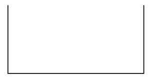
Trees – a brief digression

- You know about stacks
- And queues
- Trees are another kind of data structure

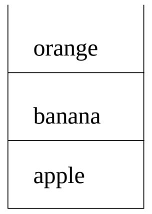
Stacks

the_stack = NEW_STACK



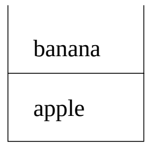
PUSH apple onto the_stack
PUSH banana onto the_stack
PUSH orange onto the_stack

TOP of the_stack = orange



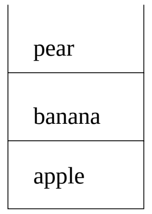
POP the_stack

TOP of the_stack = banana



PUSH pear onto the_stack

TOP of the_stack = pear

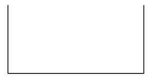


Stack Operations

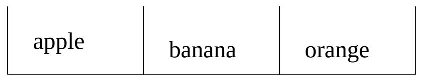
- new stack – creates an empty stack
- push
- pop
- top
- new and push are used to build stacks
- top and pop can be used to iterate over the stack elements

Queues

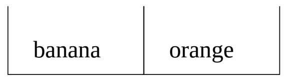
the_queue = NEW_QUEUE



JOIN apple to the_queue
JOIN banana to the_queue
JOIN orange to the_queue

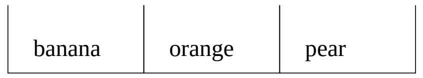


FRONT of the_queue = apple



LEAVE the_queue

FRONT of the_queue = banana



ADD pear to the_queue

FRONT of the_queue = banana

Queue Operations

- new queue – creates an empty queue
- join
- leave
- front
- new and join are used to build queues
- front and leave can be used to iterate over the queue elements

Trees: empty tree, leaf

the_empty_tree = EMPTY_TREE

the_empty_tree has no ROOT, no LCHILD, no RCHILD

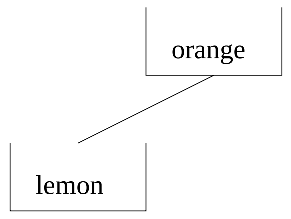
the_lemon_tree = LEAF(lemon)
same as
the_lemon_tree = TREE(EMPTY_TREE, lemon, EMPTY_TREE)

ROOT of the_lemon_tree = lemon
LCHILD of the_lemon_tree = EMPTY_TREE
RCHILD of the_lemon_tree = EMPTY_TREE

A tree with a left child

the_left_lemon_tree =

TREE(the_lemon_tree, orange, EMPTY_TREE)

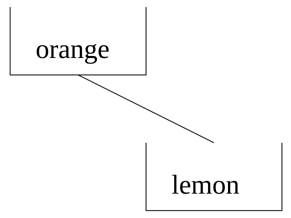


ROOT of the_left_lemon_tree = orange
RCHILD of the_left_lemon_tree = EMPTY_TREE
ROOT of LCHILD of the left_lemon_tree = lemon

A tree with a right child

the_right_lemon_tree =

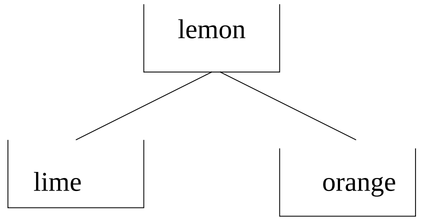
TREE(EMPTY_TREE, orange, the_lemon_tree)



ROOT of the_right_lemon_tree = orange
LCHILD of the_right_lemon_tree = EMPTY_TREE
ROOT of RCHILD of the right_lemon_tree = lemon

A tree with two children

```
the_citrus_tree =
   TREE(
      TREE( EMPTY_TREE, lime, EMPTY_TREE )
      lemon,
      TREE( EMPTY_TREE, orange, EMPTY_TREE )
```



ROOT of the_citrus_tree = lemon

ROOT of LCHILD of the_citrus_tree = lime

ROOT of RCHILD of the_citrus_tree = orange

LCHILD of LCHILD of the_citrus_tree = EMPTY_TREE

RCHILD of LCHILD of the_citrus_tree = EMPTY_TREE

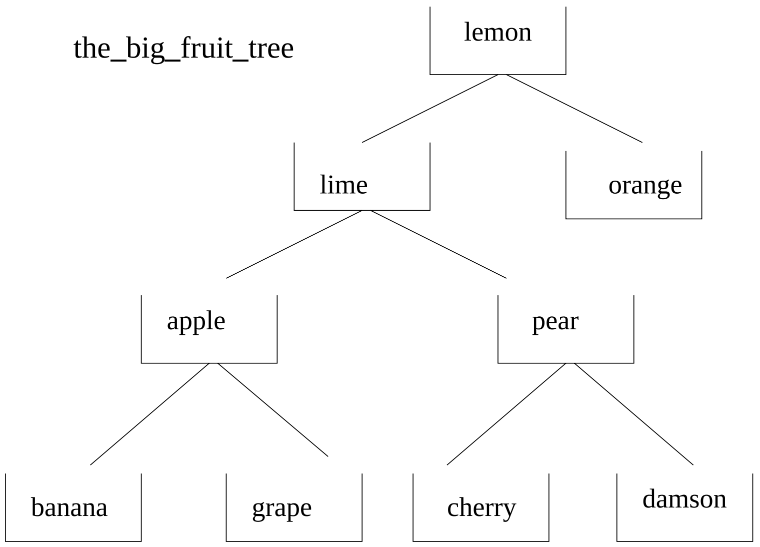
LCHILD of RCHILD of the_citrus_tree = EMPTY_TREE

RCHILD of RCHILD of the_citrus_tree = EMPTY_TREE

Tree Operations

- empty tree – creates an empty tree
- leaf
- tree
- root
- lchild
- rchild
- empty, leaf and tree are for building trees
- root, lchild and rchild can be used to iterate over tree elements

Inorder, Preorder, Postorder



INORDER (the_big_fruit_tree) =
    banana apple grape lime cherry pear damson lemon orange

PREORDER (the_big_fruit_tree) =
    lemon lime apple banana grape pear cherry damson orange

POSTORDER (the_big_fruit_tree) =
    banana grape apple cherry damson pear lime orange lemon

A C header file

```
#ifndef TREE_LABEL_T
#define TREE_LABEL_T int
#endif

typedef struct tree_node{
        TREE_LABEL_T this;
        struct tree_node *left;
        struct tree_node *right;
} tree_t;

tree_t *mknode(TREE_LABEL_T, tree_t *, tree_t *);

tree_t *mkleaf(TREE_LABEL_T);

void postorder(tree_t *, int (*)(TREE_LABEL_T));
void preorder(tree_t *, int (*)(TREE_LABEL_T));
void preorder(tree_t *, int (*)(TREE_LABEL_T));
```

Using the C tree implementation

```c
#include <stdio.h>
#include <stdlib.h>
#define TREE_LABEL_T char *
#include "tree.h"

int print_label(char *label) {
  printf("%s ",label);
  return 0;
}

int main(void) {
  tree_t *the_big_fruit_tree =
    mknode( "lemon",
      mknode( "lime",
        mknode( "apple",
          mkleaf( "banana" ), mkleaf( "grape" )),
        mknode( "pear",
          mkleaf( "cherry" ), mkleaf( "damson" ))),
      mkleaf( "orange" ));

  printf("\nInorder traversal: \n");
  inorder(the_big_fruit_tree, print_label);

  printf("\n\nPreorder traversal: \n");
  preorder(the_big_fruit_tree, print_label);

  printf("\n\nPostorder traversal: \n");
  postorder(the_big_fruit_tree, print_label);
  exit(0);
}
```

Running ‘the big fruit tree’

```
marilyn% the_big_fruit_tree

Inorder traversal:
banana apple grape lime cherry pear damson lemon orange

Preorder traversal:
lemon lime apple banana grape pear cherry damson orange

Postorder traversal:
banana grape apple cherry damson pear lime orange lemon
```