1. Nick Wright CSC285 Problem 5
2. Java Code

/\*The following program will return the result of an expression when passed as a character array.  
\* It will use method parseExpression to calculate the value of the expression, putting the expression through stacks.  
\* The program contains a generic class with pop, push, and peek methods for stacks.  
\* There are 4 classes total, StackParcer3, GenericManagerStacks<T>, Operatobj, and objOf4.\*/  
  
import java.util.\*;  
import java.lang.\*;  
  
public class StackParcer3 {  
 public static void main(String[] args)throws Exception {  
 //defining values a,b,c as objects  
 objOf4 A = new objOf4(1,2,3,4);  
 objOf4 B = new objOf4(6,6,8,8);  
 objOf4 C = new objOf4(1,2,2,1);  
 //creating an object to hold each int 0-9  
 objOf4 zero = new objOf4(0,0,0,0);  
 objOf4 one = new objOf4(1,1,1,1);  
 objOf4 two = new objOf4(2,2,2,2);  
 objOf4 three = new objOf4(3,3,3,3);  
 objOf4 four = new objOf4(4,4,4,4);  
 objOf4 five = new objOf4(5,5,5,5);  
 objOf4 six = new objOf4(6,6,6,6);  
 objOf4 seven = new objOf4(7,7,7,7);  
 objOf4 eight = new objOf4(8,8,8,8);  
 objOf4 nine = new objOf4(9,9,9,9);  
  
 objOf4[] ivalue3={A,B,C,zero,one,two,three,four,five,six,seven,eight,nine};  
  
 //ivalue1 contains the values of the integers from the example code  
 int [] ivalue1={7,6,-2,3,1,0,1,2,3,4,5,6,7,8,9};//A=7,B=6,C=-2,D=3,E=1  
 //ivalue2 contains the values of the integers from the assignment  
 int [] ivalue2={8,12,2,3,15,4,0,1,2,3,4,5,6,7,8,9};//A=8,B=12,C=2,D=3,E=15,F=4  
 // A\*3\*(B-C)/((A-C)+D) || this is the original expression that was used in the example code  
 char [] express={'A','\*','3','\*','(','B','-','C',')','/','(','(','A','-','C',')','+','D',')','#'};  
 //express1 = A@(2\*(A-C\*D))+(9\*B/(2\*C+1)-B\*3)+E%(A-F)  
 char[] express1={'A','@','(','2','\*','(','A','-','C','\*','D',')',')','+','(','9','\*','B','/','(','2','\*','C','+','1',')','-','B','\*','3',')','+','E','%','(','A','-','F',')','#'};  
 //express2 = B\*(3@(A-D)%(B-C@D))+4@D\*2  
 char[] express2={'B','\*','(','3','@','(','A','-','D',')','%','(','B','-','C','@','D',')',')','+','4','@','D','\*','2','#'};  
  
 //\*\*\*\*\*\*\*Part 2 of assignment\*\*\*\*\*\*\*\*\*\*  
  
  
 //using method parseExpression to calculate the expressions  
 //parseExpression(express,ivalue1); commenting out the example expression to help reduce the size of the output on the submission document  
// parseExpression(express1,ivalue2);  
// parseExpression(express2,ivalue2);  
  
 //the expression from part 2 of the assignment  
 char[] part2Express={'2','\*','A','-','3','\*','(','(','B','-','2','\*','C',')','/','(','A','+','3',')','-','B','\*','3',')','#'};  
 *parseExpression*(part2Express,ivalue3);//parsing the expression  
  
 }//end of main  
  
 /\*created the method parseExpression to parse through the expression character array and return the value.  
 \* this was made an expression so that the while loop did not have to be repeated within the main method every time we wanted to evaluate an expression  
 \* the method will take a character array of an expression and an array of integer values\*/  
 public static void parseExpression(char[] express1, objOf4[] values1){//method to calculate expressions represented as character arrays  
 // now create a stack for the integer operands  
 GenericManagerStacks<objOf4> opnd=new GenericManagerStacks<objOf4>();  
 // now create a stack for the Operators  
 GenericManagerStacks<Opertobj> oper=new GenericManagerStacks<Opertobj>();  
 System.*out*.println("pushing Operator #with priority -100");  
 Opertobj pnode1=new Opertobj('#',-100); //creating an operator objects and pushing it onto the stack  
 oper.pushnode(pnode1);  
 int oprior;  
 objOf4 exvalue;//exvalue will hold the value that we want to return after calculation of expression is complete  
 int i, num;  
 objOf4 ivalu;  
 char[] vart2={'A','B','C','0','1','2','3','4','5','6','7','8','9'}; //the variable array for part 2 of assignment  
// char [] vart={'A','B','C','D','E','F','0','1','2','3','4','5','6','7','8','9'}; //the variable array with both letters and numbers  
 char [] opert={'\*','/','+','-',')','(','#'};//this is the set of operators that will be evaluated in expressions  
 // now create the evaluation priority for the symbols. The higher the priority, the higher the number in the table  
 int [] intvalp={2,2,1,1,99,-99,-100};//the two operators added (@ and %) have priorities of 3 and 2 respectively.  
 //WE MUST INITIALIZE the OPERATOR STACK so the first Operator can be pushed on. I must put an end of operation on the operator stack.  
 i=0;  
 while(express1[i]!='#') {  
 System.*out*.println("parsing"+express1[i]);  
 if(((express1[i]>='0')&&(express1[i]<='9'))||((express1[i]>='A')&&(express1[i]<='Z')))  
 // Check to see if this character is a variable or an operator.  
 {// we have a variable or a constant  
 System.*out*.println("this is an operand"+express1[i]);  
 // find the character in the vart table that corresponds with the value  
 ivalu=*findval*(express1[i],vart2,values1,12); //originally was ivalu=findval(express1[i],vart,values1,15);  
 if(ivalu.topLeft==-99)System.*out*.println("no value in table for"+express1[i]);  
 // now that we have the value we need to place it on the operand stack  
 System.*out*.println("were pushing it on the operand stack"+ivalu);  
 opnd.pushnode(ivalu);  
 }//end of variable stack  
 else {//we are an operator  
 System.*out*.println("this is an operator"+express1[i]);  
 if(express1[i]=='(') {//this is a left parenthesis, push it on the stack  
 // System.out.println("pushing on operator stack"+express[i]);  
 //Create node to push on stack  
 Opertobj pnodeo=new Opertobj(express1[i],-99);  
 oper.pushnode(pnodeo);  
 }else if(express1[i]==')') {// this is a right parenthesis, we must begin to pop operands and pereators  
 //until we find the a left parenthesis (  
 while((oper.peeknode()).operator!='(') {//must pop and evaluate the stuff on operand and operator stack  
 *popevalandpush*(oper,opnd);  
 }  
 // now pop the ( node  
 oper.popnode();  
 }//end of this is a right parenthesis  
 else {//this is not either ( or ) is is another operator  
 oprior=*findval2*(express1[i],opert,intvalp,5);  
 System.*out*.println("peeking at top of stack"+(oper.peeknode()).priority);  
 //\*\*\*\*\*\*\*\*\*\*oprior MUST BE STRICTLY GREATER THAN BEFORE WE CAN PUT IT ON THE STACK\*\*\*\*\*\*\*\*  
 while(oprior<=(oper.peeknode()).priority)*popevalandpush*(oper,opnd);  
 //now push this operator on the stack.  
 System.*out*.println("pushing Operator"+express1[i]+"with priority"+oprior);  
 Opertobj pnodeo=new Opertobj(express1[i],oprior);  
 oper.pushnode(pnodeo);  
 }//this is the end of this is not () operator  
 }//end of on operator stack  
 i++;  
 }//end of while express loop  
 //we have found the # in the evaluation now we must evaluate the operator stack  
 while((oper.peeknode()).operator!='#') {//we are finishing up operator stack  
 *popevalandpush*(oper,opnd);  
 }// end of finishing up operator stack  
 //we're done, get value of opnd stack and print  
 exvalue=opnd.popnode();  
 System.*out*.println ("the value for this expression is "+exvalue+"\n");//prints out the expression value  
 }  
  
 //IntEval method will look at the operator and calculate the result of two operands dependent on the operator  
 public static objOf4 IntEval(objOf4 obj1, char oper, objOf4 obj2){  
 objOf4 result = new objOf4(-99,-99,-99,-99);  
 switch(oper){  
 case '+':  
 result.setTopLeft(obj1.getTopLeft()+obj2.getTopLeft());  
 result.setTopRight(obj1.getTopRight()+obj2.getTopRight());  
 result.setBottomLeft(obj1.getBottomLeft()+obj2.getBottomLeft());  
 result.setBottomRight(obj1.getBottomRight()+obj2.getBottomRight());  
 System.*out*.println("\*\*\*Evaluation: "+obj1+" "+oper+" "+obj2+" \*\*\*Result: "+result);  
 return result;  
 case '-':  
 result.setTopLeft(obj1.getTopLeft()-obj2.getTopLeft());  
 result.setTopRight(obj1.getTopRight()-obj2.getTopRight());  
 result.setBottomLeft(obj1.getBottomLeft()-obj2.getBottomLeft());  
 result.setBottomRight(obj1.getBottomRight()-obj2.getBottomRight());  
 System.*out*.println("\*\*\*Evaluation: "+obj1+" "+oper+" "+obj2+" \*\*\*Result: "+result);  
 return result;  
 case '\*':  
 result.setTopLeft(obj1.getTopLeft()\*obj2.getTopLeft());  
 result.setTopRight(obj1.getTopRight()\*obj2.getTopRight());  
 result.setBottomLeft(obj1.getBottomLeft()\*obj2.getBottomLeft());  
 result.setBottomRight(obj1.getBottomRight()\*obj2.getBottomRight());  
 System.*out*.println("\*\*\*Evaluation: "+obj1+" "+oper+" "+obj2+" \*\*\*Result: "+result);  
 return result;  
 case '/':  
 if(obj2.topLeft!=0&&obj2.topRight!=0&&obj2.bottomLeft!=0&&obj2.bottomRight!=0){  
 result.setTopLeft(obj1.getTopLeft()/obj2.getTopLeft());  
 result.setTopRight(obj1.getTopRight()/obj2.getTopRight());  
 result.setBottomLeft(obj1.getBottomLeft()/obj2.getBottomLeft());  
 result.setBottomRight(obj1.getBottomRight()/obj2.getBottomRight());  
 System.*out*.println("\*\*\*Evaluation: "+obj1+" "+oper+" "+obj2+" \*\*\*Result: "+result);  
 return result;  
 }else{  
 System.*out*.println("attempted to divide by zero error");  
 return result;  
 }  
 default:  
 System.*out*.println("bad operator "+oper);  
 return result;  
 }//end of switch statement  
 }//end of intEval  
  
 //findval will be used if we have an operand  
 public static objOf4 findval(char x, char [] vtab, objOf4[] valtb, int last) {  
 int i;  
 objOf4 vreturn = new objOf4(-99,-99,-99,-99);  
 // this finds the character x in the value table vtab and returns the  
 //correspond interger value table from valtb  
 for(i=0;i<=last; i++)  
 if(vtab[i]==x){  
 vreturn=valtb[i];  
 //vreturn=valtb[i];  
 }  
 System.*out*.println("found this char"+x+"value is"+vreturn);  
 return vreturn;  
 }//end of findval;  
  
 //findval2 will be used if we have an operator  
 public static int findval2(char x, char [] vtab, int [] valtb, int last) {  
 int i, vreturn=-99;  
 // this finds the character x in the value table vtab and returns the  
 //correspond interger value table from valtb  
 for(i=0;i<=last; i++)  
 if(vtab[i]==x)vreturn=valtb[i];  
 System.*out*.println("found this char"+x+"value is"+vreturn);  
 return vreturn;  
 }//end of findval;  
  
 public static void popevalandpush(GenericManagerStacks<Opertobj> x, GenericManagerStacks<objOf4> y) {//this is the start of pop and push  
 objOf4 a,b,c;  
 char operandx;  
 operandx=(x.popnode()).Getopert();  
 a=y.popnode();  
 b=y.popnode();  
 System.*out*.println("in popeval"+b+operandx+a);  
 c=*IntEval*(b,operandx,a);  
 //now push the value back on the stack for integers  
 y.pushnode(c);  
 return;  
 }//This is the end of popevalandpush  
}// this is the end of Stackparcer3 class  
  
class GenericManagerStacks<T>{//generic manager class for stacks  
 protected ArrayList<T> mystack;  
 protected int number;  
  
 public GenericManagerStacks() {// this is the generic constructor  
 number=0;//mcount is the next available value in array myarray  
 mystack=new ArrayList<T>(100);//creates an initial arraylist of 100  
 }  
  
 public int getnumber(){return number;}  
 public int pushnode(T x) {  
 System.*out*.println("in pushnode"+number+"x is"+x);  
 //this pushes a node on the stack. It will always add to the front(top) of the stack  
 mystack.add(number,x);  
 number++;  
 System.*out*.println("leaving pushnode");  
 return number;  
 }// end of pushnode  
  
 public T popnode() {//this function returns the first node in the list  
 T nodeval;//this is the value in the node to be popped  
 // find the node at the head of the list  
 nodeval=mystack.get(number-1);  
 //now pop the node by taking it off the list and moving head  
 mystack.remove(number-1);  
 number--;  
 //now return the value of this node.  
 return nodeval;  
 }// this is the end of popnode  
  
 public T peeknode() {//this function returns the contents of the top of the stack. It does not  
 //pop the node, just allows the user to look (peek) at the contents of the  
 //first node on the stack.  
 T nodeval; //this is the value to be peeked  
 nodeval=mystack.get(number-1);  
 return nodeval;  
 }//this is the end of peeknode  
  
 boolean stackempty(){if(number==0)return true;  
 else return false;}  
}//end of GenericManager class  
  
class Opertobj {// this is an operator class it will hold a character operator and it's stack priority  
 protected char operator;  
 protected int priority;  
  
 public Opertobj(char opert, int pri) {//this is the constructor for the operator object  
 operator=opert;  
 priority=pri;  
 }  
 public int Getprior(){return priority;};  
 public char Getopert(){return operator;};  
}//this is the end of the operator class  
  
class objOf4{ //this is the class that will be used to create each "array"  
 protected int topLeft;  
 protected int topRight;  
 protected int bottomLeft;  
 protected int bottomRight;  
  
 //constructor method for objOf4  
 public objOf4(int topLeft, int topRight, int bottomLeft, int bottomRight){  
 this.topLeft=topLeft;  
 this.topRight=topRight;  
 this.bottomLeft=bottomLeft;  
 this.bottomRight=bottomRight;  
 }  
  
 //getters for each of the integers  
 public int getTopLeft() {return topLeft;}  
 public int getTopRight() {return topRight;}  
 public int getBottomLeft() {return bottomLeft;}  
 public int getBottomRight() {return bottomRight;}  
  
 //setters for each of the integers  
 public void setTopLeft(int topLeft) {this.topLeft = topLeft;}  
 public void setTopRight(int topRight) {this.topRight = topRight;}  
 public void setBottomLeft(int bottomLeft) {this.bottomLeft = bottomLeft;}  
 public void setBottomRight(int bottomRight) {this.bottomRight = bottomRight;}  
  
 //toString override  
 @Override  
 public String toString() {  
 return "objOf4{" +  
 "topLeft=" + topLeft +  
 ", topRight=" + topRight +  
 ", bottomLeft=" + bottomLeft +  
 ", bottomRight=" + bottomRight +  
 '}';  
 }  
}

1. Input Files
   1. No input files used
2. Output Files

pushing Operator #with priority -100

in pushnode0x isOpertobj@1e80bfe8

leaving pushnode

parsing2

this is an operand2

found this char2value isobjOf4{topLeft=2, topRight=2, bottomLeft=2, bottomRight=2}

were pushing it on the operand stackobjOf4{topLeft=2, topRight=2, bottomLeft=2, bottomRight=2}

in pushnode0x isobjOf4{topLeft=2, topRight=2, bottomLeft=2, bottomRight=2}

leaving pushnode

parsing\*

this is an operator\*

found this char\*value is2

peeking at top of stack-100

pushing Operator\*with priority2

in pushnode1x isOpertobj@5387f9e0

leaving pushnode

parsingA

this is an operandA

found this charAvalue isobjOf4{topLeft=1, topRight=2, bottomLeft=3, bottomRight=4}

were pushing it on the operand stackobjOf4{topLeft=1, topRight=2, bottomLeft=3, bottomRight=4}

in pushnode1x isobjOf4{topLeft=1, topRight=2, bottomLeft=3, bottomRight=4}

leaving pushnode

parsing-

this is an operator-

found this char-value is1

peeking at top of stack2

in popevalobjOf4{topLeft=2, topRight=2, bottomLeft=2, bottomRight=2}\*objOf4{topLeft=1, topRight=2, bottomLeft=3, bottomRight=4}

\*\*\*Evaluation: objOf4{topLeft=2, topRight=2, bottomLeft=2, bottomRight=2} \* objOf4{topLeft=1, topRight=2, bottomLeft=3, bottomRight=4} \*\*\*Result: objOf4{topLeft=2, topRight=4, bottomLeft=6, bottomRight=8}

in pushnode0x isobjOf4{topLeft=2, topRight=4, bottomLeft=6, bottomRight=8}

leaving pushnode

pushing Operator-with priority1

in pushnode1x isOpertobj@51521cc1

leaving pushnode

parsing3

this is an operand3

found this char3value isobjOf4{topLeft=3, topRight=3, bottomLeft=3, bottomRight=3}

were pushing it on the operand stackobjOf4{topLeft=3, topRight=3, bottomLeft=3, bottomRight=3}

in pushnode1x isobjOf4{topLeft=3, topRight=3, bottomLeft=3, bottomRight=3}

leaving pushnode

parsing\*

this is an operator\*

found this char\*value is2

peeking at top of stack1

pushing Operator\*with priority2

in pushnode2x isOpertobj@1b4fb997

leaving pushnode

parsing(

this is an operator(

in pushnode3x isOpertobj@deb6432

leaving pushnode

parsing(

this is an operator(

in pushnode4x isOpertobj@28ba21f3

leaving pushnode

parsingB

this is an operandB

found this charBvalue isobjOf4{topLeft=6, topRight=6, bottomLeft=8, bottomRight=8}

were pushing it on the operand stackobjOf4{topLeft=6, topRight=6, bottomLeft=8, bottomRight=8}

in pushnode2x isobjOf4{topLeft=6, topRight=6, bottomLeft=8, bottomRight=8}

leaving pushnode

parsing-

this is an operator-

found this char-value is1

peeking at top of stack-99

pushing Operator-with priority1

in pushnode5x isOpertobj@694f9431

leaving pushnode

parsing2

this is an operand2

found this char2value isobjOf4{topLeft=2, topRight=2, bottomLeft=2, bottomRight=2}

were pushing it on the operand stackobjOf4{topLeft=2, topRight=2, bottomLeft=2, bottomRight=2}

in pushnode3x isobjOf4{topLeft=2, topRight=2, bottomLeft=2, bottomRight=2}

leaving pushnode

parsing\*

this is an operator\*

found this char\*value is2

peeking at top of stack1

pushing Operator\*with priority2

in pushnode6x isOpertobj@f2a0b8e

leaving pushnode

parsingC

this is an operandC

found this charCvalue isobjOf4{topLeft=1, topRight=2, bottomLeft=2, bottomRight=1}

were pushing it on the operand stackobjOf4{topLeft=1, topRight=2, bottomLeft=2, bottomRight=1}

in pushnode4x isobjOf4{topLeft=1, topRight=2, bottomLeft=2, bottomRight=1}

leaving pushnode

parsing)

this is an operator)

in popevalobjOf4{topLeft=2, topRight=2, bottomLeft=2, bottomRight=2}\*objOf4{topLeft=1, topRight=2, bottomLeft=2, bottomRight=1}

\*\*\*Evaluation: objOf4{topLeft=2, topRight=2, bottomLeft=2, bottomRight=2} \* objOf4{topLeft=1, topRight=2, bottomLeft=2, bottomRight=1} \*\*\*Result: objOf4{topLeft=2, topRight=4, bottomLeft=4, bottomRight=2}

in pushnode3x isobjOf4{topLeft=2, topRight=4, bottomLeft=4, bottomRight=2}

leaving pushnode

in popevalobjOf4{topLeft=6, topRight=6, bottomLeft=8, bottomRight=8}-objOf4{topLeft=2, topRight=4, bottomLeft=4, bottomRight=2}

\*\*\*Evaluation: objOf4{topLeft=6, topRight=6, bottomLeft=8, bottomRight=8} - objOf4{topLeft=2, topRight=4, bottomLeft=4, bottomRight=2} \*\*\*Result: objOf4{topLeft=4, topRight=2, bottomLeft=4, bottomRight=6}

in pushnode2x isobjOf4{topLeft=4, topRight=2, bottomLeft=4, bottomRight=6}

leaving pushnode

parsing/

this is an operator/

found this char/value is2

peeking at top of stack-99

pushing Operator/with priority2

in pushnode4x isOpertobj@593634ad

leaving pushnode

parsing(

this is an operator(

in pushnode5x isOpertobj@20fa23c1

leaving pushnode

parsingA

this is an operandA

found this charAvalue isobjOf4{topLeft=1, topRight=2, bottomLeft=3, bottomRight=4}

were pushing it on the operand stackobjOf4{topLeft=1, topRight=2, bottomLeft=3, bottomRight=4}

in pushnode3x isobjOf4{topLeft=1, topRight=2, bottomLeft=3, bottomRight=4}

leaving pushnode

parsing+

this is an operator+

found this char+value is1

peeking at top of stack-99

pushing Operator+with priority1

in pushnode6x isOpertobj@3581c5f3

leaving pushnode

parsing3

this is an operand3

found this char3value isobjOf4{topLeft=3, topRight=3, bottomLeft=3, bottomRight=3}

were pushing it on the operand stackobjOf4{topLeft=3, topRight=3, bottomLeft=3, bottomRight=3}

in pushnode4x isobjOf4{topLeft=3, topRight=3, bottomLeft=3, bottomRight=3}

leaving pushnode

parsing)

this is an operator)

in popevalobjOf4{topLeft=1, topRight=2, bottomLeft=3, bottomRight=4}+objOf4{topLeft=3, topRight=3, bottomLeft=3, bottomRight=3}

\*\*\*Evaluation: objOf4{topLeft=1, topRight=2, bottomLeft=3, bottomRight=4} + objOf4{topLeft=3, topRight=3, bottomLeft=3, bottomRight=3} \*\*\*Result: objOf4{topLeft=4, topRight=5, bottomLeft=6, bottomRight=7}

in pushnode3x isobjOf4{topLeft=4, topRight=5, bottomLeft=6, bottomRight=7}

leaving pushnode

parsing-

this is an operator-

found this char-value is1

peeking at top of stack2

in popevalobjOf4{topLeft=4, topRight=2, bottomLeft=4, bottomRight=6}/objOf4{topLeft=4, topRight=5, bottomLeft=6, bottomRight=7}

\*\*\*Evaluation: objOf4{topLeft=4, topRight=2, bottomLeft=4, bottomRight=6} / objOf4{topLeft=4, topRight=5, bottomLeft=6, bottomRight=7} \*\*\*Result: objOf4{topLeft=1, topRight=0, bottomLeft=0, bottomRight=0}

in pushnode2x isobjOf4{topLeft=1, topRight=0, bottomLeft=0, bottomRight=0}

leaving pushnode

pushing Operator-with priority1

in pushnode4x isOpertobj@6aa8ceb6

leaving pushnode

parsingB

this is an operandB

found this charBvalue isobjOf4{topLeft=6, topRight=6, bottomLeft=8, bottomRight=8}

were pushing it on the operand stackobjOf4{topLeft=6, topRight=6, bottomLeft=8, bottomRight=8}

in pushnode3x isobjOf4{topLeft=6, topRight=6, bottomLeft=8, bottomRight=8}

leaving pushnode

parsing\*

this is an operator\*

found this char\*value is2

peeking at top of stack1

pushing Operator\*with priority2

in pushnode5x isOpertobj@2530c12

leaving pushnode

parsing3

this is an operand3

found this char3value isobjOf4{topLeft=3, topRight=3, bottomLeft=3, bottomRight=3}

were pushing it on the operand stackobjOf4{topLeft=3, topRight=3, bottomLeft=3, bottomRight=3}

in pushnode4x isobjOf4{topLeft=3, topRight=3, bottomLeft=3, bottomRight=3}

leaving pushnode

parsing)

this is an operator)

in popevalobjOf4{topLeft=6, topRight=6, bottomLeft=8, bottomRight=8}\*objOf4{topLeft=3, topRight=3, bottomLeft=3, bottomRight=3}

\*\*\*Evaluation: objOf4{topLeft=6, topRight=6, bottomLeft=8, bottomRight=8} \* objOf4{topLeft=3, topRight=3, bottomLeft=3, bottomRight=3} \*\*\*Result: objOf4{topLeft=18, topRight=18, bottomLeft=24, bottomRight=24}

in pushnode3x isobjOf4{topLeft=18, topRight=18, bottomLeft=24, bottomRight=24}

leaving pushnode

in popevalobjOf4{topLeft=1, topRight=0, bottomLeft=0, bottomRight=0}-objOf4{topLeft=18, topRight=18, bottomLeft=24, bottomRight=24}

\*\*\*Evaluation: objOf4{topLeft=1, topRight=0, bottomLeft=0, bottomRight=0} - objOf4{topLeft=18, topRight=18, bottomLeft=24, bottomRight=24} \*\*\*Result: objOf4{topLeft=-17, topRight=-18, bottomLeft=-24, bottomRight=-24}

in pushnode2x isobjOf4{topLeft=-17, topRight=-18, bottomLeft=-24, bottomRight=-24}

leaving pushnode

in popevalobjOf4{topLeft=3, topRight=3, bottomLeft=3, bottomRight=3}\*objOf4{topLeft=-17, topRight=-18, bottomLeft=-24, bottomRight=-24}

\*\*\*Evaluation: objOf4{topLeft=3, topRight=3, bottomLeft=3, bottomRight=3} \* objOf4{topLeft=-17, topRight=-18, bottomLeft=-24, bottomRight=-24} \*\*\*Result: objOf4{topLeft=-51, topRight=-54, bottomLeft=-72, bottomRight=-72}

in pushnode1x isobjOf4{topLeft=-51, topRight=-54, bottomLeft=-72, bottomRight=-72}

leaving pushnode

in popevalobjOf4{topLeft=2, topRight=4, bottomLeft=6, bottomRight=8}-objOf4{topLeft=-51, topRight=-54, bottomLeft=-72, bottomRight=-72}

\*\*\*Evaluation: objOf4{topLeft=2, topRight=4, bottomLeft=6, bottomRight=8} - objOf4{topLeft=-51, topRight=-54, bottomLeft=-72, bottomRight=-72} \*\*\*Result: objOf4{topLeft=53, topRight=58, bottomLeft=78, bottomRight=80}

in pushnode0x isobjOf4{topLeft=53, topRight=58, bottomLeft=78, bottomRight=80}

leaving pushnode

the value for this expression is objOf4{topLeft=53, topRight=58, bottomLeft=78, bottomRight=80}

Process finished with exit code 0

1. Class Documentation

public **class StackParcer3**

**StackParcer3 Data**

The following data is located within the main method:

objOf4 A = new objOf4(1,2,3,4);

objOf4 B = new objOf4(6,6,8,8);

objOf4 C = new objOf4(1,2,2,1);

objOf4 zero = new objOf4(0,0,0,0);

objOf4 one = new objOf4(1,1,1,1);

objOf4 two = new objOf4(2,2,2,2);

objOf4 three = new objOf4(3,3,3,3);

objOf4 four = new objOf4(4,4,4,4);

objOf4 five = new objOf4(5,5,5,5);

objOf4 six = new objOf4(6,6,6,6);

objOf4 seven = new objOf4(7,7,7,7);

objOf4 eight = new objOf4(8,8,8,8);

objOf4 nine = new objOf4(9,9,9,9);

objOf4[] ivalue3={A,B,C,zero,one,two,three,four,five,six,seven,eight,nine};

int [] ivalue1={7,6,-2,3,1,0,1,2,3,4,5,6,7,8,9};

int [] ivalue2={8,12,2,3,15,4,0,1,2,3,4,5,6,7,8,9};

char [] express={'A','\*','3','\*','(','B','-','C',')','/','(','(','A','-','C',')','+','D',')','#'};

char[] express1={'A','@','(','2','\*','(','A','-','C','\*','D',')',')','+','(','9','\*','B','/','(','2','\*','C','+','1',')','-','B','\*','3',')','+','E','%','(','A','-','F',')','#'};

char[] express2={'B','\*','(','3','@','(','A','-','D',')','%','(','B','-','C','@','D',')',')','+','4','@','D','\*','2','#'};

char[] part2Express={'2','\*','A','-','3','\*','(','(','B','-','2','\*','C',')','/','(','A','+','3',')','-','B','\*','3',')','#'};

**StackParcer3 Functions**

public static void main(String[] args) throws exception (main method)

public static void parseExpression(char[] express1, objOf4[] values1)

public static int IntEval(objOf4 oper1, char oper, objOf4 oper2)

public static objOf4 findval(char x, char[] vtab, objOf4[] valtb, int last)

public static int findval2(char x, char[] vtab, int[] valtb, int last)

Public static void popevalandpush(GenericManagerStacks<Operatobj> x, GenericManagerStacks<objOf4> y)

**class GenericManagerStacks**<T>

**GenericManagerStacks Data**

protected ArrayList<T> mystack;

protected int number;

**GenericManagerStacks Functions**

public GenericManagerStacks()

public int getnumber()

public int pushnode(T x)

public T popnode

public T peeknode

boolean stackempty()

**class Operatobj**

**Operatobj Data**

protected char operator;

protected int priority

**Operatobj Functions**

public Operatobj(char opert, int pri)

public int Getprior()

public char Getopert()

**class objOf4**

**objOf4 Data**

protected int topLeft;

protected int topRight;

protected int bottomLeft;

protected int bottomRight;

**objOf4 Functions**

public objOf4(int topLeft, int topRight, int bottomLeft, int bottomRight)

public int getTopLeft()

public int getTopRight()

public int getBottomLeft()

public int getBottomRight()

public void setTopLeft(int topLeft)

public void setTopRight(int topRight)

public void setBottomLeft(int bottomLeft)

public void setBottomRight(int bottomRight)

public String toString()