Fundamentals of Data Structure

Mahesh Shirole

VJTI, Mumbai-19

Slides are prepared from

1. Data Structures and Algorithms in Java, 6th edition, by M. T. Goodrich, R. Tamassia, and M. H. Goldwasser, Wiley, 2014

2.Data Structures and Algorithms in Java, by Robert Lafore, Second Edition, Sams Publishing

Applications of Tree Traversals

- Table of Contents
- Computing Disk Space
- Parenthetic Representations of a Tree
- Using Inorder Traversal for Tree Drawing
- Euler Tours

A Table of Contents-Applications of Tree Traversals

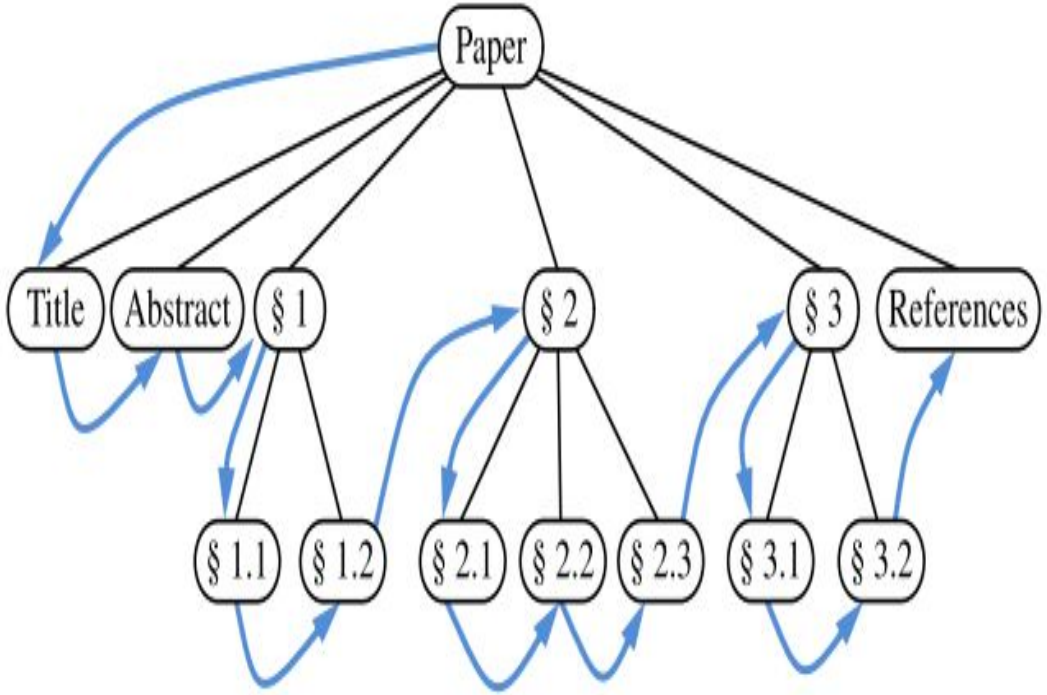
- A tree can represent the hierarchical structure of a document as we have discussed in earlier class
- A table of contents for the document can be produced using a preorder traversal of the tree

```
Paper
Title
Abstract
§1
§1.1
§1.2
§2
§2.1
...
```

(a)

```
Paper
  Title
  Abstract
  §1
    §1.1
    §1.2
  §2
    §2.1
    ...
```

(b)



A table of contents

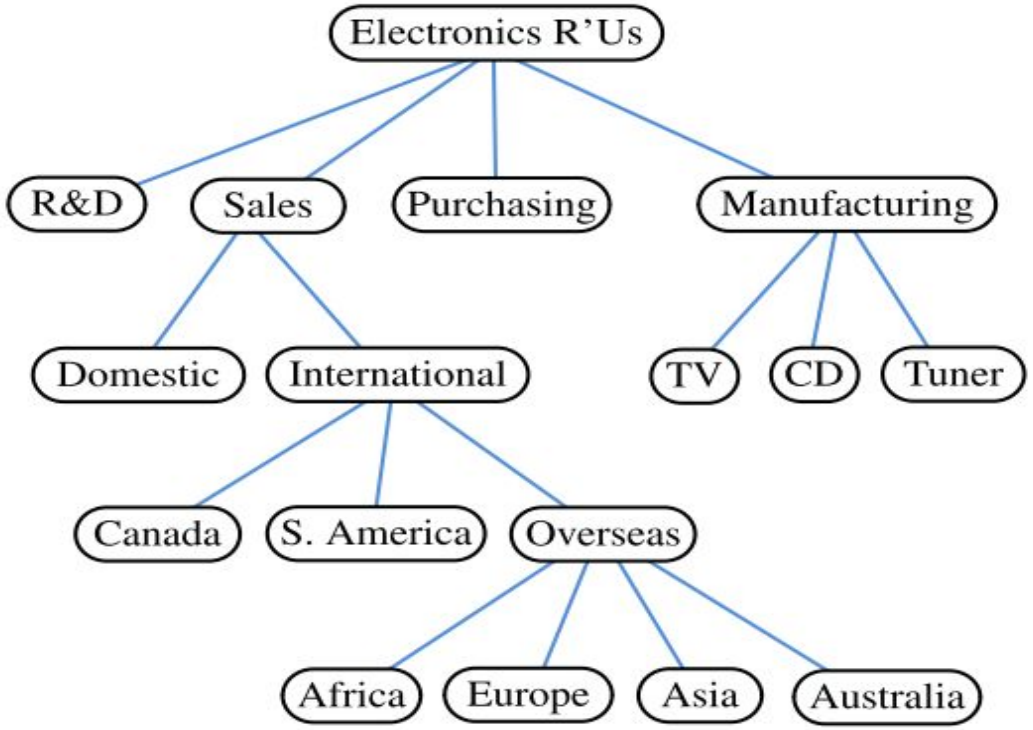
- To produce the presentation of Figure (b), we indent each element with a number of spaces equal to twice the element's depth in the tree

```
/** Prints preorder representation of subtree of T rooted at p having depth d. */
public static <E> void printPreorderIndent(Tree<E> T, Position<E> p, int d) {
  System.out.println(spaces(2*d) + p.getElement( ));      // indent based on d
  for (Position<E> c : T.children(p))
    printPreorderIndent(T, c, d+1);                       // child depth is d+1
}
```

Home Assignemnt - A table of contents

- Write a program using tree to display tree in following manner
- Add the numbering and indentation with each depth

```
Electronics R'Us
  1 R&D
  2 Sales
    2.1 Domestic
    2.2 International
      2.2.1 Canada
      2.2.2 S. America
```



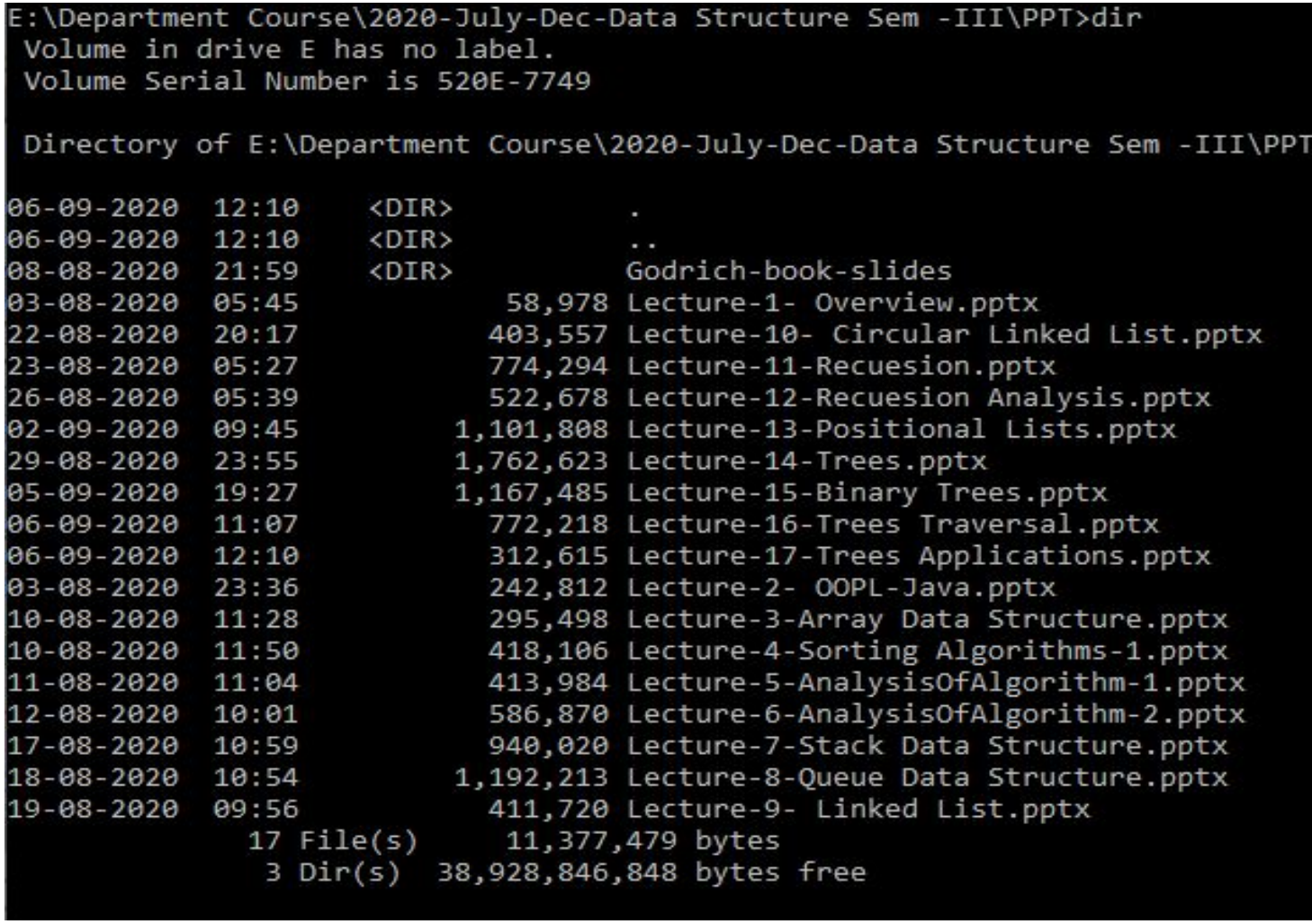
Computing Disk Space

- Considere the use of a tree as a model for a file-system structure, with internal positions representing directories and leaves representing files
- The recursive computation of disk space is similar to a postorder traversal
- For disk space computation, a mechanism for children to return information to the parent as part of the traversal process is required

```
/** Returns total disk space for subtree of T rooted at p. */
public static int diskSpace(Tree<Integer> T, Position<Integer> p) {
  int subtotal = p.getElement( );          // we assume element represents space usage
  for (Position<Integer> c : T.children(p))
    subtotal += diskSpace(T, c);
  return subtotal;
}
```

Home Assignemnt - Computing Disk Space

- Consider the directory shown in the figure
- Create node representing file information
  - name
  - type (0-file, 1-directory)
  - size
  - date
- Write a program to display diectory contents and its total disk space as shown in figure



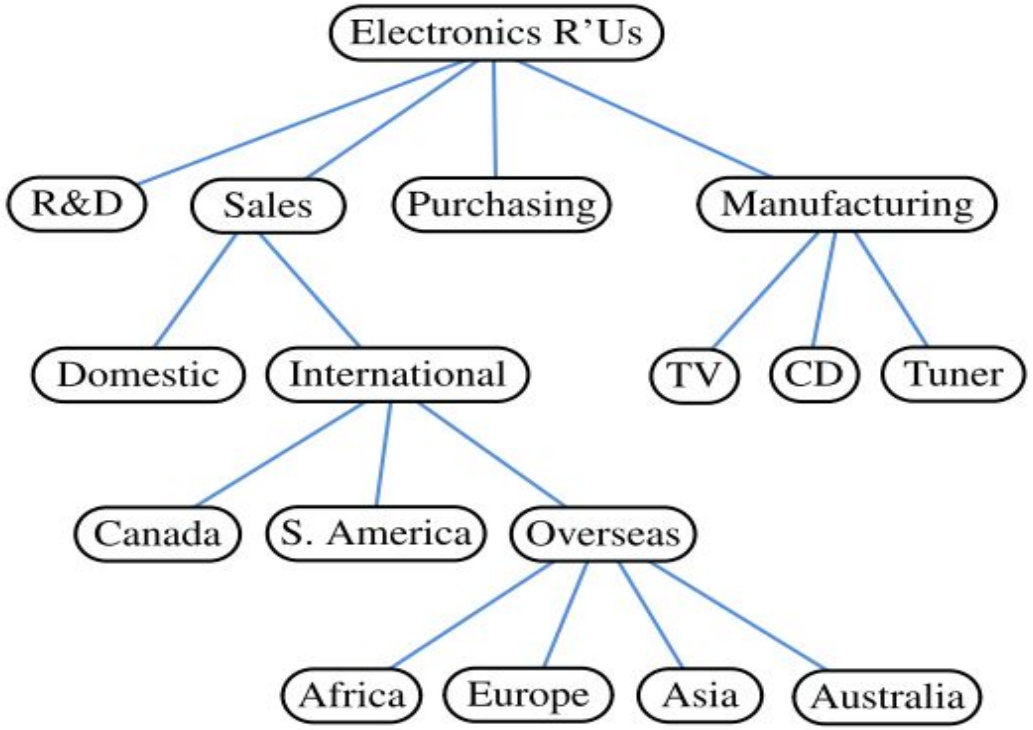
```
E:\Department Course\2020-July-Dec-Data Structure Sem -III\PPT>dir
 Volume in drive E has no label.
 Volume Serial Number is 520E-7749

 Directory of E:\Department Course\2020-July-Dec-Data Structure Sem -III\PPT

06-09-2020  12:10    <DIR>          .
06-09-2020  12:10    <DIR>          ..
08-08-2020  21:59    <DIR>          Godrich-book-slides
03-08-2020  05:45            58,978 Lecture-1- Overview.pptx
22-08-2020  20:17           403,557 Lecture-10- Circular Linked List.pptx
23-08-2020  05:27           774,294 Lecture-11-Recuesion.pptx
26-08-2020  05:39           522,678 Lecture-12-Recuesion Analysis.pptx
02-09-2020  09:45         1,101,808 Lecture-13-Positional Lists.pptx
29-08-2020  23:55         1,762,623 Lecture-14-Trees.pptx
05-09-2020  19:27         1,167,485 Lecture-15-Binary Trees.pptx
06-09-2020  11:07           772,218 Lecture-16-Trees Traversal.pptx
06-09-2020  12:10           312,615 Lecture-17-Trees Applications.pptx
03-08-2020  23:36           242,812 Lecture-2- OOPL-Java.pptx
10-08-2020  11:28           295,498 Lecture-3-Array Data Structure.pptx
10-08-2020  11:50           418,106 Lecture-4-Sorting Algorithms-1.pptx
11-08-2020  11:04           413,984 Lecture-5-AnalysisOfAlgorithm-1.pptx
12-08-2020  10:01           586,870 Lecture-6-AnalysisOfAlgorithm-2.pptx
17-08-2020  10:59           940,020 Lecture-7-Stack Data Structure.pptx
18-08-2020  10:54         1,192,213 Lecture-8-Queue Data Structure.pptx
19-08-2020  09:56           411,720 Lecture-9- Linked List.pptx
              17 File(s)     11,377,479 bytes
               3 Dir(s)  38,928,846,848 bytes free
```

Parenthetic Representations of a Tree

- The parenthetic string representation of tree shown in figure is:
  - Electronics R'Us (R&D, Sales (Domestic, International (Canada, S. America, Overseas (Africa, Europe, Asia, Australia))), Purchasing, Manufacturing (TV, CD, Tuner))



- If T consists of a single position p, then P(T) = p.getElement().
- Otherwise, it is defined recursively as,

P(T) = p.getElement()+"("+P(T1)+", "+ · · · +", "+P(Tk)+")"

where p is the root of T and T1,T2, . . . ,Tk are the subtrees rooted at the children of p, which are given in order if T is an ordered tree

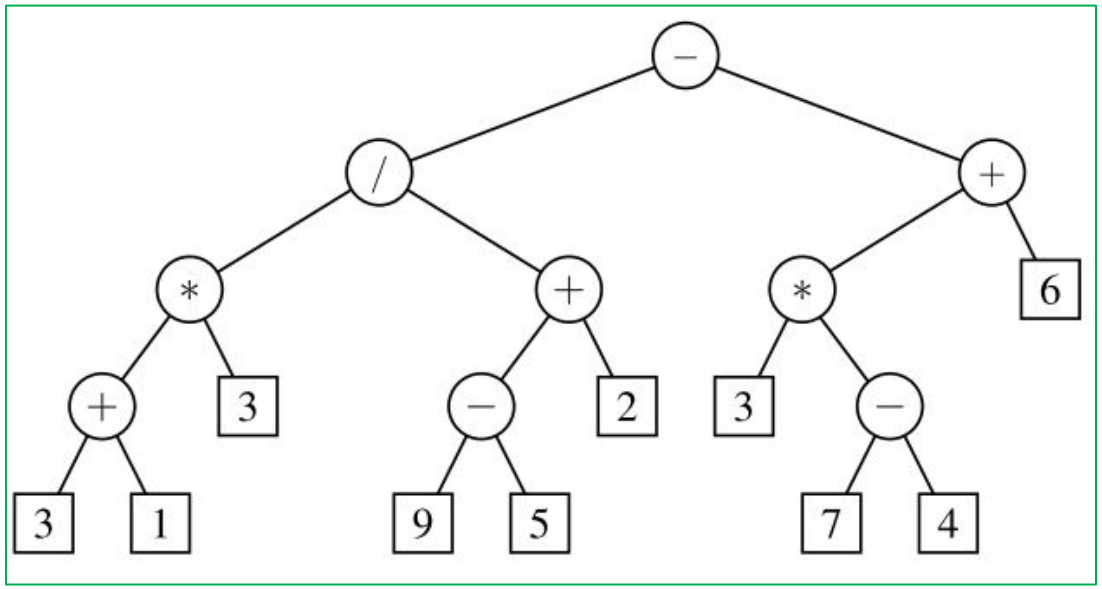
Parenthetic Representations of a Tree

- The Java method parenthesize, shown in Code Fragment, is a custom traversal that prints such a parenthetic string representation of a tree

```
/** Prints parenthesized representation of subtree of T rooted at p. */
public static <E> void parenthesize(Tree<E> T, Position<E> p) {
  System.out.print(p.getElement( ));
  if (T.isInternal(p)) {
    boolean firstTime = true;
    for (Position<E> c : T.children(p)) {
      System.out.print( (firstTime ? " (" : ", ") ); // determine proper punctuation
      firstTime = false;                             // any future passes will get comma
      parenthesize(T, c);                            // recur on child
    }
    System.out.print(")");
  }
}
```

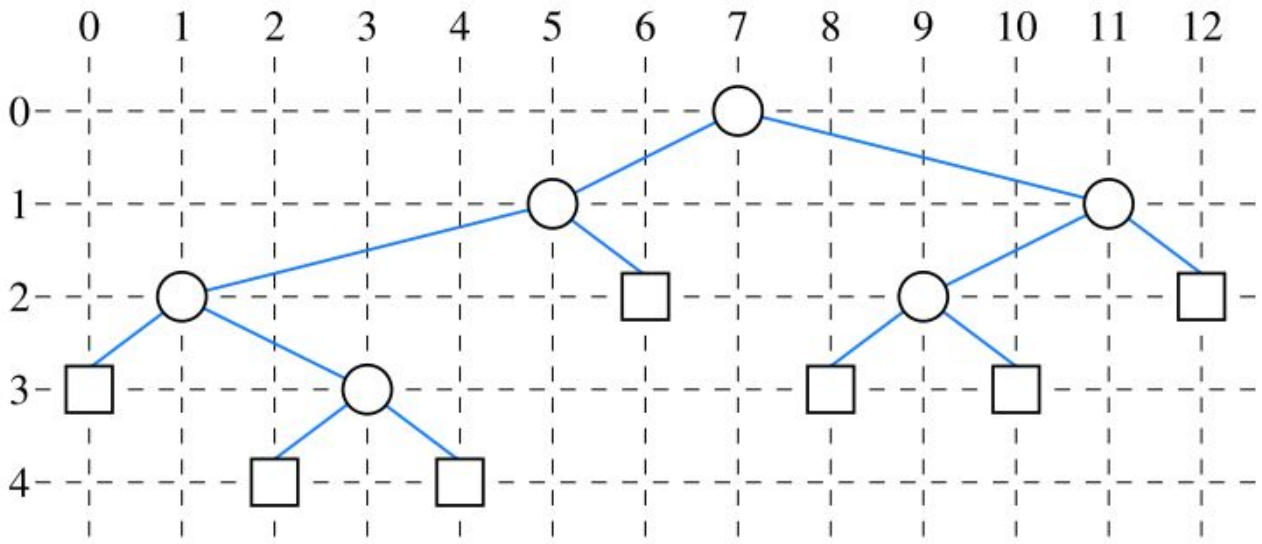
Home Assignemnt - Parenthetic Representations of arithmatic expression stored in a Tree

- Write a Java code to display arithmetic expression for a given tree
- ((((3+1)*3)/((9−5)+2))−((3*(7−4))+6))



Using Inorder Traversal for Tree Drawing

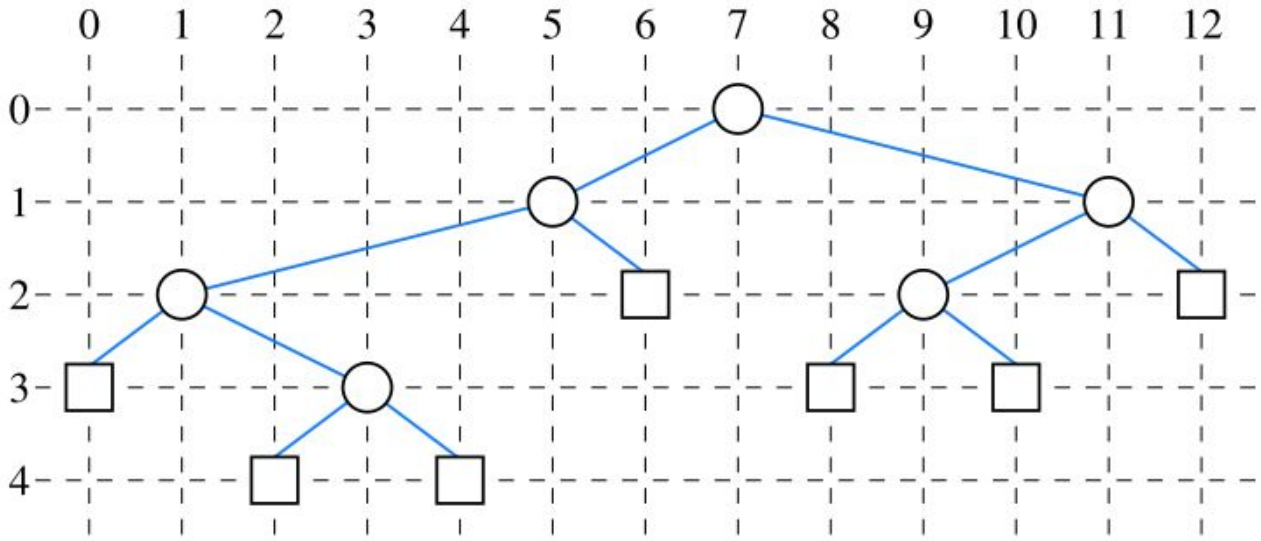
- An inorder traversal can be applied to the problem of computing a graphical layout of a binary tree
- In computer graphics, x-coordinates increase left to right and y-coordinates increase top to bottom, so that the origin is in the upper left corner of the drawing



Using Inorder Traversal for Tree Drawing

- The geometry is determined by an algorithm that assigns x- and y-coordinates to each position p of a binary tree T using the following two rules:

1. x(p) is the number of positions visited before p in an inorder traversal of T
2. y(p) is the depth of p in T



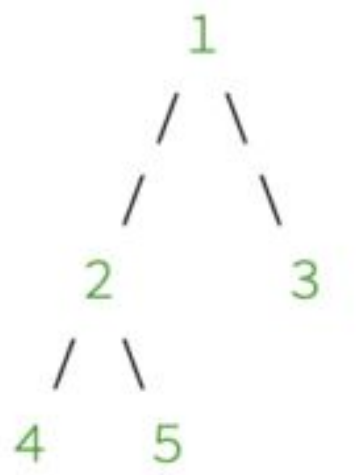
Using Inorder Traversal for Tree Drawing

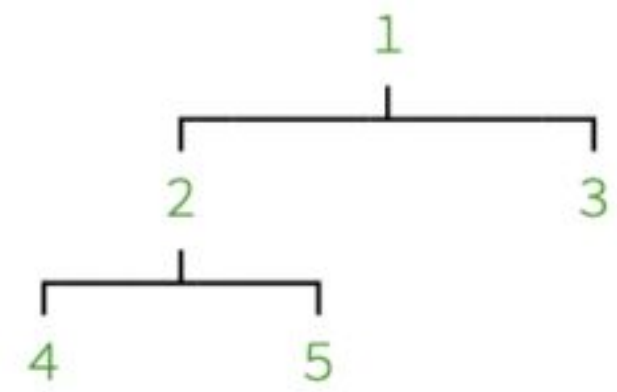
- We assume that the element type for the tree supports setX and setY methods
- The initial call should be layout(T, T.root(), 0, 0)

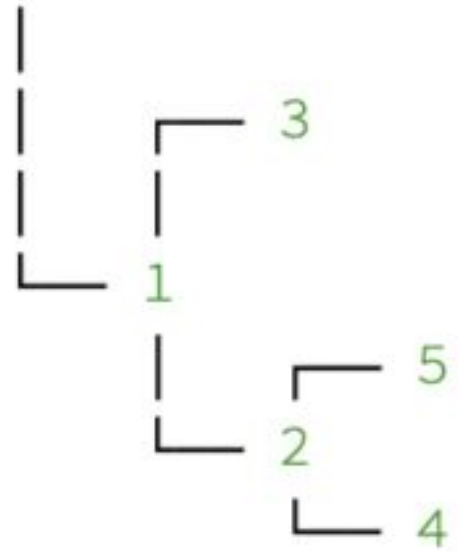
```
public static <E> int layout(BinaryTree<E> T, Position<E> p, int d, int x) {
  if (T.left(p) != null)
    x = layout(T, T.left(p), d+1, x);          // resulting x will be increased
  p.getElement( ).setX(x++);                   // post-increment x
  p.getElement( ).setY(d);
  if (T.right(p) != null)
    x = layout(T, T.right(p), d+1, x);         // resulting x will be increased
  return x;
}
```

Home Assignemnt - Using Inorder Traversal for Tree Drawing

- Write a Java program to display the binary tree on console in any one of the shown methods using tree traversal method.

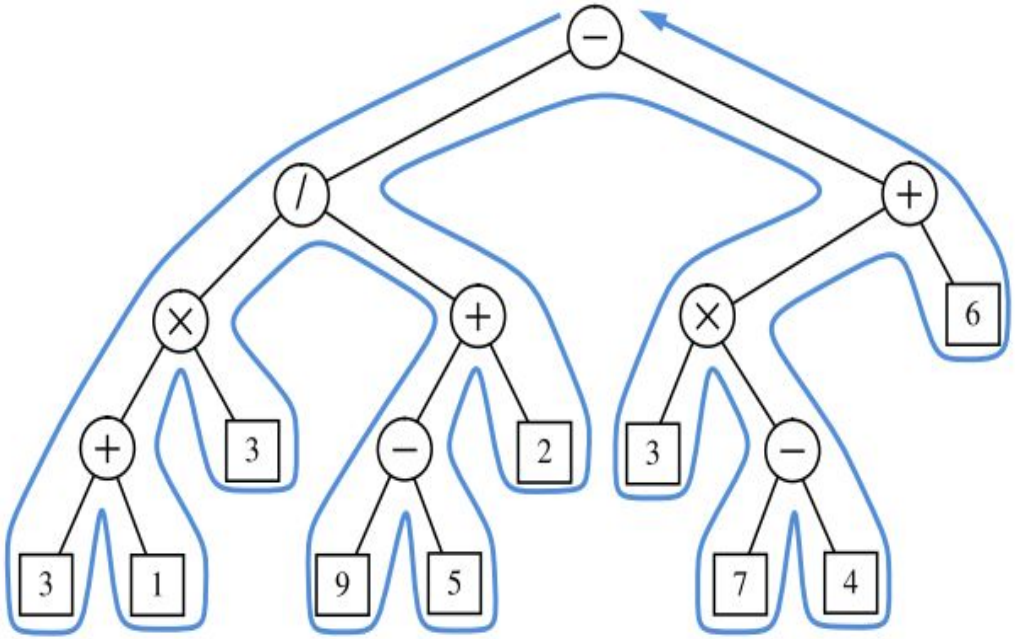






Euler Tours

- The Euler tour traversal of a tree T can be informally defined as a “walk” around T, where we start by going from the root toward its leftmost child, viewing the edges of T as being “walls” that we always keep to our left
- The complexity of the walk is O(n), for a tree with n nodes
- A “pre visit” occurs when first reaching the position, that is, when the walk passes immediately left of the node in our visualization
- A “post visit” occurs when the walk later proceeds upward from that position, that is, when the walk passes to the right of the node in our visualization



Algorithm eulerTour(T, p):

perform the “pre visit” action for position p
for each child c in T.children(p) **do**
  eulerTour(T, c) { recursively tour the subtree rooted at c }
perform the “post visit” action for position p

Euler Tours

- For the case of a binary tree, we can customize the algorithm to include an explicit “in visit” action

Algorithm eulerTourBinary(T, p):
  perform the “pre visit” action for position p
  if p has a left child lc **then**
    eulerTourBinary(T, lc) { recursively tour the left subtree of p }
  perform the “in visit” action for position p
  if p has a right child rc **then**
    eulerTourBinary(T, rc) { recursively tour the right subtree of p }
  perform the “post visit” action for position p

Euler Tours

- For example, a binary Euler tour can produce a traditional parenthesized arithmetic expression, such as "((((3+1)x3)/((9-5)+2))-((3x(7-4))+6))" for the tree
- “Pre visit” action: if the position is internal, print “(”
- “In visit” action: print the value or operator stored at the position
- “Post visit” action: if the position is internal, print “)”