5 Intermediate Code Generation

1. Generation of intermediate code based on an abstract machine model is useful in compilers because:
   (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 .

2. Which one of the following is a purpose of using intermediate code in compilers:
   (A) make parsing and semantic analysis simpler.
   (B) improve error recovery and error reporting.
   (C) increase the chances of reusing the machine-independent code optimizer in other compilers.
   (D) improve the register allocation.

3. Which one of the following statements is true regarding three address code?
   (i) At most three operands and at most two operators can be at the right side of the assignment sign while only one memory address can be there at left side in three address expression. (ii) It is also quite uncommon that operand names are numbered sequentially.
   (iii) Three address code is an intermediate code used by compilers to aid in the implementation of code-improving transformations.
   (iv) The operands will most likely be concrete memory addresses or processor registers.
   (A) (i) only
   (B) (ii) and (iii) only
   (C) (iii) only
   (D) (iii) and (iv) only
   (E) (ii), (iii) and (iv) only

4. Which of the following is not a representation of intermediate level code produced by analysis phase of compiler:
   (A) Syntax tree
   (B) Triples
   (C) Two address code
   (D) Polish notation
   (E) Quadruples
   (F) DAG

5. For a C program accessing $X[p][q][r][s]$, the following intermediate code is generated by a compiler. Assume that the size of an integer is 32 bits and the size of a character is 8 bits.
   $t_0 = p * 204800$
   $t_1 = r * 80$
   $t_2 = q * 3200$
   $t_3 = s * 4$
   $t_4 = t_1 + t_0$
   $t_4 = t_4 + t_2$
   $t_4 = t_4 + t_3$
   $t_5 = X[t_4]$
   Which one of the following statements about the source code for the C program is CORRECT?
   (A) X is declared as "int X[22] [128] [10] [40]".
   (B) X is declared as "int X[44] [32] [40] [40]".
   (C) X is declared as "int X[42] [128] [40] [10]".
   (D) X is declared as "int X[26] [64] [40] [20]".

6. Which one of the following is FALSE?
   (A) A basic block is a sequence of instructions where control enters the sequence at the beginning and may exit at any point.
   (B) Available expression analysis can be used for common subexpression elimination.
   (C) Live variable analysis can be used for dead code elimination.
   (D) $x = 4 * 5 \rightarrow x = 20$ is an example of common constant folding.

7. The least number of temporary variables required to create a three-address code in static single assignment form for the expression $q + r/3 + s - t * 5 + u * v$ is:
   (A) 6
   (B) 7
   (C) 8
   (D) 9

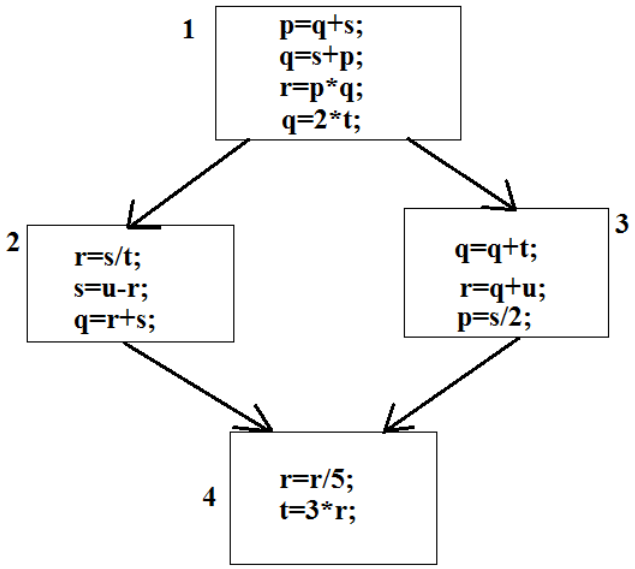
8. Which of the following is essential for converting an infix expression to the postfix form efficiently?
   (A) An operator stack
   (B) An operand stack
   (C) An operand stack and an operator stack
   (D) A parse tree

9. In the context of abstract-syntax-tree (AST) and control-flow-graph (CFG), which one of the following is TRUE?

(A) In both AST and CFG, let node N_2 be the successor of node N_1. In the input program, the code corresponding to N_2 is present after the code corresponding to $N1$.
(B) For any input program, neither AST nor CFG will contain a cycle
(C)The maximum number of successors of a node in an AST and a CFG depends on the input program
(D) Each node in AST and CFG corresponds to at most one statement in the input program

10. Some code optimizations are carried out on the intermediate code as:
(A) They enhance the portability of the compiler to the target processor.
(B) Program analysis is more accurate on intermediate code than on machine code.
(C) The information from dataflow analysis cannot otherwise be used for optimization.
(D) The information from the front end cannot otherwise be used for optimization.

11. Consider following control flow graph:



The variables which are live both at the statement in basic block 2 and at the statement in basic block 3 of the above control flow graph are:
(A) s, t, u
(B) r, s, t
(C) r, u

(D) t, u

The variables which are live both at the statement in basic block 2 and at the statement in basic block 3 of the above control flow graph are A. p, s, u B. r, s, u C. r, u D. q, v

12. In an optimizing compiler where the compiler performs some sort of optimization before generating the final code, an expression $y = 4x^7 + 2x^6 + 4x + 5 - 3$ is evaluated. What is the minimum number of arithmetic operations required to be executed to evaluate this expression once the target code is generated?
(A) 10
(B) 12
(C) 14
(D) 15
(E) 16

13. Consider following expression:
$a + a * (b - c) + (b - c) * d$
The number of nodes in its corresponding syntax tree is S and number of nodes in the corresponding Directed acyclic graph, DAG, is D. The value of $|D - S|$ is:
(A) 6
(B) 5
(C) 1
(D) 4
(E) 2

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 above basic block respectively are:
(A) 6 and 6
(B) 8 and 10
(C) 9 and 12
(D) 4 and 4

15. Translate the arithmetic expression $(a + b * c) + d + (a + b * c) - d + e$ into:
(1) syntax tree
(2) DAG (Directed Acyclic Graph) representation

(3) Three-Address Code representation

16. Translate the arithmetic expression $a + -(b + c)$ into:
(1) syntax tree
(2) quadruples
(3) triples
(4) indirect triples

17. The arithmetic expression corresponding to following three-address code given below, is given by:
$t_1 := -c$
$t_2 := b * t_1$
$t_3 := -c$
$t_4 := b * t_3$
$t_5 := t_2 + t_4$
$a := t_5$
(A) $a := b * c - b * -c$
(B) $a := (b - c) * (b - c)$
(C) $a := b * c - b * -c$
(D) $a := b * -c + b * -c$

18. Consider following statement:
$a := b * -c + c * -d$
Which of the following represents minimum number of three address instructions needed to express the above statement?
(A) 3
(B) 4
(C) 5
(D) 6

19. Consider following conditional statement:
if (x < 100) || x > 200 && x != y)
x = 0;
An SDD is designed and used to produce the three-address instructions for above conditional statement. Which of the following three address code represents above conditional expression?
(A)

```
        if x < 100 goto L2
           goto L3
L3:     if x > 200 goto L1
           goto L3
L4:     if x != y goto L2
           goto L1
```

```
L2:     x = 0
L1:
```

(B)

```
        if x < 100 goto L4
           goto L3
L3:     if x > 200 goto L2
           goto L1
L4:     if x != y goto L2
           goto L1
L2:     x = 0
L1:
```

(C)

```
        if x < 100 goto L2
           goto L3
L3:     if x > 200 goto L4
           goto L1
L4:     if x == y goto L2
           goto L1
L2:     x = 0
L1:
```

(D)

```
        if x < 100 goto L2
           goto L3
L3:     if x > 200 goto L4
           goto L1
L4:     if x != y goto L2
           goto L1
L2:     x = 0
L1:
```

20. Consider following loop instruction:

```
do
      i = i + 1;
while (a[i] < v);
```

Which of the following three address code represents above conditional expression?

(A)

```
L:    i = i + 1
      t1 = i
      t2 = a[t1]
      if t2 < v goto L
```

(B)
L: $i = t_1 + 1$
$t_1 = i$
$t_2 = a[t_1]$
if $t_2 < v$ goto L

(C)
L: $t1 = i + 1$
$i = t_1$
$t_2 = a[t_1]$
if $t_2 < v$ goto L

(D) None of the above

21. Consider the grammar rule $E \rightarrow E_1 - E_2$ for arithmetic expressions. The code generated is targeted to a CPU having a single user register. The subtraction operation requires the first operand to be in the register. If E_1 and E_2 do not have any common sub expression, in order to get the shortest possible code:
(A) E1 should be evaluated first
(B) E2 should be evaluated first
(C) Evaluation of E1 and E2 should necessarily be interleaved
(D) Order of evaluation of E1 and E2 is of no consequence

22. Consider following three address instructions:
p = a + b
q = p - c
p = q * d
p = e - p
q = p + q
The number of additional variables needed to convert this code into Static Single Assignment form is:
(A) 8
(B) 9
(C) 10
(D) 11
(E) 12

23. The program below uses six temporary variables .
a = 1
b = 10
c = 20

d = a + b
e = c + d
f = c + e
b = c + e
e = b + f
d = 5 + e
return d + f
Assuming that all operations take their operands from registers, what is the minimum number of registers needed to execute this program without spilling?
(A) 2
(B) 3
(C) 4
(D) 6

24. Consider following code fragment with simple if-else construct:
if (flag)
x = -1;
else
x = 1;
y = x * a
If $flag, x, y, a$ all are variables of type integer, the number of total variables needed to convert this code fragment into Static Single Assignment form is:
(A) 3
(B) 4
(C) 5
(D) 6
(E) 7

25. Consider evaluating the following expression tree on a machine with load-store architecture in which memory can be accessed only through load and store instructions. The variables and are initially stored in memory. The binary operators used in this expression tree can be evaluated by the machine only when operands are in registers. The instructions produce result only in a register. If no intermediate results can be stored in memory, what is the minimum number of registers needed to evaluate this expression?
o.7o.7
(A) 2
(B) 3
(C) 4
(D) 5
(E) 6

26. Consider the expression $(a-1)*(((c-5)*3/(b+d))+a*2))$. Let X be the minimum number of registers required by a n optimal code generation (without any register spill) algorithm for a load/store architecture, in which
(i) only load and store instructions can have memory operands a n d
(ii) arithmetic instructions can have only register or immediate operands.
The value of X is:
(A) 2
(B) 3
(C) 4
(D) 5
(E) 6