

Announcements

Reminder: Candidates on campus in next few weeks
Day-by-day schedule may change, so check often

Thursday Extra:

Today @ 4:00

Tenure-track candidate #1

Kevin Angstadt, U. of Michigan

All Computers Great and Small: Supporting and Security the
Systems of Tomorrow

Announcements

Stacks and Queues

Two levels of interest

Conceptual: FIFO or FILO/LIFO

Implementation: Either arrays or linked lists

Both widely used in practice!

Stacks: piles of materials (e.g., on a desk)
constrained run-time stack

Queues: lines in stores/airports
jobs sent to printer (if strict FIFO)

Questions; Clicker Questions

Assume:

add	means either	push	or	enqueue
remove	means either	pop	or	dequeue
get	means either	top	or	front

Notes:

- most stack ADTs include `top`
- queue ADTs may or may not include `front`

For which ADTs can a get operation be implemented simply from add and remove? (A simple extra variable might be used, but not an extra array, linked list, or other structure.)

- A. Stack
- B. Queue
- C. Both
- D. Neither

Assume:

- add means either push or enqueue
- remove means either pop or dequeue
- get means either top or front

Consider the program:

```
Container a;  
add(a, "Orange");  
add(a, "Red");  
add(a, "Green");  
remove(a);  
add(a, "Blue");  
printf("%s", get(a));
```

The program prints "Red".
Container must be a ...

- A.Stack
- B.Queue
- C.Neither
- D.Impossible to tell

Assume:

- add means either push or enqueue
- remove means either pop or dequeue
- get means either top or front

Consider the program:

```
Container b;  
add(b, "Happy");  
remove(b);  
add(b, "Sneezy");  
add(b, "Dopey");  
add(b, "Grumpy");  
printf("%s", get(b));
```

The program prints
"Dopey".

Container must be a ...

- A.Stack
- B.Queue
- C.Neither
- D.Impossible to tell

Assume:

- add means either push or enqueue
- remove means either pop or dequeue
- get means either top or front

Consider the program:

```
Container c;  
add(c, "Helium");  
add(c, "Neon");  
remove(c);  
printf("%s", get(c));
```

The program prints
"Helium".

Container must be a ...

- A.Stack
- B.Queue
- C.Neither
- D.Impossible to tell

Today's Project: Underlying questions:

When using a stack to store a string,

- what should be stored on the stack as part of push?
 - a pointer
 - to the original string
 - to a copy of the string
 - a copy of the string
- what pointer should be returned as part of pop?
 - a pointer to the original
 - a pointer to a copy on the run-time stack
 - a pointer to a copy in dynamic memory

Approach of project:

- four different implementations (all callable by same main)
- each implementation yields different output
- practical question: why:
 - why does each approach produce its output?
 - explain carefully!