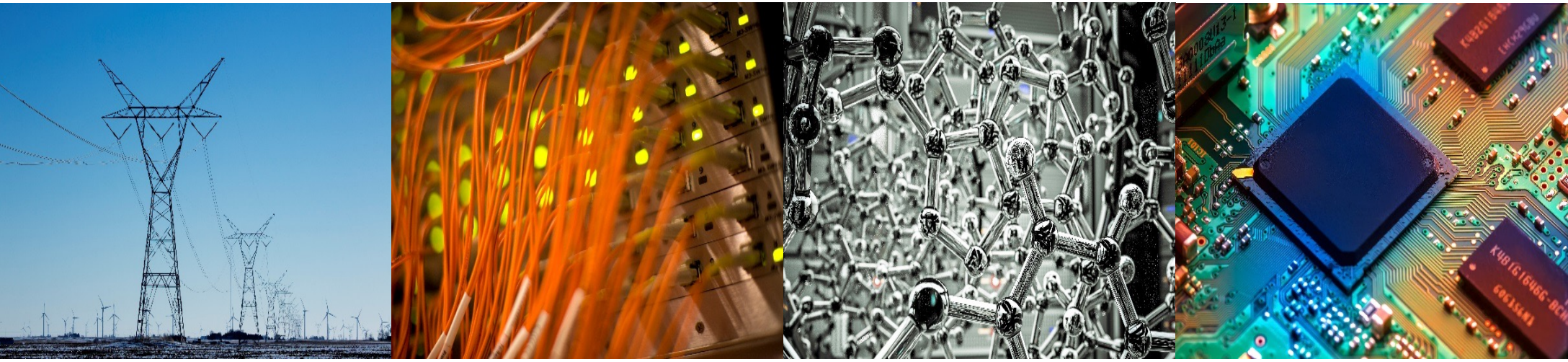
ECE 220 Computer Systems & Programming

Lecture 19 – C to LC-3 with Linked Data Structure

April 7, 2026



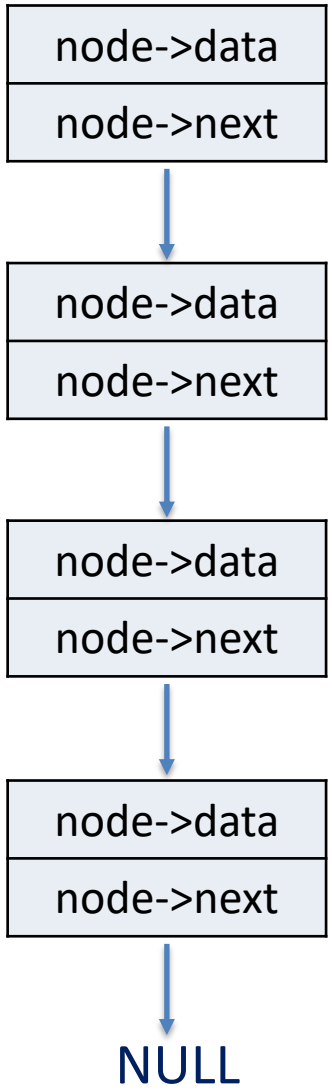
ILLINOIS
Electrical & Computer Engineering
GRAINGER COLLEGE OF ENGINEERING

Exercise: C to LC-3 Conversion - Linked List Traversal

```
typedef struct nodeStruct node;
struct nodeStruct{
   char data;
   node *next;
};

void print_list(node *head){
   if(head == NULL)
      return;
   printf("%c", head->data);
   print_list(head->next);
}
```

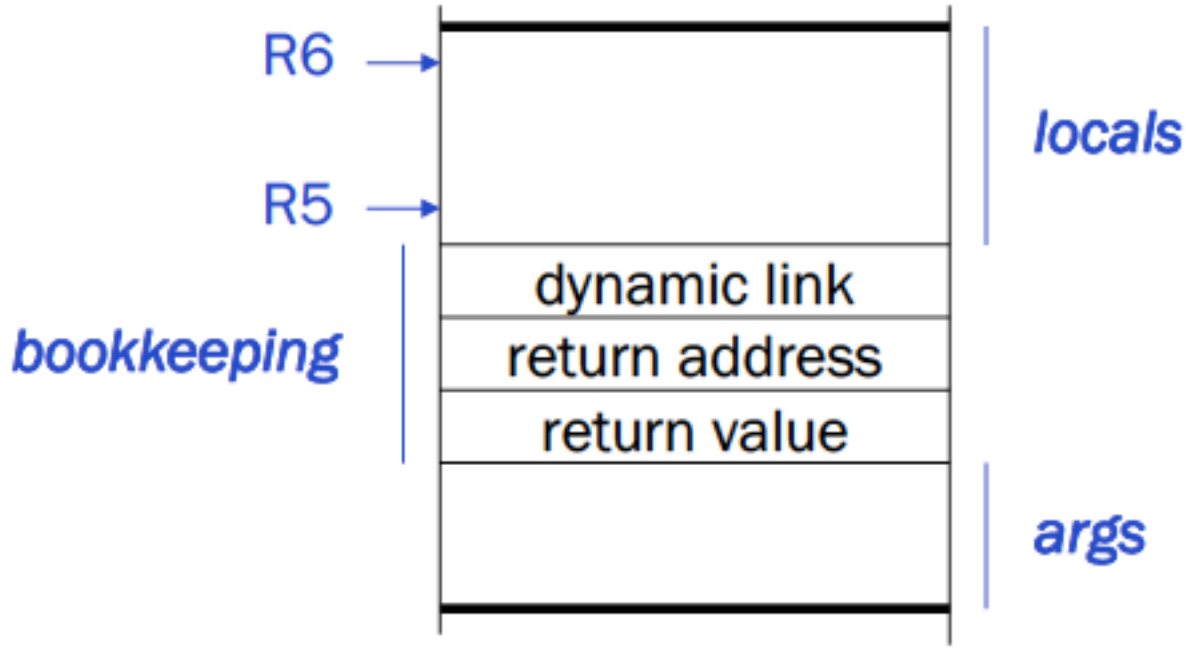
C to LC-3 Conversion

Linked list starts at **x4004**




```
;data.asm

.ORIG x4000

.FILL x45
.FILL x4006

.FILL x43
.FILL x4000

.FILL x45
.FILL x4002

.FILL x42
.FILL x0

.END
```

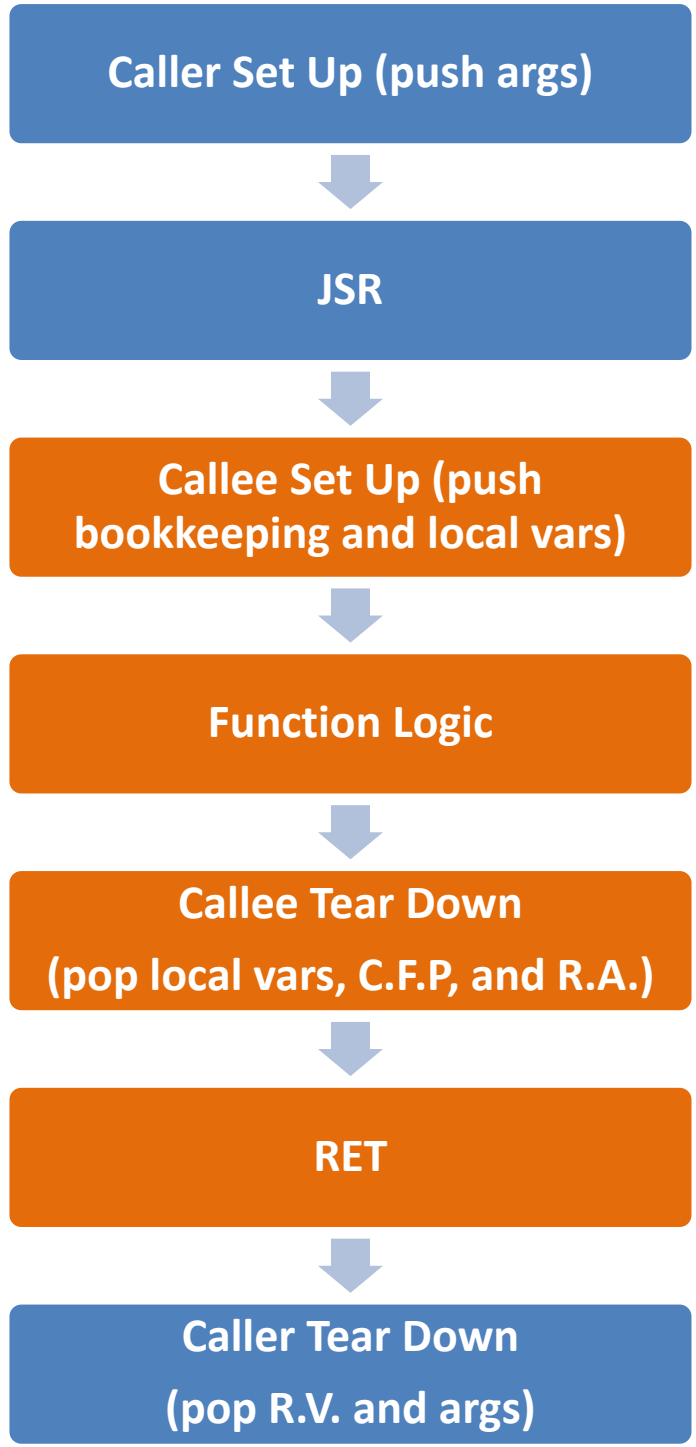
Function's Activation Record



	← R5
Caller's Frame Pointer	← R6
Return Address	
Return Value	
head (x4004)	

x4004	head->data
x4005	head->next

Stack Built-up and Tear-down



```
.ORIG x3000
MAIN LD R5, RSTACK      ;set R5 to the bottom of the stack
     LD R6, RSTACK      ;set R6 to the same address
     LD R0, HEAD        ;load head pointer to R0
     STR R0, R6, #0     ;set up local variable 'head' in main
     ADD R6, R6, #-1    ;caller setup: push callee's argument to RTS
     LDR R1, R5, #0     ;load 'head' from RTS to R1
     STR R1, R6, #0     ;push 'head' (argument) to RTS
     JSR PRINT_LIST
     ADD R6, R6, #2     ;caller tear-down
     HALT
HEAD .FILL x4004
RSTACK .FILL x7000
```

```
PRINT_LIST
;;Part 1 - callee setup: push bookkeeping info




;;Part 2 - implement function logic
   ;if(head == NULL) return;
```

```
;printf("%c", head->data);


;print_list(head->next);
;;;caller setup: push 'head->next' to RTS




;;;transfer control to callee

;;;caller tear-down: pop callee's R.V. and argument
```

```
;;Part 3 - callee tear down: prepare to return
TEAR_DOWN




RET
.END
```