CS2351
Data Structures

“Pass by Value” or “Pass by Pointer”
In today’s lecture, we mentioned two ways to perform enqueue in a queue.

- In fact, one of the ways is WRONG.

We shall see why it is wrong.
The wrong enqueue

- The following function tries to perform enqueue using pass by value:

```c
void enqueue( struct node *head,
              struct node *tail, struct node y )
{
    if ( head != NULL ) // if not empty
    {
        tail->next = &y ;  tail = &y;  
    }
    else
    {
        head = tail = &y ; 
    }
}
```
The wrong enqueue

• Unfortunately, it is WRONG
• Suppose in the main program, we write:

```c
struct node *head, *tail, x;
head = NULL;
enqueue( head, tail, x );
```

• What happens is that after `enqueue`, `head` nor `tail` do not point at `x` as we expect
• Reason: During `enqueue`, there is a temp memory space for the local variable `y`...
The wrong enqueue

- Then, \( y \) copies all the contents of \( x \) from the main program.

- Then inside \texttt{enqueue}, the local variables \texttt{head} and \texttt{tail} are assigned to point at \( y \).

- Consequently, \texttt{head} and \texttt{tail} in main program have no change, and no one points at \( x \)!
The correct enqueue

• We need to ensure that after enqueue, both head and tail in the main program are set correctly (point at x, not y)

• In this case, we shall use pass by pointers
  - Another scheme called pass by reference may be used instead

• Since values of head and tail need to be changed, we need to know their actual memory addresses inside enqueue
The correct enqueue

- To pass the address of a pointer $p$ is easy
  - Simply use a pointer that points at $p$

```c
void enqueue( struct node **head,
              struct node **tail, struct node *y )
{
    if ( *head != NULL ) // if not empty
    { (*tail)->next = y ; (*tail) = y; }
    else
    { (*head) = (*tail) = y ; }
}
```
The correct enqueue

• Then inside the main program, we write:

```c
struct node *head, *tail, x;
head = NULL;
enqueue( &head, &tail, &x );
```

• After the enqueue call, the actual memory locations of both head and tail will be filled with the actual address of x