//Liang Fang

//Emplid: 23134567

//MAC286

//Take home exam

Question 1:

//Take the program you have submitted for evaluating a mathematical expression. Edit to print all the intermediate operations in the order they are executed along with the final result. Also print the postfix notation of the mathematical expression.

**public** **class** IntStack{

**int** size;

**int**[] arr;

**int** top;

**public** IntStack(**int** size){

**this**.size = size;

arr = **new** **int**[size];

top = -1;

}

**public** **boolean** isEmpty(){

**return** (top == -1)? **true**:**false**;

}

**public** **boolean** isFull(){

**return** (top == size-1)? **true**:**false**;

}

**public** **void** push(**int** value){

**if**(!isFull())

arr[++top] = value;

**else**

**return**;

}

**public** **int** pop(){

**return** (!isEmpty())? arr[top--]:-1;

}

**public** **int** peek(){

**return** (!isEmpty())? arr[top]:-1;

}

}

**public** **class** CharStack{

**int** size;

**char**[] arr;

**int** top;

**public** CharStack(**int** size){

**this**.size = size;

arr = **new** **char**[size];

top = -1;

}

**public** **boolean** isEmpty(){

**return** (top == -1)? **true**:**false**;

}

**public** **boolean** isFull(){

**return** (top == size-1)? **true**:**false**;

}

**public** **void** push(**char** ch){

**if**(!isFull())

arr[++top] = ch;

**else**

**return**;

}

**public** **char** pop(){

**return** (!isEmpty())? arr[top--]:**null**;

}

**public** **char** peek(){

**return** (!isEmpty())? arr[top]:**null**;

}

}

**public** **class** InfixToPostfix{

CharStack aStack;

String input;

String output = "";

**int** size;

**public** InfixToPostfix(String s){

input = s;

size = input.length();

aStack = **new** CharStack(size);

}

**public** String convert(){

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

**char** ch = input.charAt(i);

**switch**(ch){

**case** '+':

**case** '-':

gotOper(ch,1);

**break**;

**case** '\*':

**case** '/':

gotOper(ch,2);

**break**;

**case** '(': //if it's open paren, then push to stack

aStack.push(ch);

**break**;

**case** ')':

gotParen(); //call function and go to see if the next pop is (

**break**;

**default**:

output +=ch; //if it's an operand, then write to output

}

}

**while**(!aStack.isEmpty()){

output += aStack.pop(); //pop remaining write to output

}

**return** output;

}

**public** **void** gotOper(**char** ch, **int** prece){

**while**(!aStack.isEmpty()){

**char** top = aStack.pop();

**if**(top == '('){

aStack.push(top);// if it's opening paren, then push back to stack

**break**;

}

**int** newPrece; //precedent of new operator

**if**(top == '+' || top == '-')

newPrece = 1;

**else**

newPrece = 2;

**if**(newPrece < prece){

aStack.push(top); //if new precedence is less than old precedence, save back

**break**;

}

**else**

output += top; //if new precedence is larger than old one, then write to output

}

aStack.push(ch);//push the remaining

}

**public** **void** gotParen(){

**while**(!aStack.isEmpty()){

**char** chx = aStack.pop();

**if**(chx == '(') //find ( , matching with )

**break**;

**else**

output += chx; //the pop is a operand, then write to output

}

}

}

**public** **class** Calculation{

String input;

IntStack aStack;//create a instance of integer stack

**int** size;

**public** Calculation(String input){

**this**.input = input;

size = input.length();

aStack = **new** IntStack(size);

}

**public** **int** calc(){

**char** ch;

**int** num1, num2, result;

**for**(**int** i =0; i<size; i++){

ch = input.charAt(i);

**if**(ch >= '0' && ch <= '9'){

aStack.push((**int**)(ch-'0'));

}

**else**{

num2 = aStack.pop();

num1 = aStack.pop();

**switch**(ch){

**case** '+':

result = num1 + num2;

System.***out***.println(num1 + " + "+ num2 + " = "+result);

**break**;

**case** '-':

result = num1 - num2;

System.***out***.println(num1 + " - "+ num2 + " = "+result);

**break**;

**case** '\*':

result = num1 \* num2;

System.***out***.println(num1 + " \* "+ num2 + " = "+result);

**break**;

**case** '/':

result = num1 / num2;

System.***out***.println(num1 + " / "+ num2 + " = "+result);

**break**;

**default**:

result = 0;

**break**;

}

aStack.push(result); //push the result back to stack

}

}

result = aStack.pop();//pop up the final calculated result and return it

**return** result;

}

}

**import** java.util.\*;

**public** **class** CalcMain{

**public** **static** **void** main(String[] args){

String input, output;

**int** calcOutput;

System.***out***.print("Enter an expression Infix: ");

Scanner scan = **new** Scanner(System.***in***);

input = scan.nextLine();

InfixToPostfix exp = **new** InfixToPostfix(input);

output = exp.convert();

System.***out***.println("Postfix is " + output);

Calculation newExp = **new** Calculation(output);

calcOutput = newExp.calc();

System.***out***.println("The calulated result is: " + calcOutput);

}

}



