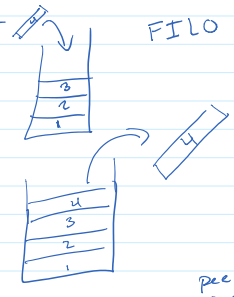
Stacks & Queues (cont.)

Thursday, September 24, 2015 11:40 AM

```
(define sum (list)
  ....)
(sum 1 2 3)
→ 6
```

Software Engineering Club
Tuesdays 6-7pm ACM room (2A...)
Kargers Algorithm (next week)

Stack — FILO

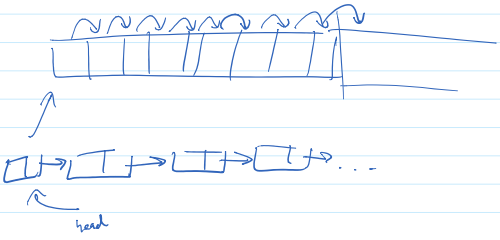


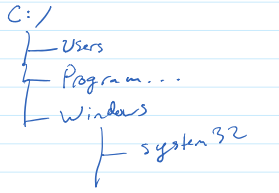
Stack<E> {

peek
pop
push

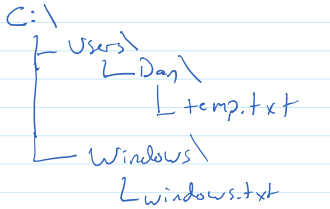
```
List<E> theList;
```

```
void push(E element) {
    theList.insert(0, element);
}

E pop() {
    return theList.remove(0);
}

E peek() {
    return theList.get(0);
}
int size() { return theList.size(); }
```

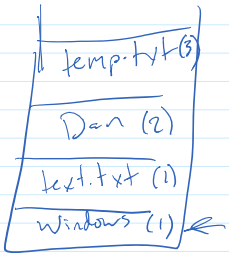





```
function listContents(File root) {
    if (root.isDirectory()) {
        print(root.name);
        return;
    }
    Stack<File> stack = new Stack<File>();
    stack.push(root);
    while stack is not empty:
```



listContents('C:\')



C:\

```
Stack<File> stack = new Stack<File>();
stack.push(root);
while stack is not empty:
    File curFile = stack.pop();
    print(curFile.name());
    if (!curFile.isDirectory()) {
        continue;
    }
    File[] curContents = curFile.getContents();
    for (File child : curContents) {
        stack.push(child);
    }
}
```

listContents('C:\')

C:\
Windows
windows.txt
Users
Dan
temp.txt

C:\
Users\
text.txt
Windows\

depth+1

```
class FileAndDepth {
    File file;
    int depth;
}

void listContents(root) {
    if ... {
        break
    }
    Stack<FileAndDepth> stack = ...;
    stack.push(new FAD(root, 0));
    while stack is not empty {
        curFile = stack.pop();
        for (int i=0; i<curFile.depth*2; i++) {
            print(" ");
        }
        println(curFile.file.name());
        if(...) {
            continue
        }
        File[] curContents = curFile.file.contents();
        for (child : curContents) {
            stack.push(new FAD(child, curFile.depth+1));
        }
    }
}
```

C:\
Users\
Dan\
temp.txt
temp.txt (1)
Windows\ (1)
windows.txt

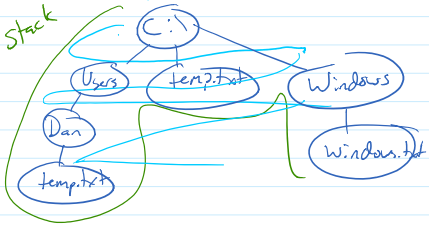
listContents('C:\')

C:\
Windows\
windows.txt
temp.txt
Users\
Dan\
temp.txt

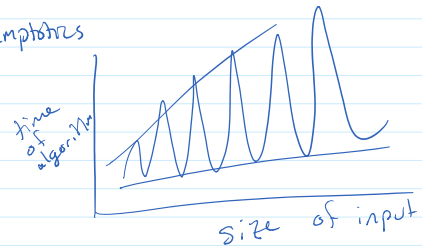
C:\
Users\
temp.txt
Windows\
Dan
windows.txt
temp.txt

Stack

Queue



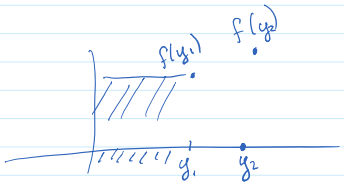
Asymptotics



Defn a monotonic increasing function is a function that satisfies

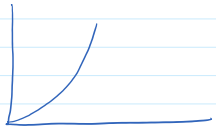
$$\forall\ x < y,\quad f(x) < f(y)$$

$\forall$: "for all"



$f(x) = x$

$f(x) = x^2$



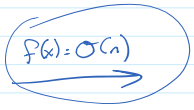
Defn: A positive semidefinite function is a function s.t.

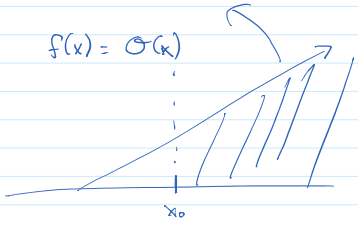
$$\forall\ x,\quad f(x) \geq 0$$

We say a function $f \in O(n)$ if

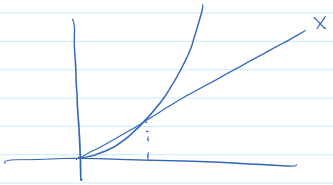
$$\exists\ c, n_0 \text{ s.t. } \forall\ n > n_0 \quad f(n) \leq cn$$

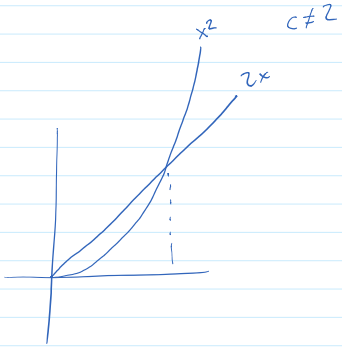
$\in$ "in"
$\exists$ "exists"
$\forall$ "for all"



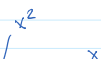


$x^2 \stackrel{?}{=} O(x)$





$x^2 \neq O(x)$ "$x^2 > x$"



$x^2 \neq \mathcal{O}(x)$

$x \overset{?}{=} \mathcal{O}(x^2)$ ✓

" $x^2 > x$ "