Experiment 11

SAKTHI RAM

RA2111026010346

**Write a program to traverse syntax trees and perform action arithmetic operations**

**CODE :-**

#include <stdio.h>

#include <stdlib.h>

typedef struct Node {

char\* value;

struct Node\* left;

struct Node\* right;

} Node;

Node\* createNode(char\* value) {

Node\* newNode = (Node\*)malloc(sizeof(Node));

newNode->value = value;

newNode->left = NULL;

newNode->right = NULL;

return newNode;

}

double evaluateSyntaxTree(Node\* root) {

if (root == NULL)

return 0.0;

if (root->value[0] == '+') {

return evaluateSyntaxTree(root->left) + evaluateSyntaxTree(root->right);

}

else if (root->value[0] == '-') {

return evaluateSyntaxTree(root->left) - evaluateSyntaxTree(root->right);

}

else if (root->value[0] == '\*') {

return evaluateSyntaxTree(root->left) \* evaluateSyntaxTree(root->right);

}

else if (root->value[0] == '/') {

double denominator = evaluateSyntaxTree(root->right);

if (denominator == 0) {

fprintf(stderr, "Error: Division by zero\n");

exit(1);

}

return evaluateSyntaxTree(root->left) / denominator;

}

else {

return atof(root->value);

}

}

int main() {

Node\* root = createNode("+");

root->left = createNode("\*");

root->left->left = createNode("3");

root->left->right = createNode("4");

root->right = createNode("+");

root->right->left = createNode("10");

root->right->right = createNode("2");

double result = evaluateSyntaxTree(root);

printf("Result: %.2f\n", result);

return 0;

}

