CD lab assignment 9 20CS01035.

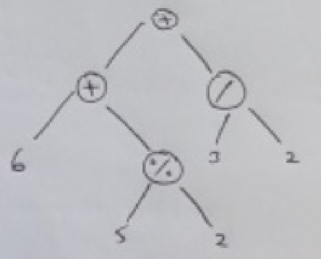
ques-1) AST.
ques-2) DAG

non-terminals ⇒ E, T, F, G.
terminals ⇒ +, -, *, /, (,).
Attributes → isOp, value, left, right.

G → E { G.isOp = E.isOp , G.value = E.value , G.left = E.left , G.Right = E.right }
E → E₁ + T { E.isOp = 1 , E.value = 0 , E.left = E₁ , E.right = T]
E → E₁ - T { E.isOp = 1 , E.value = 1 , E.left = E₁ , E.right = T }
E → T { T.isOp = T.isOp , E.value = T.value, E.left = T.left , E.right = T.right }
T → T₁ * F { T.isOp = 1 , T.value = 2 , T.left = T₁ , T.right = F }
T → T₁ / F { T.isOp = 1 , T.value = 3 , T.left = T₁ , T.right = F }
T → F { T.isOp = F.isOp , T.value = F.value , T.left = F.left , T.right = F.right }.
F → num { F.isOp = 0 , F.value = num.lexval }.
F → (E) { F.isOp = E.isOp , F.val = E.val , F.right = E.right , F.left = E.left }
E → E₁ % T { E.isOp = 1 , E.value = 4 , E.left = E₁ , E.right = T].

We are printing Preorder and inorder of the Abstract Syntax tree.

eg → 6 + 5 % 2 + 3/2



Preorder ⇒
+ + 6 % 5 2 / 3 2
inorder →
6 + 5 % 2 + 3 / 2.

For DAG we are using expression as another attribute and a unique id to each expression. and just check that the same node exists among previously made nodes or not.

Run → python3 script.py.