Agbo, Ana Marie, S.

BSCpE-4

335541

Valid Statement Assignment

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **State** | **Iden** | **=** | **Expression** | **;** | **Others** | **Simple Nos.** |
| 0 | 1 | 5 | 5 | 5 | 5 | 5 |
| 1 | 5 | 2 | 5 | 4 | 5 | 5 |
| 2 | 1 | 5 | 3 | 5 | 5 | 1 |
| 3 | 5 | 5 | 5 | 4 | 5 | 5 |
| 4 | 5 | 5 | 5 | 5 | 5 | 5 |
| 5 | 5 | 5 | 5 | 5 | 5 | 5 |

Q= q0, q1, q2, q3, q4, q5

Initial State = q0

Final State = q4

**;**

**Q5**

**Q0**

**Q4**

**Q3**

**Q1**

**Q2**

**Iden**

**Iden = Expression ;**

=, iden

Expression,

Others,

Simple Nos.

=, ;,

Others,

Iden,

Expression,

Others,

Simple Nos.

=, iden,

Expression,

Others,

Simple Nos.

=, ;,

Expression,

Others,

Simple Nos.

Iden, =, ;,

Expression,

Others,

Simple Nos.

using System;

using System.Collections.Generic;

namespace midterm

{

class Program

{

char []save1;

string []save2;

string []save3;

public int splitting(string state\_ment)

{

//converting string to char array

char []str = state\_ment.ToCharArray();

//to save the delimiter, make a copy

char []strdup = state\_ment.ToCharArray();

save1 = state\_ment.ToCharArray();

string[]split = state\_ment.Split(new char[] {'=',';'},StringSplitOptions.RemoveEmptyEntries);

try{

var str1 = new List <string>();

int from = 0;

int testme=0;

//hello=num;

//5

//

// Console.WriteLine("adding on the list");

foreach(string ch in split)

{

// Console.WriteLine("string {0}",ch);

str1.Add(ch);

// Console.WriteLine("inserting {0}",ch);

from = ch.Length+from;

// Console.WriteLine(from);

str1.Add(str[from+testme].ToString());

testme++;

}

//testing the List

//saving it to public variable

// Console.WriteLine("Transferring");

int cap = str1.Count;

save2 = new String[cap];

int count=0;

foreach(string gh in str1)

{

// Console.WriteLine(gh);

save2[count] = gh.ToString();

// Console.WriteLine(save2[count]);

count++;

}

}

catch{

// Console.WriteLine("Invalid");

}

return 1;

}

int check\_simpleno(string simplenum)

{

int state=0;

int [,]table = new int [,] {

{1,4,4},

{1,2,4},

{3,4,4},

{3,4,4},

{4,4,4}

};

int input=0;

int flag = 0;

//test simple number function

//checking each char if its digits

foreach (char n in simplenum)

{

// Console.WriteLine(n);

if(n=='.')

{

input = 1;

// Console.WriteLine("state\_1");

}

else if (char.IsDigit(n))

{

input=0;

// Console.WriteLine("state\_0");

}

else {

input=2;

// Console.WriteLine("state\_2");

}

state = table[state,input];

if(state==1)

{

flag=1;

}

else if (state==3)

{

flag =1;

}

else {

flag =0;

}

}

if(flag == 1)

{

// Console.WriteLine("simplenumber\_accepted");

return 1;

}

else

{

// Console.WriteLine("simplenumber\_notaccepted");

return 0;

}

}

int check\_identifier(string simpleidentifier)

{

int state =0;

int [,] table = new int [,] {

{1,1,2,2},

{1,1,1,2},

{2,2,2,2}

};

int input;

int flag=0;

// Console.WriteLine("identifier function");

foreach (char m in simpleidentifier)

{

if(m=='\_')

{

input=0;

// Console.WriteLine("identifier detects \_ ");

}

else if (char.IsLetter(m))

{

input = 1;

// Console.WriteLine("identifier detects letter");

}

else if (char.IsDigit(m))

{

input = 2;

// Console.WriteLine("identifier detects digit");

}

else {

input =3;

// Console.WriteLine("?");

}

state = table [state,input];

if (state==1)

{

flag =1;

}

else {

flag = 0;

}

}

// Console.WriteLine("identifing");

if(flag==1)

{

// Console.WriteLine("identifier");

return 1;

}

else {

// Console.WriteLine("Not identifier");

return 0;

}

}

int check\_expression(string expressme)

{

// Console.WriteLine("New string {0}",expressme);

int state=0;

int input;

int [,] table= new int [,] {

{1,4,1,4},

{4,2,4,4},

{3,4,3,4},

{4,2,4,4},

{4,4,4,4}

};

char a='+';

char b = '-';

char c = '\*';

char d = '/';

char e = '%';

string plus = a.ToString();

string minus = b.ToString();

string multi = c.ToString();

string divide = d.ToString();

string modul= e.ToString();

int l = splitting2(expressme);

int flag = 0;

foreach (string lastme in save3)

{

if(check\_identifier(lastme)==1)

{

input=0;

// Console.WriteLine("identifier");

}

else if (lastme == plus || lastme== minus || lastme == multi || lastme==divide || lastme==modul)

{

input=1;

// Console.WriteLine("operator");

}

else if (check\_simpleno(lastme)==1)

{

input=2;

// Console.WriteLine("simple number");

}

else {

input=3;

}

state=table[state,input];

if(state==3)

{

flag=1;

}

else {

flag=0;

}

}

if (flag == 1)

{

// Console.WriteLine("expressing");

return 1;

}

else {

// Console.WriteLine("not expressing");

return 0;

}

}

int splitting2(string state\_ment)

{

Console.WriteLine();

// Console.WriteLine("expression: {0} ",state\_ment);

char [] str = state\_ment.ToCharArray();

string[]split = state\_ment.Split(new char[] {'+','/','-','\*','%'},StringSplitOptions.RemoveEmptyEntries);

var str1 = new List <string>();

str1.Clear();

int from = 0;

int testme=0;

// Console.WriteLine("max capacity {0} ",str1.Count);

//hello=num;

//5

//

// Console.WriteLine("split array lenght  {0}",split.Length);

foreach(string ch in split)

{

// Console.WriteLine("string {0} length {1}",ch,ch.Length);

str1.Add(ch);

// Console.WriteLine("inserting {0}",ch);

if(from+ch.Length+testme < str.Length)

{

from = ch.Length+from;

// Console.WriteLine("{0} {1}",from,str[from+testme]);

str1.Add(str[from+testme].ToString());

testme++;

}

// Console.WriteLine("list count {0} from {1} testme {2}",str1.Count,from,testme);

}

// Console.WriteLine("data in list str1 {0}",str1.Count);

int cap = str1.Count;

save3 = new String[cap];

int count=0;

// Console.WriteLine("transferring list count= {0}",str1.Count);

foreach (string hh in str1)

{

// Console.WriteLine(hh);

save3[count] = hh.ToString();

// Console.WriteLine(save3[count]);

count++;

}

return 1;

}

public static void Main(string[] args)

{

Program a = new Program();

int [,]table = new int[,] {

{1,5,5,5,5,5},

{5,2,5,4,5,5},

{1,5,3,5,5,3},

{5,5,5,4,5,5},

{5,5,5,5,5,5}

};

string statement;

/\*getting the string statement \*/

statement = Console.ReadLine();

int h=0;

int state=0;

int input;

char eq='=';

char semi=';';

string equall = eq.ToString();

string semicol = semi.ToString();

int flag=0;

/\*splitting the statement \*/

try{

h = a.splitting(statement);

// Console.WriteLine(a.save2.Length);

// valid statement operation

foreach(string testme in a.save2)

{

try{

// Console.ReadKey(true);

if(a.check\_identifier(testme)==1)

{

input=0;

// Console.WriteLine("identifier");

} /\*

else if (a.check\_simpleno(testme)==1)

{

Console.WriteLine("simple number");

}\*/

else if (testme == equall)

{

input=1;

// Console.WriteLine("equal detected");

}

else  if (a.check\_expression(testme)==1)

{

input=2;

// Console.WriteLine("expression");

}

else if(testme == semicol)

{

input=3;

// Console.WriteLine("semi");

}

else if (a.check\_simpleno(testme)==1)

{

input=5;

// Console.WriteLine("simple number");

}

else {

input=4;

// Console.WriteLine("error detected");

}

state=table[state,input];

if(state==4)

{

flag =1;

}

}

catch(Exception)

{

// Console.WriteLine("error");

}

}

}

catch

{

}

if(flag==1)

{

Console.WriteLine("Valid Assignment Statement");

}

else {

Console.WriteLine("Invalid Assignment Statement");

}

// TODO: Implement Functionality Here

// Console.Write("Press any key to continue . . . ");

// Console.ReadKey(true);

}

}

}