CSE 101-01 4-4-23

See handout ADT.pdf

Abstract Data Type

consists of

(1) a set S of 'structures' called states.

(2) a set of operations on states.

- Manipulation Procedures:
  change the state of an instance.
- Access functions:
  return information on a state.

Ex. Queue ADT (of integers)

States: finite sequence of ints.

	len
• (5, 1, -7, 3, 5, 8, -13, 12)	8
• (10)	1
• ( ) empty state	0

Operations:

Enqueue()
Dequeue()
getFront()
getLength()
isEmpty()

History of states:

OP.	State	return
	( )	—
Enqueue(6)	(6)	—
Enqueue(9)	(6, 9)	—
Enqueue(-1)	(6, 9, -1)	—
Dequeue( )	(9, -1)	—
getFront( )	(9, -1)	9
isEmpty( )	(9, -1)	false
⋮	⋮	⋮

note:

Dequeue() and getFront() are undefined on () empty state.

Preconditions

define which states an OP. can be applied to.

Both Dequeue() and getFront() have Precondition

! isEmpty()

Policy: If a precond. is violated, Print an error msg. and quit the Program.

Error msg. should state:

- what ADT
- what operation
- what precond.

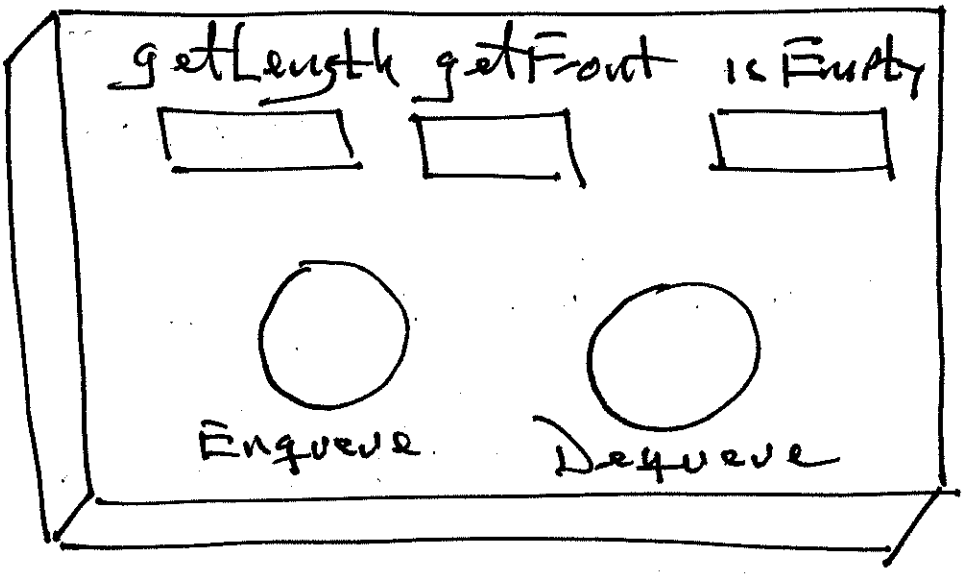
How to call Dequeue()

In C: Dequeue(Q) ← Queue instance

In C++: Q.Dequeue()

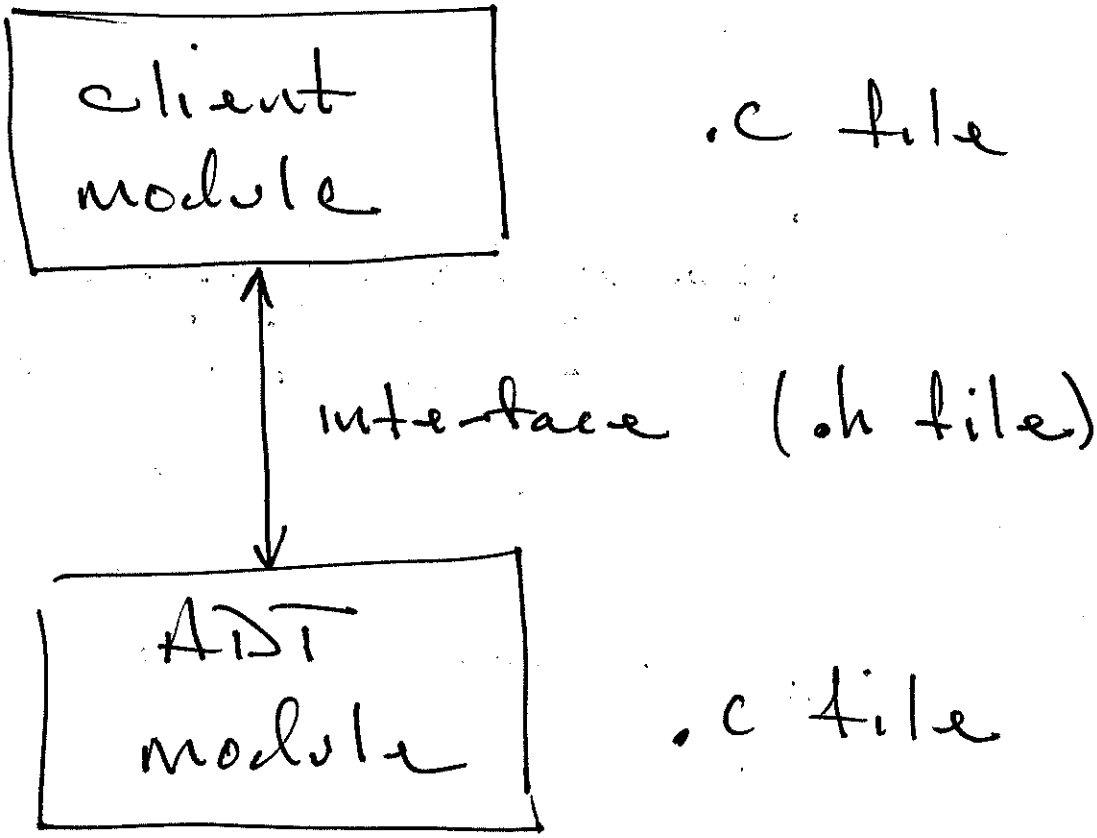
↑ implicit 1st argument

Black Box Picture:

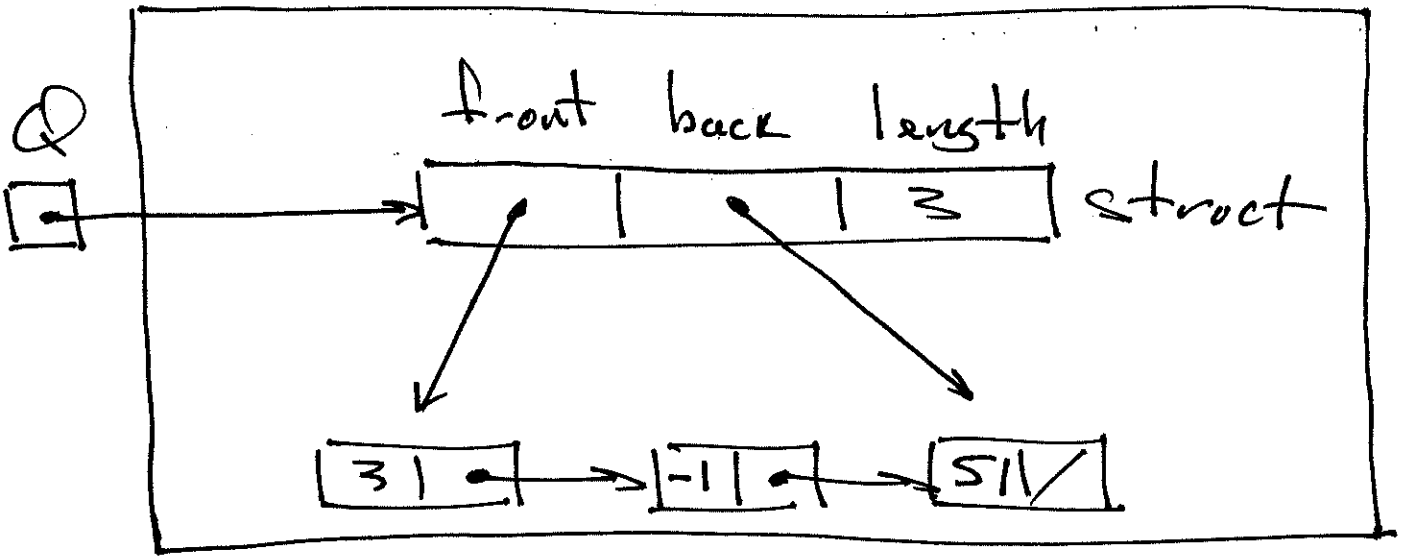


Information Hiding:

client interacts with ADT instance only through ADT ops.

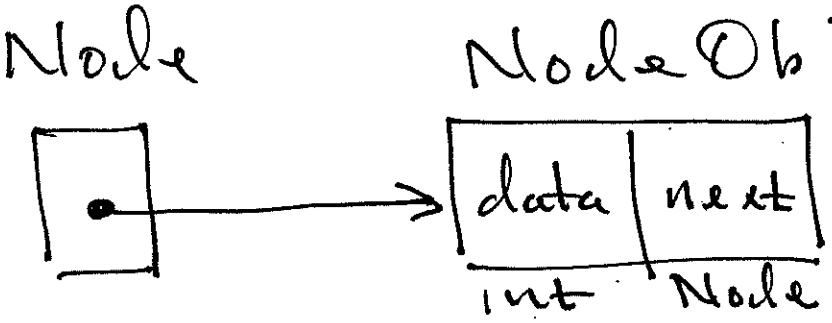


Inside the Queue black box!



Client View: (3, -1, 51)

The Node Type: (in C)



How to create Node Type!

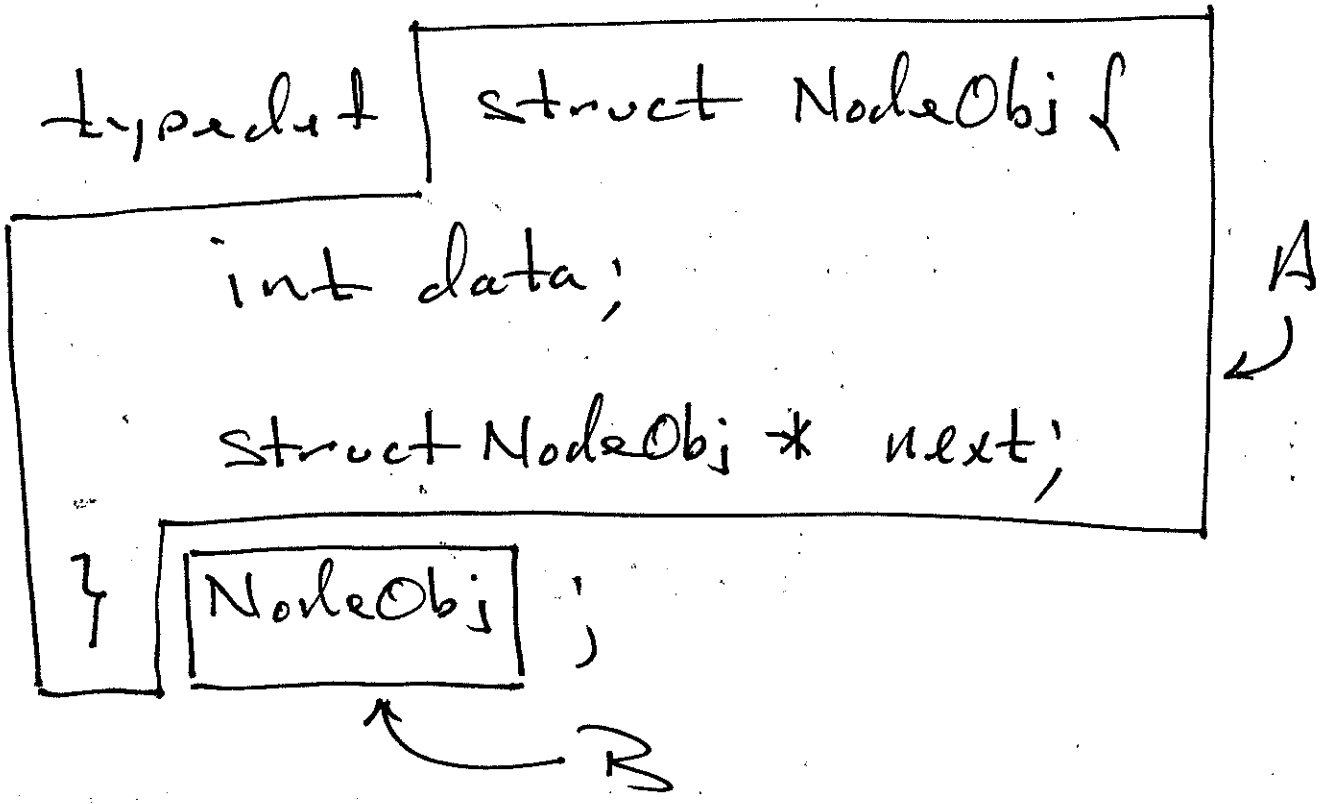
Typedef in C:

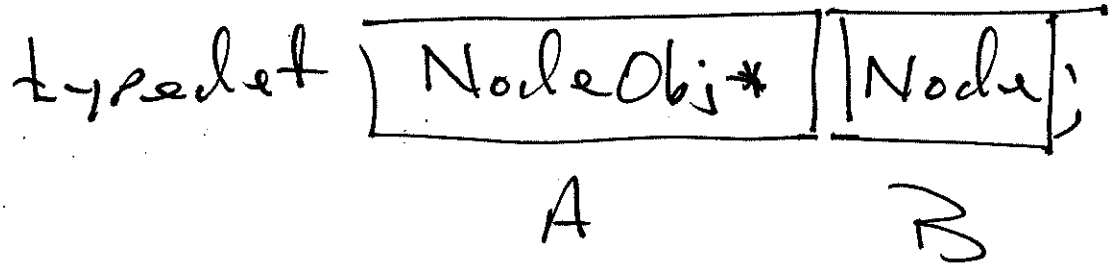
```
typedef A B;
```

↑ existing type ↑ type we are creating.

type B is an alias for A.

Ex.


typedef struct NodeObj {
int data;
struct NodeObj * next;
} NodeObj;
A
B


typedef NodeObj* Node;
A
B