Ahsanullah University of Science and Technology

Department of Computer Science & Engineering

Assignment:05

Course No: CSE4130

Course Title: Formal Languages and Compiler Lab

Date of Submission : 09/08/23

Submitted to: Md. Aminur Rahman

Assistant Professor

&

Iffatun Nessa

Adjunct Faculty

Submitted By
Group : B2
Name : MD Fardin Jaman Aranyak
Id:190204110

Implement the following CFG in the way shown above.

A → aXd

X → bbX

X → bcX

X → ε

Answer :

```
#include<bits/stdc++.h>
using namespace std;

int i = 0, f, l;
string input;

int processX()
{
   while (i < l - 1)
   {
      if (input[i] == 'b' && (input[i + 1] == 'b' || input[i + 1] == 'c'))
         i = i + 2;
      else
         return 0;
   }
   return 1;
}

void processA()
{
   if (input[i] == 'a')
```

```
    {
        i++;
        if (l == 2 && input[l - 1] == 'd')
            f = 1;
        else
        {
            if (processX() && input[l - 1] == 'd')
                f = 1;
            else
                f = 0;
        }
    }
    else
        f = 0;
}

int main()
{
    cout << "Enter string: ";
    getline(cin, input);
    l = input.length();
    processA();
    if (f)
        cout << "The string is accepted";
    else
        cout << "The string is not accepted";
    return 0;
}
```

Implement the CFG shown above for generating simple arithmetic expressions.

Answer:

```
#include<bits/stdc++.h>
using namespace std;

bool isExpn(const string& inp, int& index);
bool isTerm(const string& inp, int& index);
bool isFactor(const string& inp, int& index);
bool isNUM(const string& inp, int& index);
bool isID(const string& inp, int& index);

bool isID(const string& inp, int& index)
{
   if (inp[index] >= 'a' && inp[index] <= 'z')
   {
      index++;
      return true;
   }
   return false;
}

bool isNUM(const string& inp, int& index)
{
   if (inp[index] >= '0' && inp[index] <= '9')
   {
      index++;
      return true;
   }
```

```
    return false;
}

bool isFactor(const string& inp, int& index)
{
    if (inp[index] == '(')
    {
        index++;
        if (isExpn(inp, index))
        {
            if (inp[index] == ')')
            {
                index++;
                return true;
            }
        }
        return false;
    }
    return isID(inp, index) || isNUM(inp, index);
}

bool isTerm(const string& inp, int& index)
{
    if (isFactor(inp, index))
    {
        while (index < inp.length() && (inp[index] == '*' || inp[index] == '/'))
        {
            index++;
```

```
        if (!isFactor(inp, index))
            return false;
        }
        return true;
    }
    return false;
}

bool isExpn(const string& inp, int& index)
{
    if (isTerm(inp, index))
    {
        while (index < inp.length() && (inp[index] == '+' || inp[index] == '-'))
        {
            index++;
            if (!isTerm(inp, index))
                return false;
        }
        return true;
    }
    return false;
}

int main()
{
    string inp;
    cout << "Enter arithmetic expression: ";
    getline(cin, inp);
```

```
    int index = 0;
    if (isExpn(inp, index) && index == inp.length())
        cout << "The arithmetic expression is accepted.";
    else
        cout << "The arithmetic expression is not accepted";

    return 0;
}
```