**CECS 524 Unit 9 Assignment 1**

Name: Spuritha Mudireddy

Student ID: 030743269

**Code:**

import java.io.File;

import java.io.FileNotFoundException;

import java.util.\*;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class Expression

{

public static class Line\_Memory{

int lineNumber;

String line;

}

private static ArrayList<Line\_Memory> program=new ArrayList<Line\_Memory>() ; /\*the entire SIL program is in this array\*/

private static int curr\_line; /\*the current line that is executing\*/

private static boolean mEOL, mEOF;

public static HashMap<String,Integer> memory=new HashMap<String,Integer>();

public static Stack<Integer> st=new Stack<Integer>(); /\*Stack to store Integer values\*/

public static Stack<Integer> lln=new Stack<Integer>(); /\* Stack to store the last line number\*/

public static void main(String[] args) throws FileNotFoundException {

String in;

int k=0;

Scanner sc = new Scanner(new File("C://Users//mspur//IdeaProjects//unit 6//Input.txt"));

while (sc.hasNextLine()) {

in=sc.nextLine().toUpperCase();

Expression.Line\_Memory lm=new Expression.Line\_Memory();

lm.lineNumber=Integer.parseInt(in.substring(0, in.indexOf(' ')));

lm.line=in.substring(in.indexOf(' ')+1);

program.add(lm);

}

parseProgram();

}

public static void parseProgram()

{

int end\_line=program.size();

int i=0,low,high;

while(i!=end\_line)

{

curr\_line=program.get(i).lineNumber;

int x=parse(program.get(i).line,curr\_line,i);

if(x==curr\_line)

{

i++;

}

else if(x<curr\_line)

{

low=0;

high=i;

i=search(x, low,high);

curr\_line=program.get(i).lineNumber;

}

else

{

low=i;

high=program.size()-1;

i=search(x, low,high);

curr\_line=program.get(i).lineNumber;

}

}

}

public static void takeInput(String [] vars)

{

Scanner sc = new Scanner(System.in);

String p=sc.nextLine();

String [] nums=p.split(" ");

int [] values=new int[nums.length];

for(int i=0;i<nums.length;i++)

values[i]=Integer.parseInt(nums[i]);

if(vars.length!=values.length||values.length==0)

System.err.println("Line n missing input value");

else {

for (int i = 0; i < vars.length; i++) {

memory.put(vars[i], values[i]);

}

}

}

public static int parse(String in,int ln,int index)

{

in.trim();

if(in.equals("END"))

{

System.exit(0);

}

if(in.equals("RET"))

{

return lln.pop();

}

String ins = in.substring(0, in.indexOf(' '));

String str = in.substring(in.indexOf(' ') + 1);

if(ins.equals("LET"))

{

String [] vals=str.split("=");

memory.put(vals[0].trim(),expr(vals[1].trim()));

}

if(ins.equals("PUSH"))

{

st.push(expr(str));

}

if(ins.equals("POP"))

{

memory.put(str,st.pop());

}

if(ins.equals("GOSUB"))

{

int x=search(Integer.parseInt(str),0,program.size());

if(x==-1)

System.out.println("Line Number not found: "+program.get(x).lineNumber);

lln.push(program.get(index+1).lineNumber);

return Integer.parseInt(str);

}

if(ins.equals("INPUT"))

{

takeInput(str.split(","));

}

if(ins.equals("GOTO"))

{

return Integer.parseInt(str);

}

if(ins.equals("IF"))

{

String condition = str.split("THEN")[0];

String action= in.split("THEN")[1];

String op="";

if(condition.contains("="))

op="=";

else if(condition.contains("<"))

op="<";

else if(condition.contains(">"))

op=">";

else if(condition.contains("!"))

op="!";

int x= calculate(expr(condition.split(op)[0]),expr(condition.split(op)[1]),op);

if(x==1)

{

int p= parse(action.trim(),curr\_line,index);

return p;

}

return curr\_line;

}

if(ins.equals("INTEGER"))

{

String [] variables=str.split(",");

for(int i=0;i<variables.length;i++)

{

memory.put(variables[i].trim(),0);

}

}

if(ins.equals("PRINTLN"))

{

Pattern p = Pattern.compile("\"([^\"]\*)\"");

String res="";

Matcher m = p.matcher(str);

while (m.find()) {

res=m.group(1);

}

if (memory.containsKey(str)) {

System.out.println(expr(str));

}

else if(str.length()-2==res.length())

{

System.out.println(res);

}

else

{

String rem = str.replace(res,"");

String exp=rem.split(",")[1];

System.out.print(res);

System.out.println(expr(exp));

}

}

if(ins.equals("PRINT"))

{

Pattern p = Pattern.compile("\"([^\"]\*)\"");

String res="";

Matcher m = p.matcher(str);

while (m.find()) {

res=m.group(1);

}

if (memory.containsKey(str))

System.out.print(expr(str));

if(str.length()-2==res.length())

{

System.out.print(res);

}

else

{

String rem = str.replace(res,"");

String exp=rem.split(",")[1];

System.out.print(expr(exp));

}

}

return curr\_line;

}

public static int search(int target,int low,int high)

{

int mid=low + ((high - low) / 2);

while(low<=high)

{

mid=low + ((high - low) / 2);

if(program.get(mid).lineNumber==target)

return mid;

else if(program.get(mid).lineNumber<target)

low=mid+1;

else

high=mid-1;

}

return -1;

}

public static int expr(String s)

{

Stack<Integer> v=new Stack<Integer>();

Stack<String> op=new Stack<String>();

int i=0;

while(i<s.length())

{

String p=toString(s.charAt(i));

if(isNumeric(p))

{

String num="";

int in=i;

while(in<s.length()&&isNumeric(s.substring(in,in+1)))

{

num+=s.charAt(in);

in++;

}

i=in-1;

v.push(Integer.parseInt(num));

}

else if(memory.containsKey(p))

{

v.push(memory.get(p));

}

else if(s.charAt(i)=='(')

{

String brac="";

int x=i+1;

while(x<s.length()&&s.charAt(x)!=')')

{

brac+=s.charAt(x);

x++;

}

i=x;

if(isNumeric(brac))

v.push(Integer.parseInt(brac));

else

v.push(expr(brac));

}

else if(isOperand(p))

{

while(op.size()>0&&precedence(op.peek(),p))

{

String c=op.pop();

int op1=v.pop();

int op2=v.pop();

v.push(calculate(op1,op2,c));

}

op.push(p);

}

i++;

}

while(op.size()>0)

{

String c=op.pop();

int op1=v.pop();

int op2=v.pop();

v.push(calculate(op1,op2,c));

}

return v.pop();

}

public static int atom(String s)

{

if(isNumeric(s))

return Integer.parseInt(s);

else if(memory.containsKey(s))

return memory.get(s);

else

return expr(s);

}

public static String toString(char ch)

{

return Character.toString(ch);

}

public static boolean isNumeric(String s)

{

try

{

Integer.parseInt(s);

return true;

}

catch( Exception e )

{

return false;

}

}

public static boolean isOperand(String s)

{

if(s.contains("+")||s.contains("-")||s.contains("\*")||s.contains("/"))

return true;

return false;

}

public static boolean containsOperand(String s)

{

if(s.contains("+")||s.contains("-")||s.contains("\*")||s.contains("/"))

return true;

if(s.contains("THEN"))

if(s.contains("<")||s.contains(">")||s.contains("=")||s.contains("!"))

return true;

return false;

}

public static int calculate(int op2,int op1,String op)

{

if(op.equals("+"))

return op1+op2;

else if(op.equals("-"))

return op1-op2;

else if(op.equals("\*"))

return op1\*op2;

else if(op.equals("/"))

return op1/op2;

else if(op.equals("<"))

return (op2<op1)?1:0;

else if(op.equals(">"))

return (op2>op1)?1:0;

else if(op.equals("="))

return (op1==op2)?1:0;

else

return (op1!=op2)?1:0;

}

public static boolean precedence(String op1,String op2)

{

HashMap<String,Integer> hm=new HashMap<String,Integer>();

hm.put("+",1);

hm.put("-",1);

hm.put("\*",2);

hm.put("/",2);

if(hm.get(op1)-hm.get(op2)>0)

return true;

else

return false;

}

}

**Input 1**

10 println "line 10"  
20 gosub 200  
30 println "line 30"  
40 end  
200 println "in the sub"  
210 gosub 300  
220 println "back from 300"  
230 ret  
300 println "start of 300 sub"  
310 ret  
320 end

**Output 1**

**Text

Description automatically generated**

**Input 2**

10 PRINTLN "This program finds the sum of 1 to n where n is entered by the user"

20 INTEGER N, S, I

30 PRINT "Enter n:"

40 INPUT n

50 GOSUB 100

60 PRINT "The sum of 1 to n is "

65 PRINTLN S

70 PRINT "Enter 0 to quit, 1 to do another sum:"

80 INPUT S

90 IF S = 1 THEN GOTO 30

95 END

100 PRINTLN "Finding the sum of 1 to ", n

105 LET S = 0

110 LET I = 1

120 IF I>N THEN GOTO 160

130 LET S = S + I

140 LET I = I + 1

150 GOTO 120

160 RET

170 END

**Output 2:**

**Text

Description automatically generated**

**INPUT 3:**

5 integer x, y

10 println "pop and push test"

15 integer b

18 let b = 23

20 push 50\*3 + b

30 integer a

40 pop a

50 println "a=" , a

60 push a

70 pop b

80 println "b=",b

90 push 5

100 push 7

110 gosub 200

111 push 2

112 push 4

113 gosub 200

120 end

200 println "in sub"

210 pop y

220 pop x

230 println "x+y=", x+y

240 ret

250 end

**Output 3**

**Text

Description automatically generated**