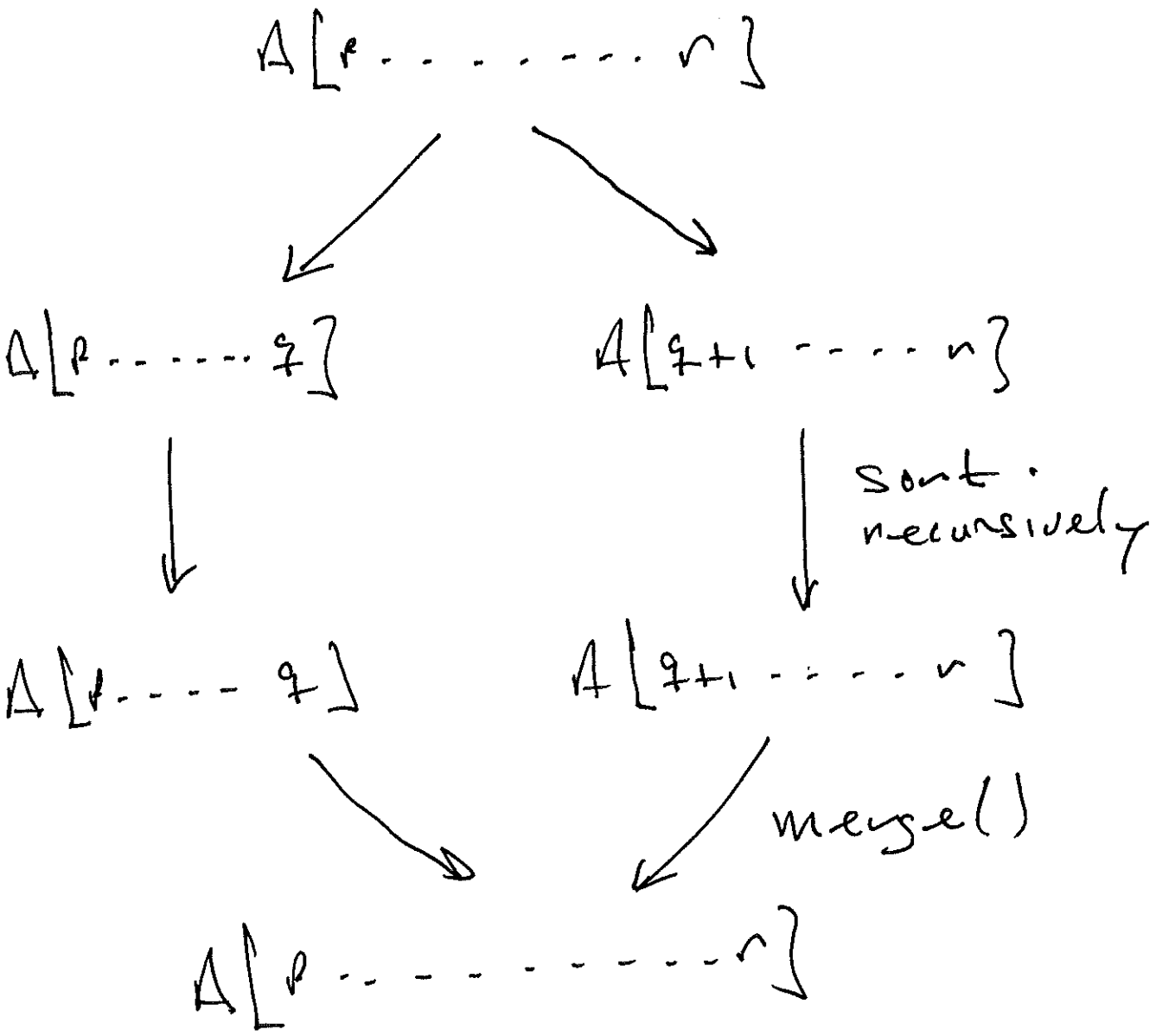
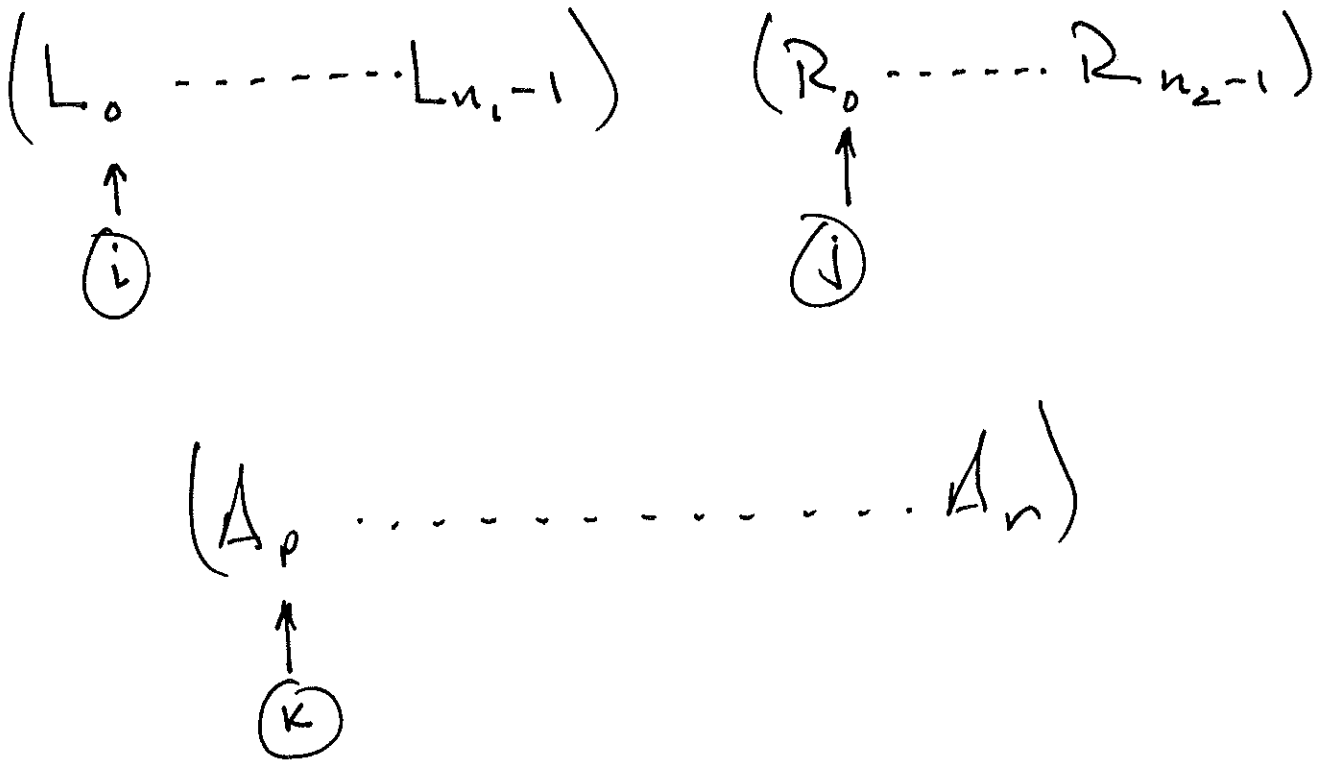
- lab 8 Posted soon
- asg 5 Posted soon
- mid 2: Wed 3-2 whole period.

recall mergeSort( )



what does merge do?

- let $n_1 = \text{length } A[p \ldots q] = q - p + 1$
- $n_2 = \text{length } A[q+1 \ldots r] = r - q$
- Copy $A[p \ldots q]$ into $L[0 \ldots n_1 - 1]$
- Copy $A[q+1 \ldots r]$ into $R[0 \ldots n_2 - 1]$



See MergeSort.java from W09.

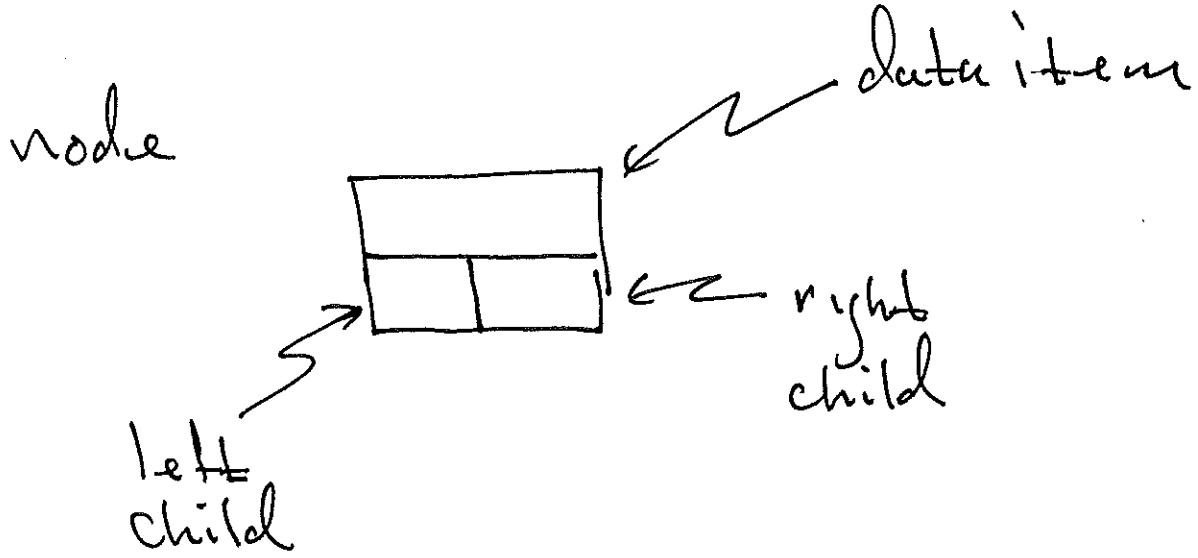
To convert to String:

change L[i] < R[j]

to L[i].compareTo(R[j]) < 0

Binary Search Trees (examples in C)

- A linked data structure



To create this in C:

```
typedef struct Node {
    int item;
    struct Node * left;
    struct Node * right;
} Node;
```

foo

bar

how does typedef work:

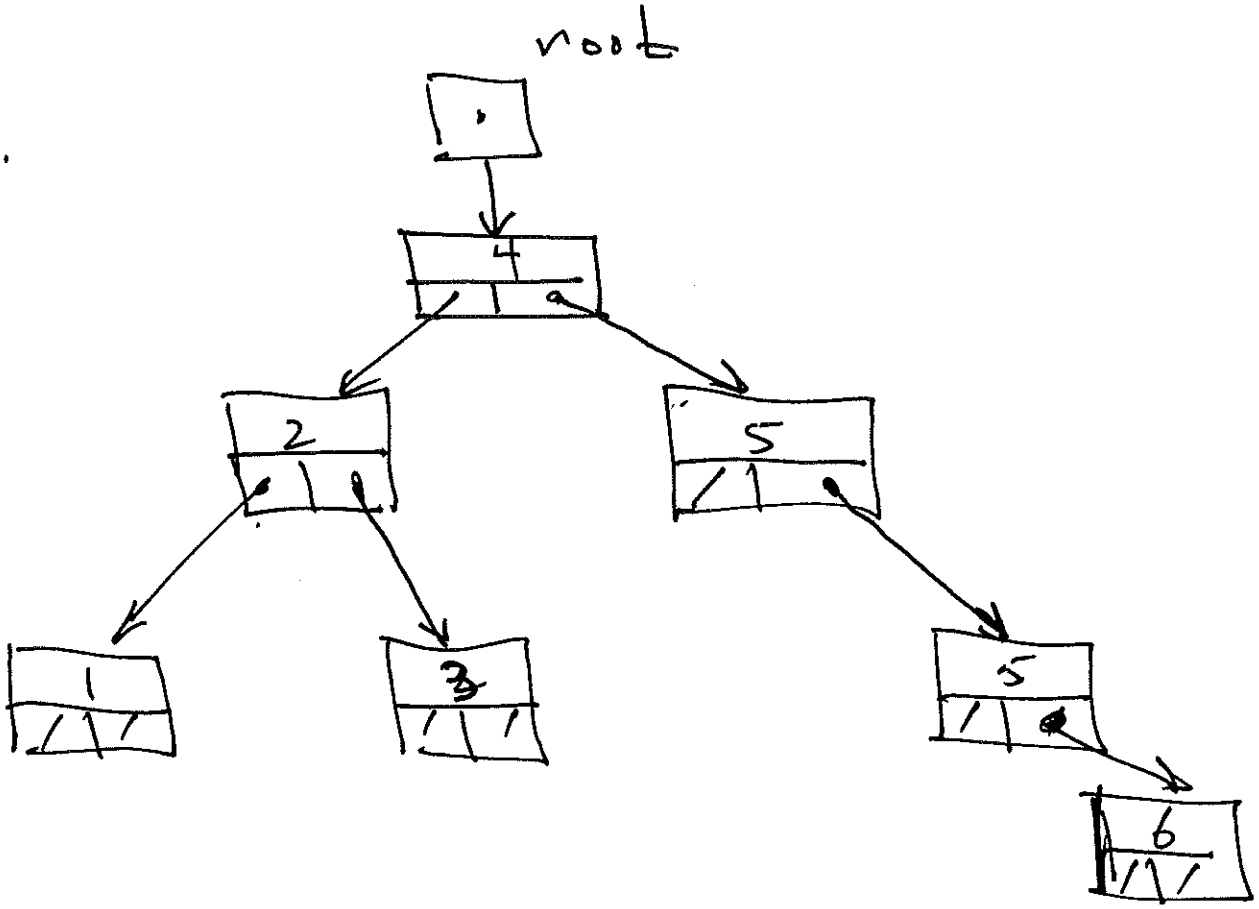
typedef foo bar;

↑ now an alias for foo

also do

typedef | foo: Node * | bar: NodeRef |;

Ex.



Recall BST Properties:

let x, y be NodeRef's

- if y is in left subtree of x:

  y->item <= x->item

- if y is in right subtree of x:

  x->item <= y->item

Ex. void PrintInOrder(NodeRef x){
    if (x != NULL){
        PrintInOrder(x->left);
        Printf("%d ", x->item);
        PrintInOrder(x->right);

Top level call: PrintInOrder(root);
output: 1 2 3 4 5

Exercises:

write:

```
void printPreOrder(NodeRef x){
    :
}
```

and:

```
void printPostOrder(NodeRef x){
    :
}
```

To Search a BST:

Ex
```
NodeRef findItem(NodeRef R, int K){
    if(R==NULL || K==R->item){
        return R;
    }
    if( K < R->item){
        return findItem(R->left, K);
    }else{
        return findItem(R->right, K);
    }
```

Top level call:

findItem(root, K);

(worst case)

runtime: O(height of tree)