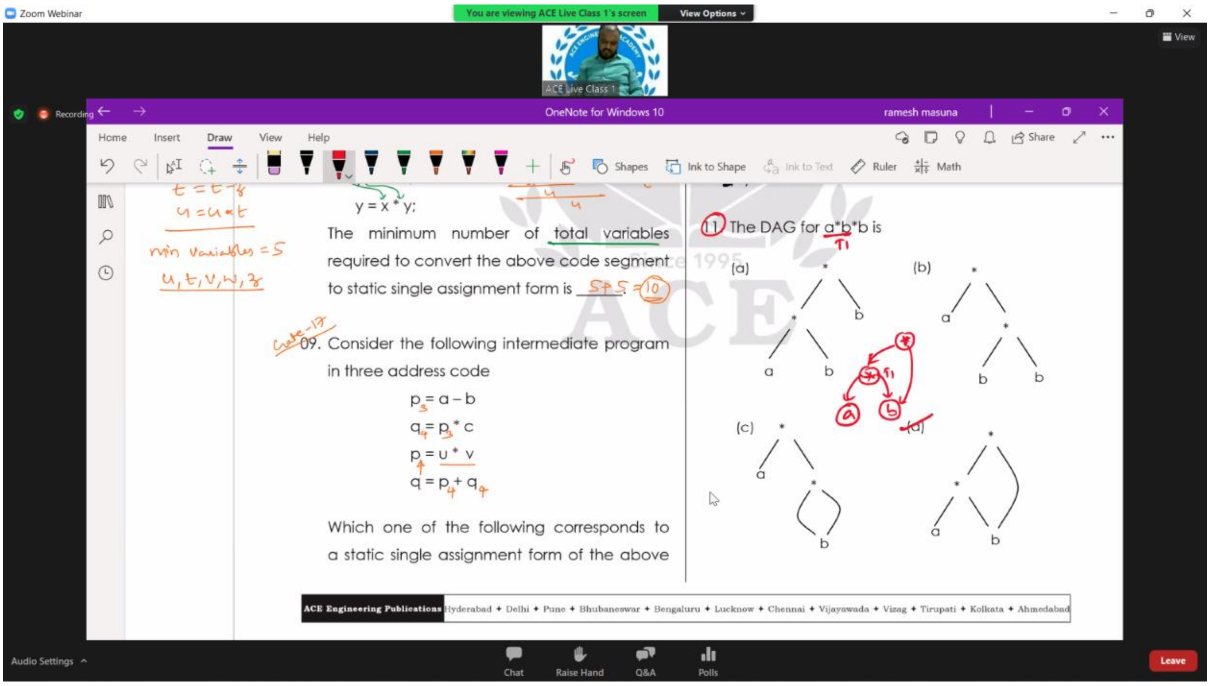
The minimum number of total variables required to convert the above code segment to static single assignment form is 5+5 = 10

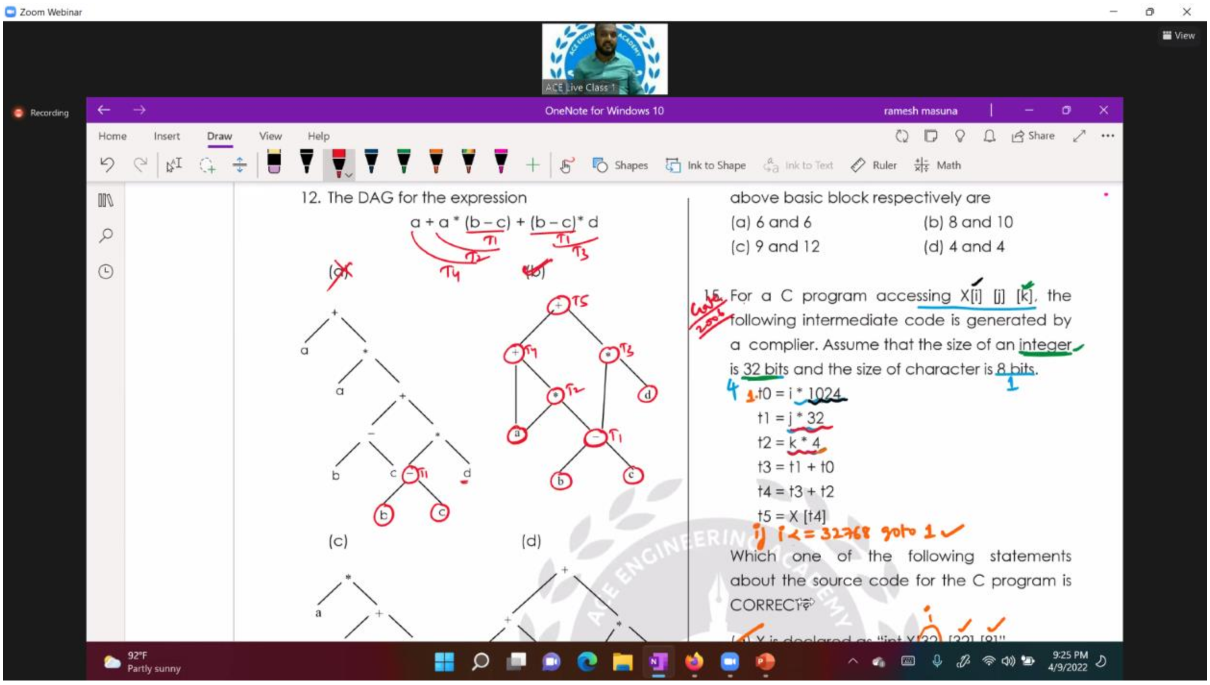
10. Choose the incorrect statement
(a) DAG representation of a basic block enables elimination of common subexpression. (T)
(b) DAG representation of a basic block enables dead code elimination. (T)
(c) DAG representation allows optimal register usage by specifying & controlling the lifetime of variables in registers. (T)
✓(d) none

11. The DAG for a*b*b is

(a)


(b)



Zoom Webinar
You are viewing ACE Live Class 1's screen
View Options
ACE Live Class 1
Recording
OneNote for Windows 10
ramesh masuna
Home Insert Draw View Help
Shapes Ink to Shape Ink to Text Ruler Math
y = x * y;
min variables = 5
u, t, v, w, z
The minimum number of total variables required to convert the above code segment to static single assignment form is 5+5 = 10
Gate-17
09. Consider the following intermediate program in three address code
p = a – b
q = p * c
p = u * v
q = p + q
Which one of the following corresponds to a static single assignment form of the above
11. The DAG for a*b*b is
(a)
(b)
(c)
ACE Engineering Publications Hyderabad + Delhi + Pune + Bhubaneswar + Bengaluru + Lucknow + Chennai + Vijayawada + Vizag + Tirupati + Kolkata + Ahmedabad
Audio Settings
Chat
Raise Hand
Q&A
Polls
Leave


Zoom Webinar
ACE Live Class 1
Recording
OneNote for Windows 10
ramesh masuna
Home Insert Draw View Help
Shapes Ink to Shape Ink to Text Ruler Math
12. The DAG for the expression
a + a * (b – c) + (b – c) * d
(c)
(d)
above basic block respectively are
(a) 6 and 6
(b) 8 and 10
(c) 9 and 12
(d) 4 and 4
15. For a C program accessing X[i] [j] [k], the following intermediate code is generated by a complier. Assume that the size of an integer is 32 bits and the size of character is 8 bits.
t0 = i * 1024
t1 = j * 32
t2 = k * 4
t3 = t1 + t0
t4 = t3 + t2
t5 = X [t4]
Which one of the following statements about the source code for the C program is CORRECT?
92°F
Partly sunny
9:25 PM
4/9/2022

Zoom Webinar

ACE Live Class 1

Recording

OneNote for Windows 10 ramesh masuna

Home Insert Draw View Help

Shapes Ink to Shape Ink to Text Ruler Math

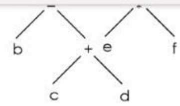
GATE 2015

Q4. Consider the basic block given below.

$a = b + c$
$c = a + d$
$d = b + c$
$e = d - b$
$a = e + b$

$\Rightarrow$
$a = e + b$
$= d - \cancel{b} + \cancel{b}$
$= b + c$
$= b + (a + d)$
$= b + ((b+c) + d)$

The minimum number of nodes and edges present in the DAG representation of the



Represents the assignment

$a = (b - (c+d)) + e*f;$

Convert this tree to (op, arg1, arg2) triples, following these rules:

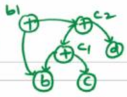
ACE Engineering Publications Hyderabad • Delhi • Pune • Bhubaneswar • Bengaluru • Lucknow • Chennai • Vijayawada • Vizag • Tirupati • Kolkata • Ahmedabad

$c = b + c$
$c = c + d$
$b = b + c$
TAC with min variables

$\Longrightarrow$

$c_1 = b + c$
$c_2 = c_1 + d$
$b_1 = b + c_2$
SSA

$\Rightarrow$



6-nodes
6-edges

ACE Engineering Publications | 17 | Objective Practice Questions

92°F Partly sunny 9:38 PM 4/9/2022

Zoom Webinar

ACE Live Class 1

Recording

OneNote for Windows 10 ramesh masuna

Home Insert Draw View Help

Shapes Ink to Shape Ink to Text Ruler Math

GATE 2015

Q4. Consider the basic block given below.

The minimum number of nodes and edges present in the DAG representation of the

Represents the assignment

Convert this tree to (op, arg1, arg2) triples, following these rules:

ACE Engineering Publications Hyderabad • Delhi • Pune • Bhubaneswar • Bengaluru • Lucknow • Chennai • Vijayawada • Vizag • Tirupati • Kolkata • Ahmedabad

TAC with min variables

SSA

6-nodes
6-edges

ACE Engineering Publications | 17 | Objective Practice Questions

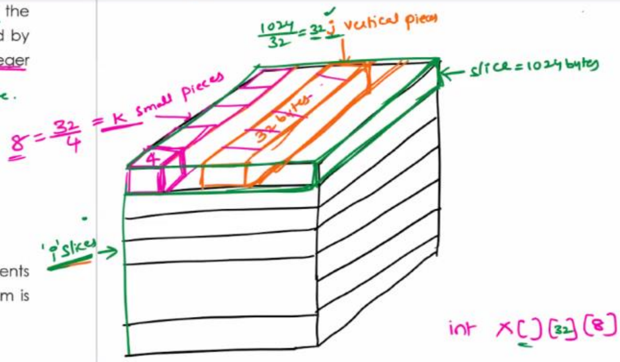
92°F Partly sunny 9:39 PM 4/9/2022

15. For a C program accessing X[i] [j] [k], the following intermediate code is generated by a complier. Assume that the size of an integer is 32 bits and the size of character is 8 bits.

t0 = i * 1024
t1 = j * 32
t2 = k * 4
t3 = t1 + t0
t4 = t3 + t2
t5 = X [t4]

Which one of the following statements about the source code for the C program is CORRECT?

(a) X is declared as "int X[32] [32] [8]"
(b) X is declared as "int X [4] [1024] [32]"
(c) X is declared as "char X[4] [32] [8] "
(d) X is declared as "char X[32] [16] [2]"



---

13. Consider the following block of three address statements

a = c + d
e = a + d
d = c + d

Then the number of nodes in the DAG constructed for the above block of statement is 4.

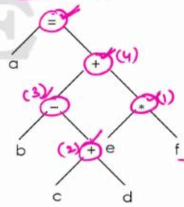
14. Consider the basic block given below.

a = b + c
c = a + d
d = b + c
e = d – b
a = e + b

The minimum number of nodes and edges present in the DAG representation of the

(d) X is declared as "char X[32] [16] [2]"

16. The following syntax tree:



	op	arg1	arg2
(1)	*	e	f
(2)	+	c	d
(3)	–	b	(2)
(4)	+	(3)	(1)
(5)	=	a	(4)

Represents the assignment

a = (b – (c+d)) + e*f;

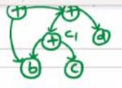
Convert this tree to (op, arg1, arg2) triples, following these rules:

$c = b+c$

$c = c+d \Rightarrow c_2 = c_1+d$

$b = b+c \quad b_1 = b+c_2$

TAC with min variables — SSA



6-nodes
6-edges

ACE Engineering Publications	17	Objective Practice Questions

(1) Evaluate the right subtree of a node before the left subtree.
(2) Number the instructions (1), (2).

Then, identify from the list below, the one triple, with its instruction number, that would appear in your translation.

~~(a)~~ (1) [+, e, f] ~~(b)~~ (3) [*, e, f]
(c) (3) [–, b, (2)] (d) (4) [–, b, (1)]

13. Consider the following block of three address statements

a = c + d
e = a + d
d = c + d

C.S.E

Then the number of nodes in the DAG constructed for the above block of statement is 4.

14. Gate 2015
Consider the basic block given below.

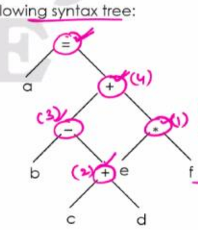
a = b + c
c = a + d
d = b + c
e = d – b
a = e + b

$a = e+b = d - \cancel{b} + \cancel{b} = b+c = b+(a+d) = b+((b+c)+d)$

The minimum number of nodes and edges present in the DAG representation of the

15. X is declared as "char X[32] [16] [2]"

16. The following syntax tree:



	op	arg1	arg2
(1)	*	e	f
(2)	+	c	d
(3)	–	b	(2)
(4)	+	(3)	(1)
(5)	=	a	(4)

Represents the assignment
a = (b – (c+d)) + e*f;
Convert this tree to (op, arg1, arg2) triples, following these rules: