# Write Three Address Code for the following expression-all three types.

-(a x b) + (c + d) – (a + b + c + d)

(1) T1 = a x b

(2) T2 = uminus T1

(3) T3 = c + d

(4) T4 = T2 + T3

(5) T5 = a + b

(6) T6 = T3 + T5

(7) T7 = T4 – T6

# Write Three Address Code for the following expression-

If A < B then 1 else 0

Three Address Code for the given expression is-

(1) If (A < B) goto (4)

(2) T1 = 0

(3) goto (5)

(4) T1 = 1

(5)

# Write Three Address Code for the following expression-

If A < B and C < D then t = 1 else t = 0

 Three Address Code for the given expression is-

(1) If (A < B) goto (3)

(2) goto (4)

(3) If (C < D) goto (6)

(4) t = 0

(5) goto (7)

(6) t = 1

(7)

# Translate the following expression to quadruple, triple and indirect triple-

a + b x c / e ↑ f + b x c

Three Address Code for the given expression is-

T1 = e ↑ f

T2 = b x c

T3 = T2 / T1

T4 = b x a

T5 = a + T3

T6 = T5 + T4

## Quadruple Representation-

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Location** | **Op** | **Arg1** | **Arg2** | **Result** |
| (0) | ↑ | e | f | T1 |
| (1) | x | b | c | T2 |
| (2) | / | T2 | T1 | T3 |
| (3) | x | b | a | T4 |
| (4) | + | a | T3 | T5 |
| (5) | + | T5 | T4 | T6 |

## Triple Representation-

|  |  |  |  |
| --- | --- | --- | --- |
| **Location** | **Op** | **Arg1** | **Arg2** |
| (0) | ↑ | e | f |
| (1) | x | b | c |
| (2) | / | (1) | (0) |
| (3) | x | b | a |
| (4) | + | a | (2) |
| (5) | + | (4) | (3) |

## Indirect Triple Representation-

|  |  |
| --- | --- |
|  | **Statement** |
| 35 | (0) |
| 36 | (1) |
| 37 | (2) |
| 38 | (3) |
| 39 | (4) |
| 40 | (5) |

|  |  |  |  |
| --- | --- | --- | --- |
| **Location** | **Op** | **Arg1** | **Arg2** |
| (0) | ↑ | e | f |
| (1) | x | b | e |
| (2) | / | (1) | (0) |
| (3) | x | b | a |
| (4) | + | a | (2) |
| (5) | + | (4) | (3) |

# Translate the following expression to quadruple, triple and indirect triple-

a = b x – c + b x – c

Three Address Code for the given expression is-

T1 = uminus c

T2 = b x T1

T3 = uminus c

T4 = b x T3

T5 = T2 + T4

a = T5

Now, we write the required representations-

## Quadruple Representation-

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Location** | **Op** | **Arg1** | **Arg2** | **Result** |
| (1) | uminus | c |  | T1 |
| (2) | x | b | T1 | T2 |
| (3) | uminus | c |  | T3 |
| (4) | x | b | T3 | T4 |
| (5) | + | T2 | T4 | T5 |
| (6) | = | T5 |  | a |

## Triple Representation-

|  |  |  |  |
| --- | --- | --- | --- |
| **Location** | **Op** | **Arg1** | **Arg2** |
| (1) | uminus | c |  |
| (2) | x | b | (1) |
| (3) | uminus | c |  |
| (4) | x | b | (3) |
| (5) | + | (2) | (4) |
| (6) | = | a | (5) |

## Indirect Triple Representation-

|  |  |
| --- | --- |
|  | **Statement** |
| 35 | (1) |
| 36 | (2) |
| 37 | (3) |
| 38 | (4) |
| 39 | (5) |
| 40 | (6) |

|  |  |  |  |
| --- | --- | --- | --- |
| **Location** | **Op** | **Arg1** | **Arg2** |
| (1) | uminus | c |  |
| (2) | x | b | (1) |
| (3) | uminus | c |  |
| (4) | x | b | (3) |
| (5) | + | (2) | (4) |
| (6) | = | a | (5) |

# Translate the arithmetic expression

(a+b\*c)+d+(a+b+\*c)-d+e into

1. syntax tree
2. Postfix notation
3. Three address code.

## Syntax Tree

Diagram

Description automatically generated with low confidence

## Postfix notation

abc\*+d+ab+c\*++d-e+

## Three address code.

1. T1 = a + b \* c
2. T2 = T1 + d
3. T3 = T2 – d + e

# FIND out given expression is of L attribute or S attribute

1. X => ABC {B.P = X.P, B.P = A.P, B.P = C.P}

This is not an L-attributed SDT because B.P = X.P and B.P = A.P are allowed, but B.P = C.P doesn’t follow the rule of L-attributed SDT definition.

1. A -> XYZ {Y.S = A.S, Y.S = X.S, Y.S = Z.S}

is not an L-attributed grammar since Y.S = A.S and Y.S = X.S are allowed but Y.S = Z.S violates the L-attributed SDT definition as attributed is inheriting the value from its right sibling.

# ****Make DAG For Following Expression (SDT)****

1. a + a \* ( b – c ) + ( b - c ) \* d

Diagram

Description automatically generated

1. E→ T R

R→ addop T {print(addop)} R | ε

T→ num {print(num)}

parse tree for 9 – 5 + 2

**Chart, radar chart, line chart

Description automatically generated**

1. S → A1 A2 {A1.in = 1, A2.in = 2}

A → a {print(A.in)}

Diagram, schematic

Description automatically generated

1. E → E1 + T E.val := E1.val + T.val

E → E1 – T E.val := E1.val – T.val

E → T E.val := T.val

T → (E) T.val := E.val

T → num T.val := num.lexval

Chart, radar chart

Description automatically generated