

Class4

Summary

A stack is a dynamic data structure that follows a LIFO order.

Dynamic Data Structures

- linked lists
- front -> node ... node -> nullptr
- represent {11, 12, 13}

To remove the first element, change the front pointer to the next item and delete the old reference

To add an element to the front, change front pointer and make the new node point to the previous front

List find operation

find(x) - true if x in on the list, otherwise false

In an array:

```
for i=0 ... n-1 {  
  if a[i] = x  
    return True  
}  
return False
```

In a linked list:

```
c <- first  
while c is not nullptr {  
  n <- node c points to  
  if data stored at n is x  
    return True  
  c <- next pointer in n  
}  
return False
```

Stack

ADT that stores a collection of objects in Last-In-First-Out order

Necessary

push(x): inserts an element at the top

pop(x): removes an element at the top

Convenient:

isEmpty(): check for emptiness

size(): return number of elements on stack

top(): return top, but don't remove it

Stack based algorithms for checking grouping symbols

```
S <- new stack
while there are symbols to read
  c <- next symbol
  if e is a left symbol
    push c on S
  else
    if S is empty report error
    d <- pop S
    if c,d do not match report error
end while
if S is not empty report error
report "ok"
```