Announcements

Reminder: Candidates on campus after break
Some may teach a section of this course
Please be attentive, engaged, respond to questions, etc.!
Schedule likely unchanged (but modest change possible)

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 get be simply implement 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
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
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
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
Header file stringStack.h is given, where details of the stringStack typedef will be included.

Implementation details will be given in file stringStack.c.

Assume a user's program will use the stringStack by writing

```c
#include "stringStack.h"
```

What changes should be needed in the program if the implementation changes?

A. None
B. Modest changes likely
C. Extensive review needed

```c
/* type declaration for a stack of strings */
typedef
...
stringStack;

/* initialize stack referenced by the parameter */
void initializeStack (stringStack * stack) ;

/* return whether the stack is empty */
int isEmpty (stringStack stack) ;

/* return whether the stack is full */
int isFull (stringStack stack) ;

/* insert given item onto top of the stack */
int push (stringStack *stack, char * item) ;

/* removes top item from stack referenced by parameter */
char * pop (stringStack *stack) ;

/* returns a pointer to top item on the stack */
char * top (stringStack stack) ;
```