Name: Atandrit Chatterjee

Reg. No.: RA2011031010042

**Experiment-13** **Construction of DAG**

Aim:

To implement the construction of DAG using C/ C++.

Procedure:

1. Start the program.

2. Include all the header files.

3. Check for postfix expression and construct the in-order DAG representation.

4. Print the output.

5. Stop the program.

Code:

#include <iostream>

#include <string>

#include <unordered\_map>

using namespace std;

class DAG

{

public:

char label;

char data;

DAG \*left;

DAG \*right;

DAG(char x)

{

label = '\_';

data = x;

left = NULL;

right = NULL;

}

DAG(char lb, char x, DAG \*lt, DAG \*rt)

{

label = lb;

data = x;

left = lt;

right = rt;

}

};

int main()

{

int n;

n = 3;

string st[n];

st[0] = "A=x+y";

st[1] = "B=A\*z";

st[2] = "C=B/x";

unordered\_map<char, DAG \*> labelDAGNode;

for (int i = 0; i < 3; i++)

{

string stTemp = st[i];

for (int j = 0; j < 5; j++)

{

char tempLabel = stTemp[0];

char tempLeft = stTemp[2];

char tempData = stTemp[3];

char tempRight = stTemp[4];

DAG \*leftPtr;

DAG \*rightPtr;

if (labelDAGNode.count(tempLeft) == 0)

{

leftPtr = new DAG(tempLeft);

}

else

{

leftPtr = labelDAGNode[tempLeft];

}

if (labelDAGNode.count(tempRight) == 0)

{

rightPtr = new DAG(tempRight);

}

else

{

rightPtr = labelDAGNode[tempRight];

}

DAG \*nn = new DAG(tempLabel, tempData, leftPtr, rightPtr);

labelDAGNode.insert(make\_pair(tempLabel, nn));

}

}

cout << "Label ptr leftPtr rightPtr" << endl;

for (int i = 0; i < n; i++)

{

DAG \*x = labelDAGNode[st[i][0]];

cout << st[i][0] << " " << x->data << " ";

if (x->left->label == '\_')

cout << x->left->data;

else

cout << x->left->label;

cout << " ";

if (x->right->label == '\_')

cout << x->right->data;

else

cout << x->right->label;

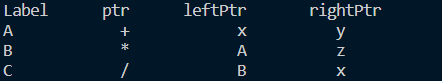
cout << endl;

}

return 0;

}

Output:



Result:

The construction of DAG was successful.