UNIVERSIDAD NACIONAL DE COLOMBIA

Estructuras de Datos

Sesión 6
Stack Applications

Yoan Pinzón

Table of Content Session 6

Stack Application

Parentheses Matching

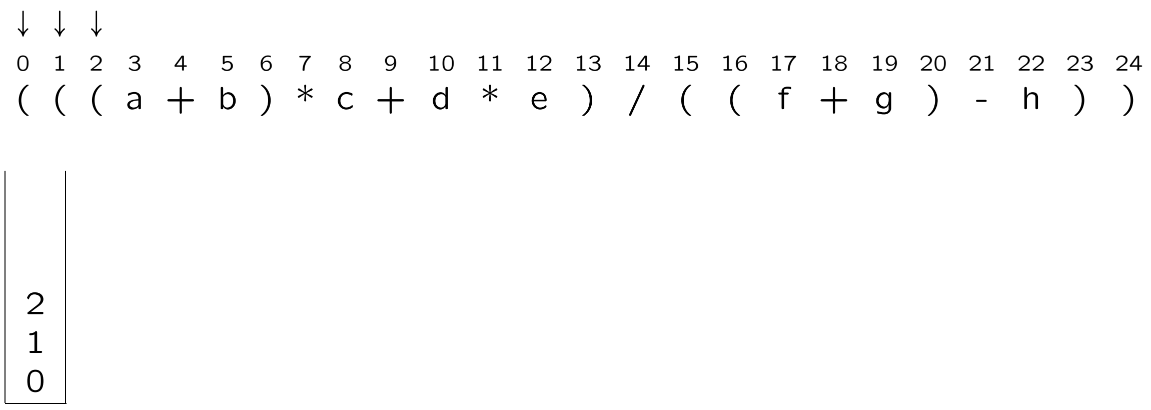
How do we match parentheses in an expression?

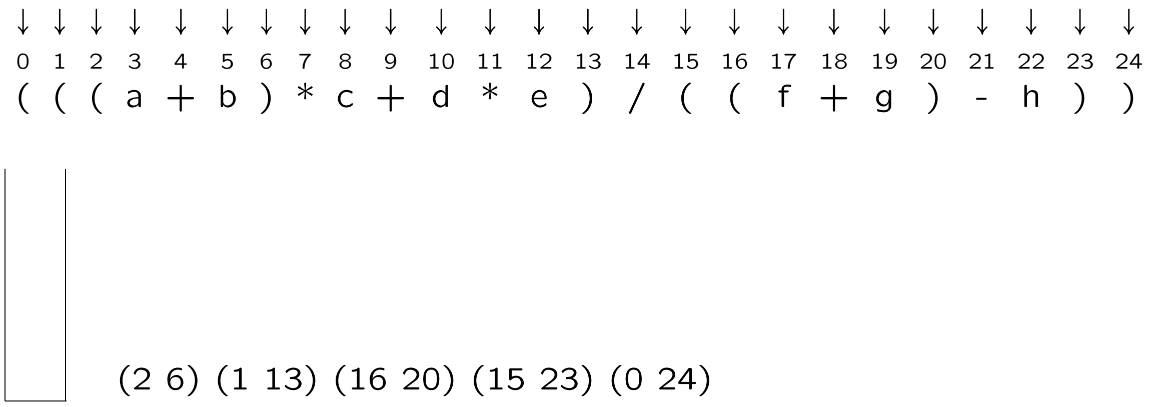
```
(((a+b)*c+d*e)/((f+g)-h+i))

(a*(a+b))/(b+d)
```

Parentheses Matching

- scan expression from left to right
- when a left parenthesis is encountered, add its position to the stack
- when a right parenthesis is encountered, remove matching position from stack





```
(2 6) (1 13) (16 20) (15 23) (0 24)
```

File `ParenthesesMatching.java`

```java
package unal.applications;
import unal.datastructures.*;
import java.util.*;
public class ParenthesisMatching
{
   public static void printMatchedPairs( String expr )
   {
      ArrayStack<Integer> s = new ArrayStack<>( );
      for( int i = 0; i < expr.length( ); i++ )
         if( expr.charAt( i ) == '(' )
            s.push( i );
         else if( expr.charAt( i ) == ')' )
            try
            { // remove location of matching '( ' from stack
               System.out.println( s.pop( ) + "␣␣" + i );
            }
            catch ( Exception e )
            { // stack was empty, no match exists
               System.out.println( "No␣match␣for␣right␣parenthesis␣↙
                  ↪ at␣" + i );
            }
      // remaining '( ' in stack are unmatched
      while( !s.isEmpty( ) )
         System.out.println( "No␣match␣for␣left␣parenthesis␣at␣" + 
            s.pop( ) );
   }

   /** test program */
   public static void main ( String[] args )
   {
      Scanner s = new Scanner( System.in );

      // input the expression
      System.out.println( "Type␣an␣expression␣with␣no␣spaces" );
      String expression = s.nextLine( );

      // output the pairs of matched parentheses
      System.out.println( "The␣pairs␣of␣matching␣parentheses␣in" );
      System.out.println( expression );
      System.out.println( "are␣(␣indexing␣begins␣at␣0␣)" );
      printMatchedPairs( expression );
   }
}
```

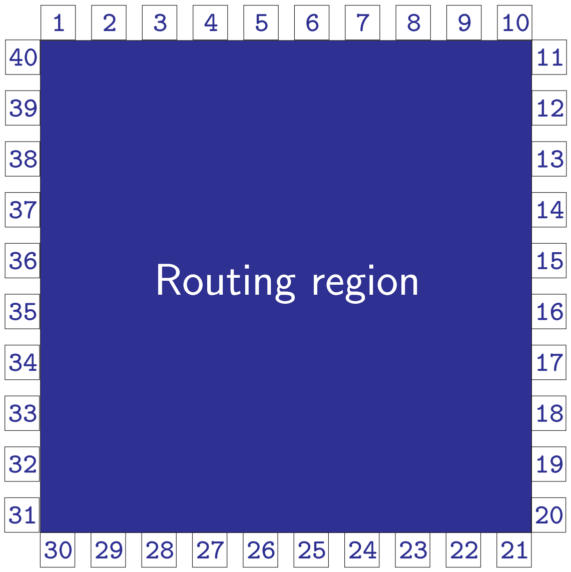
Compiling `ParenthesisMatching.java`

```
C:\2016699\code> javac unal\applications\ParenthesisMatching.java
C:\2016699\code> java unal.applications.ParenthesisMatching
Type an expression with no spaces
(((a+b)*c+d*e)/((f+g)-h))
The pairs of matching parentheses in
(((a+b)*c+d*e)/((f+g)-h))
are (indexing begins at 0)
2 6
1 13
16 20
15 23
0 24
```

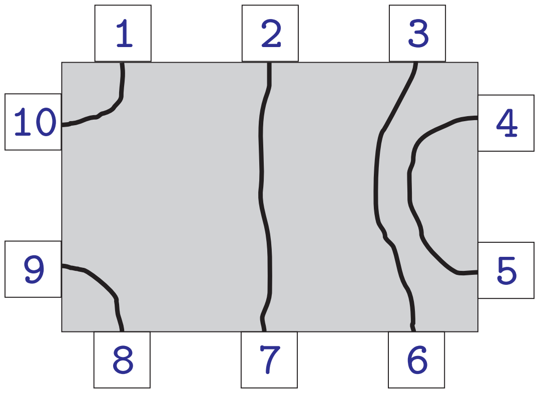
Stack Application

Switchbox Routing

The switchbox routing problem arises in the fabrication of computer chips, where certain components need to be connected to other components.

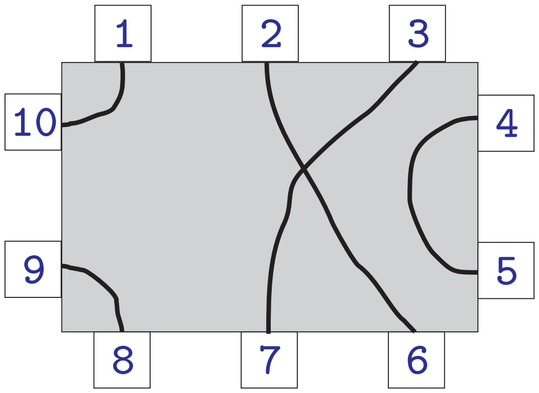


$$\begin{array}{lcccccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ \text{Net}=\{ & 1, & 2, & 3, & 4, & 4, & 3, & 2, & 5, & 5, & 1 \quad \} \end{array}$$



Routable!

$$\text{Net}=\{\begin{array}{cccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ 1, & 2, & 3, & 4, & 4, & 2, & 3, & 5, & 5, & 1 \end{array}\}$$



Not Routable!

File SwitchBox.java

```java
package unal.applications;

import unal.datastructures.*;
import java.util.*;

public class SwitchBox
{
   /** determine whether the switch box is routable
    * @param net array of pin to net assignments */
   public static boolean checkBox ( int[] net )
   {
      ArrayStack<Integer> s = new ArrayStack<>( );
      for( int i = 0; i < net.length; i++ )
         if( !s.isEmpty( ) )
            // check with top net
            if( net[ i ] == net[ s.peek( ) ] )
               // net[ i ] is routable, delete from stack
               s.pop( );
            else s.push( i );
         else s.push( i );
      // any unrouted nets left?
      if( s.isEmpty( ) )
      {   // no nets remain
         System.out.println( "Switch␣box␣is␣routable" );
         return true;
      }

      System.out.println( "Switch␣box␣is␣not␣routable" );

      return false;
   }

   /** test program */
   public static void main ( String[] args )
   {
      // define the input stream to be the standard input stream
      Scanner s = new Scanner( System.in );

      // input the number of pins and their net assignment
      System.out.println( "Type␣number␣of␣pins␣in␣switch␣box" );
      int n = s.nextInt( );
```

```
      // create net assignment array
      int[] net = new int[ n ];

      // input the net assignments
      System.out.println( "Type␣net␣numbers␣for␣pins␣1␣through␣" + ↙
         ↳ n );
      for( int i = 0; i < n; i++ )
         net[ i ] = s.nextInt( );

      // see if the switch box is routable
      checkBox( net );
   }
}
```

Compiling `SwitchBox.java`

```
C:\2016699\code> javac unal\applications\SwitchBox.java ↵
C:\2016699\code> java unal.applications.SwitchBox ↵
Type number of pins
20
Type net numbers for pins 1 through 20
1 2 3 4 4 5 6 6 7 8 9 9 10 10 8 7 5 3 2 1
Switch box is routable
```