
Recitation Week #10
Project 2-3:
Help Session

Alejandro Cabrera
Operating Systems
COP4610 / CGS5765

Today's Recitation

- Project 2-3 Clarifications
 - Deadline Extension
 - Documentation Requirements
 - procFS Buffer Limit
 - Bank Test Driver
 - Kernel Linked List

Deadline Extension

- Previously due: Yesterday, 5:00PM
- Now due: Monday, by midnight

Documentation

- Minimum required for full credit:
 - Cover only project 2-3
 - README.txt
 - report.txt
- Optional:
 - Cover all of project 2
- Submitted project documentation helps me plan future projects better. Any additional documentation is appreciated.

procFS Buffer Limit

- Using simple procFS read function implementation, it is impossible to print out all bank account info for a large number of bank accounts.
- Requirement:
 - Make sure your procFS output can handle at least 10 bank accounts

procFS Buffer Limit

- Buffer limit seems to be about 2 or 4KB – a typical memory page size.
- Breaking the limit:
 - Investigate seq_file interface for procFS entries.
 - Seq file documentation:

http://lxr.linux.no/linux+v2.6.33/Documentation/filesystems/seq_file.txt

Bank Test Driver

- Tests:
 - Concurrent account adding
 - Concurrent account removal
 - Concurrent withdrawals and deposits
- Uses OpenMP to manage threads

Bank Test Driver

- Concurrent account addition/removal details:
 - Uses multiple threads to add/remove accounts in parallel
 - Specified using:
`#pragma omp parallel for ...`
 - Parallel for construct has threads cooperating to work on different accounts at the same time
- Test passed if:
 - NUM_ACCTS are available in your system and accessible

Bank Test Driver

- Concurrent account withdrawal/deposit details:
 - Specified using:
`#pragma omp parallel sections...`
 - Each block of code in “parallel sections” is executed by a different thread.
 - Each block either withdraws a cent or deposits a cent
- Test passed if:
 - Final result == initial value

Kernel Linked List

- The following is an illustrated guide to understanding the kernel linked lists.

Kernel Linked List Illustrated

```
INIT_LIST_HEAD(&acct_list.list)
```



Kernel Linked List Illustrated

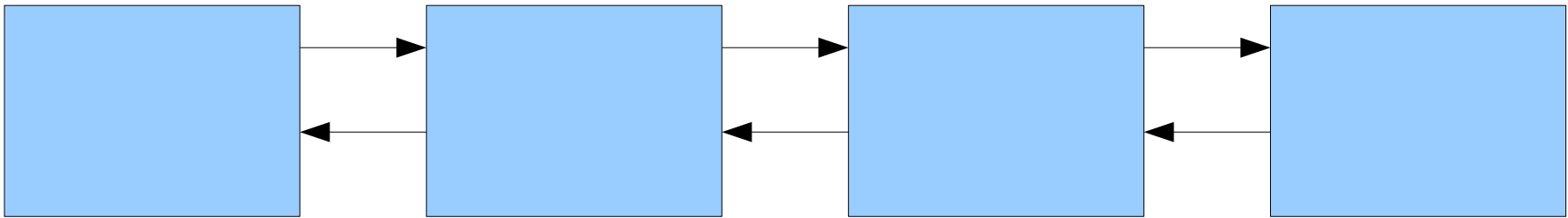
```
INIT_LIST_HEAD(&acct_list.list)
```



List points to self.

Kernel Linked List Illustrated

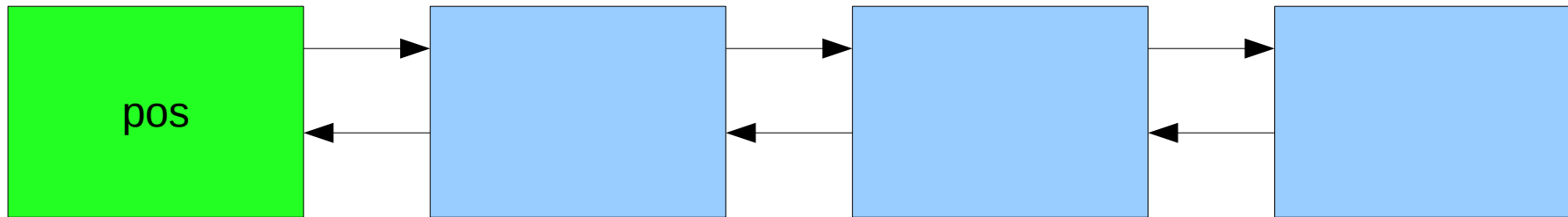
```
list_for_each(pos, &acct_list.list)
```



Kernel Linked List Illustrated

```
list_for_each(pos, &acct_list.list)
```

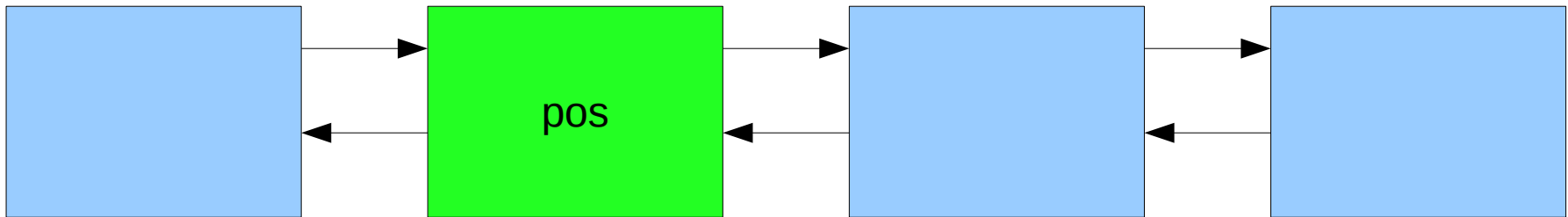
Iteration 0



Kernel Linked List Illustrated

`list_for_each(pos, &acct_list.list)`

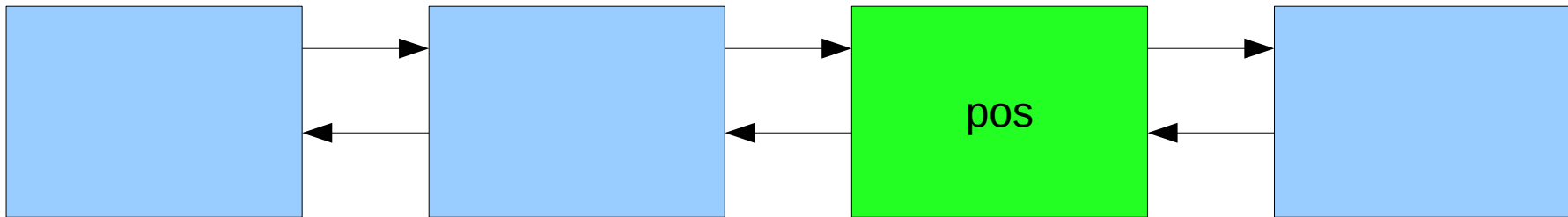
Iteration 1



Kernel Linked List Illustrated

```
list_for_each(pos, &acct_list.list)
```

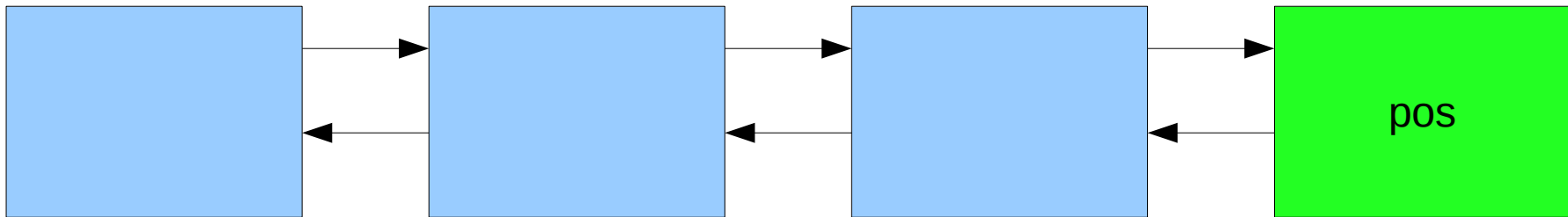
Iteration 2



Kernel Linked List Illustrated

`list_for_each(pos, &acct_list.list)`

Iteration 3



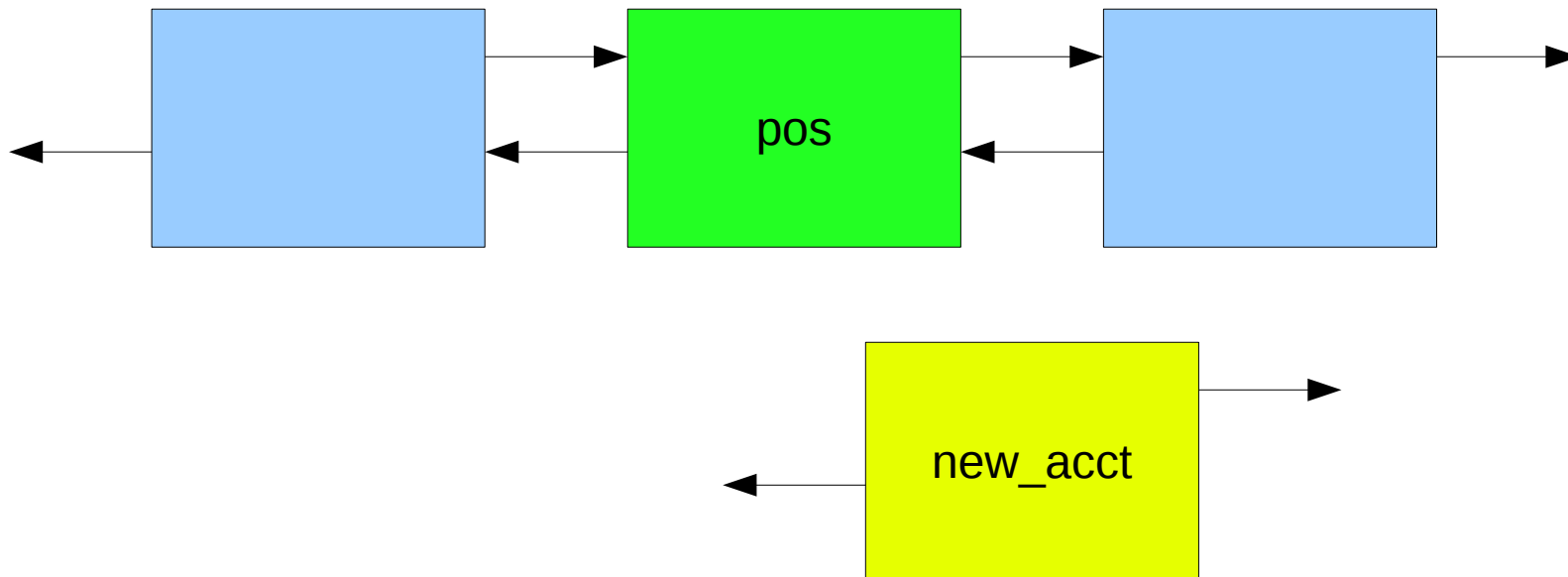
Kernel Linked List Illustrated

```
list_add(new_acct, pos)
```



Kernel Linked List Illustrated

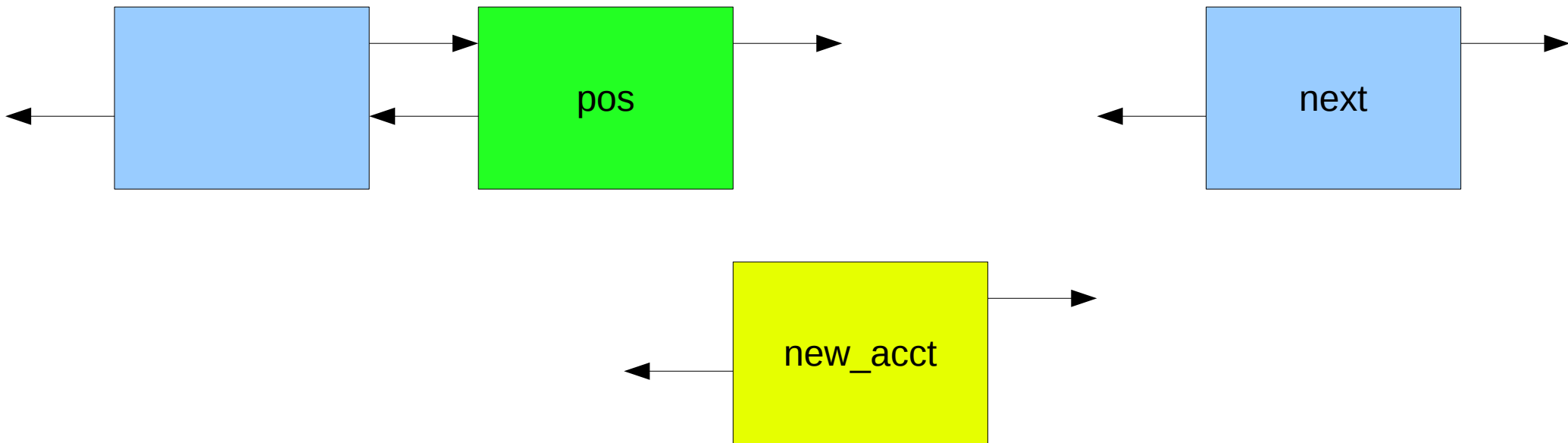
`list_add(new_acct, pos)`



Kernel Linked List Illustrated

```
list_add(new_acct, pos)
```

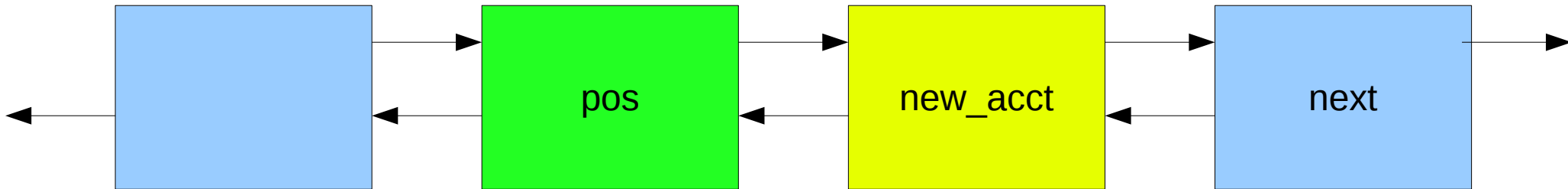
```
__list_add(new_acct, pos, pos->next)
```



Kernel Linked List Illustrated

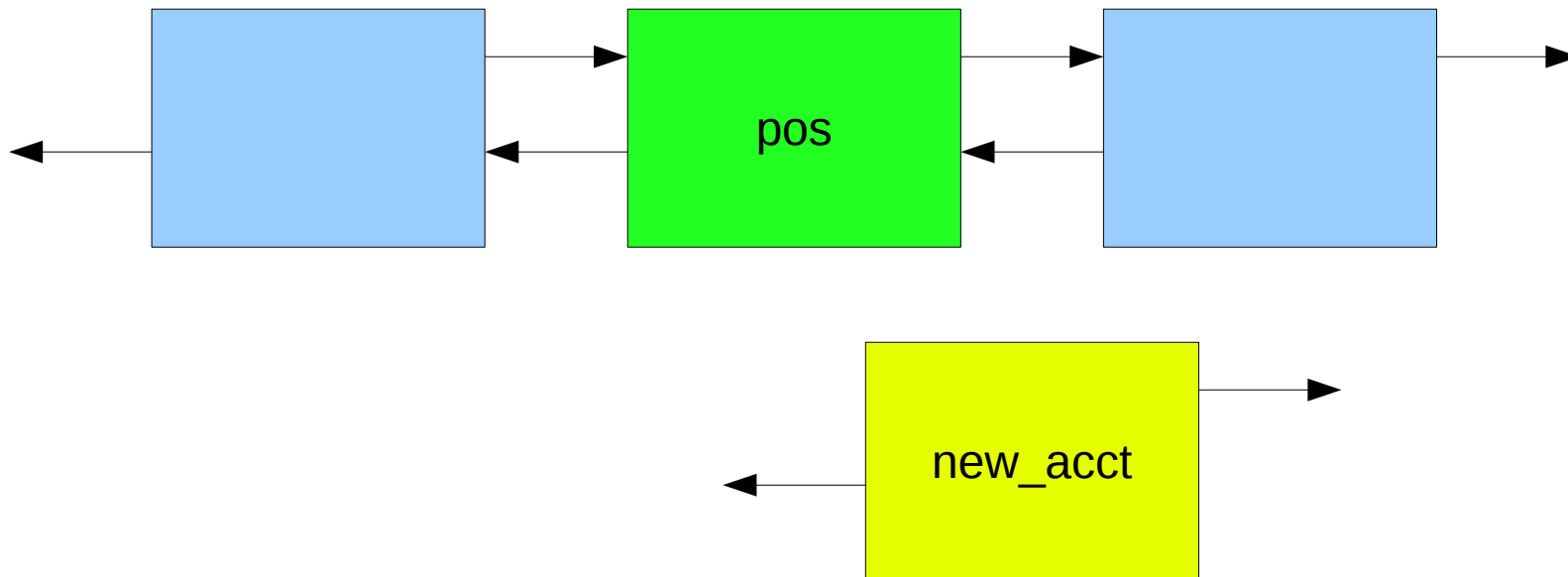
```
list_add(new_acct, pos)
```

```
__list_add(new_acct, pos, pos->next)
```



Kernel Linked List Illustrated

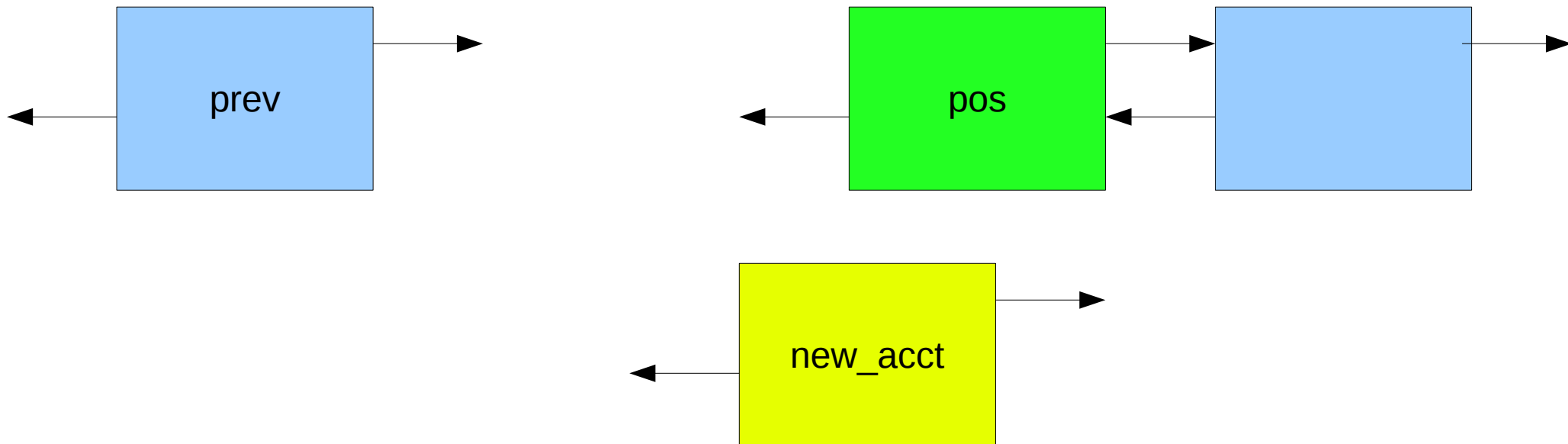
`list_add_tail(new_acct, pos)`



Kernel Linked List Illustrated

```
list_add_tail(new_acct, pos)
```

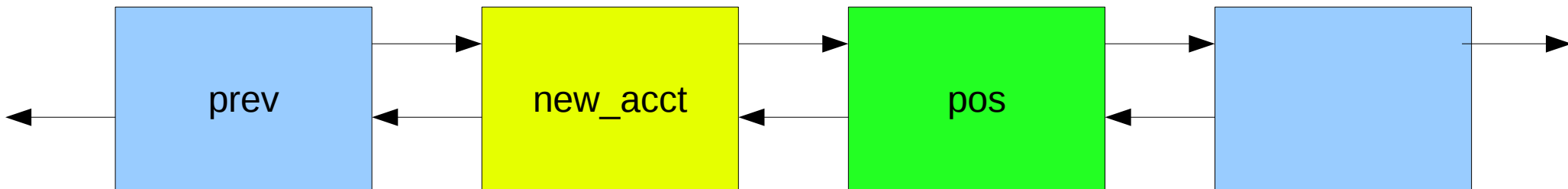
```
__list_add(new_acct, pos->prev, pos)
```



Kernel Linked List Illustrated

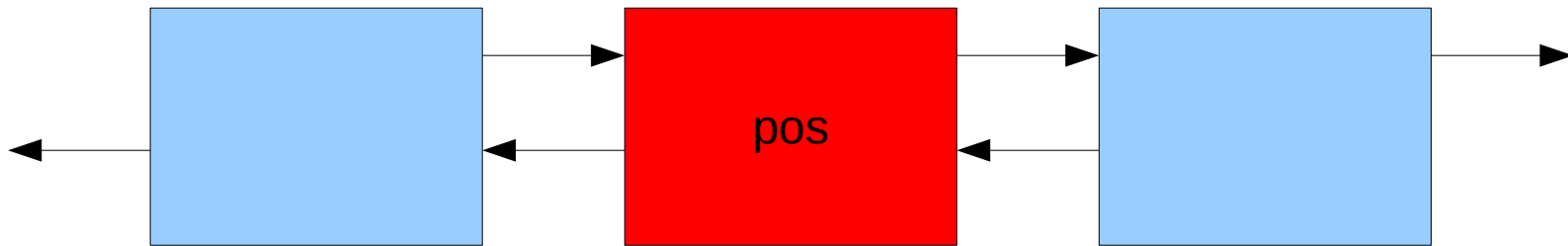
```
list_add_tail(new_acct, pos)
```

```
__list_add(new_acct, pos->prev, pos)
```



Kernel Linked List Illustrated

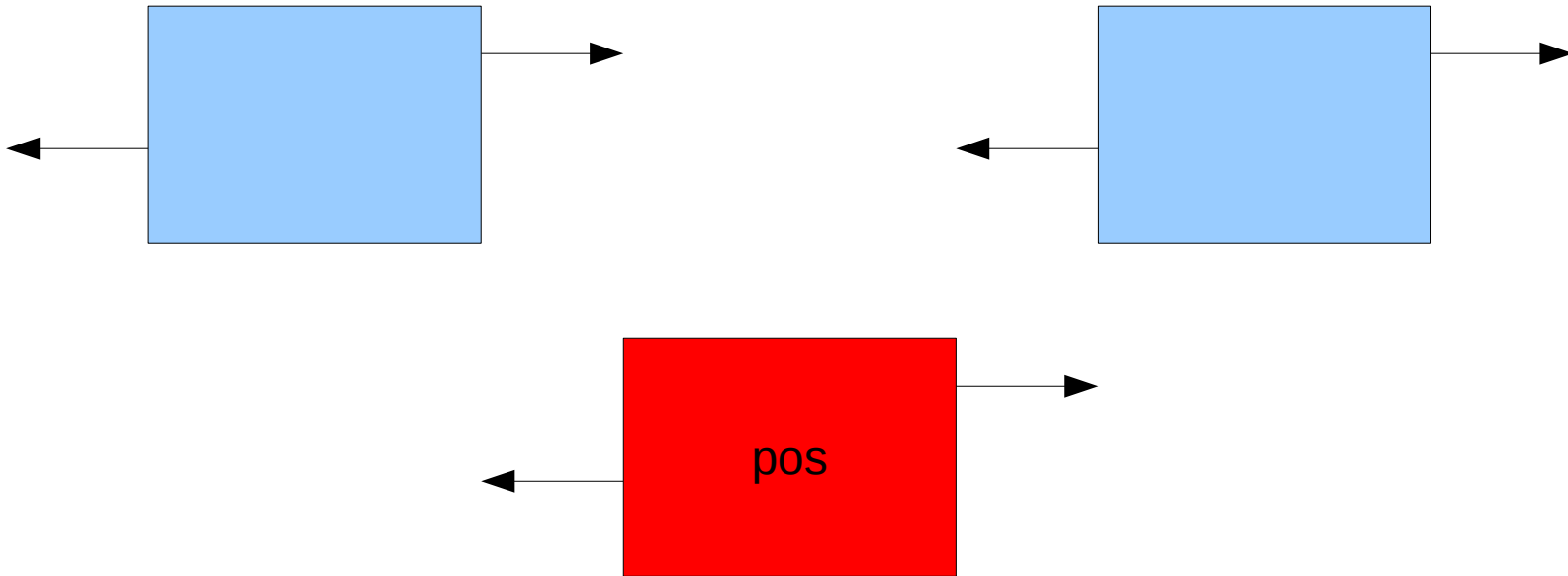
`list_del(pos)`



Kernel Linked List Illustrated

`list_del(pos)`

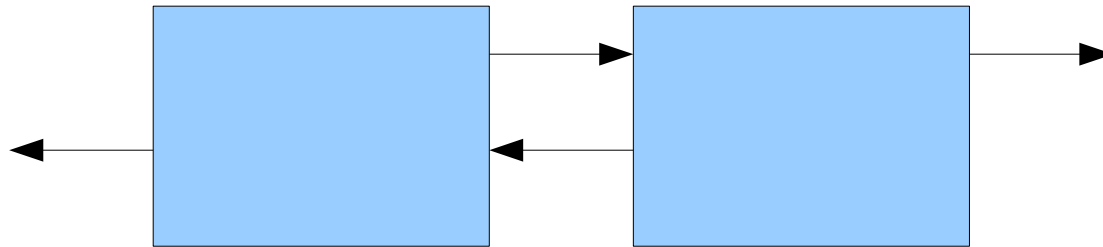
`__list_del(pos->prev, pos->next)`



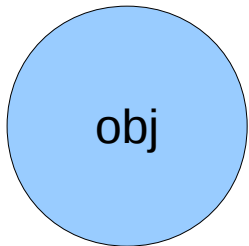
Kernel Linked List Illustrated

`list_del(pos)`

`__list_del(pos->prev, pos->next)`



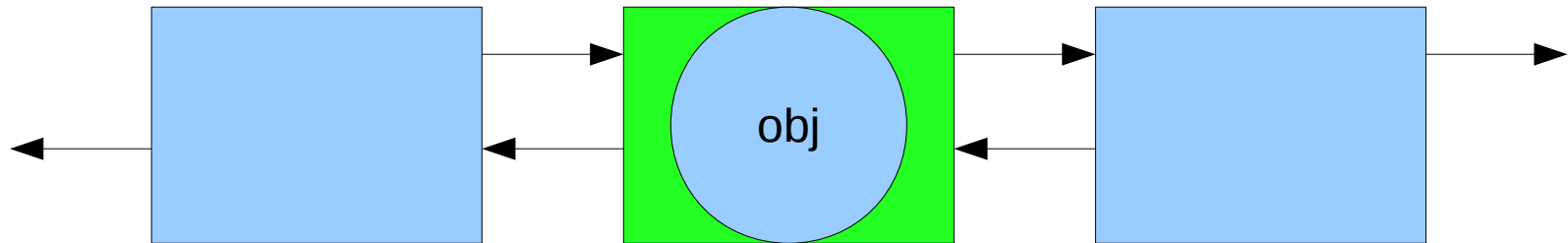
Kernel Linked List Illustrated



```
struct obj{  
    int id;  
    struct list_head list;  
};
```

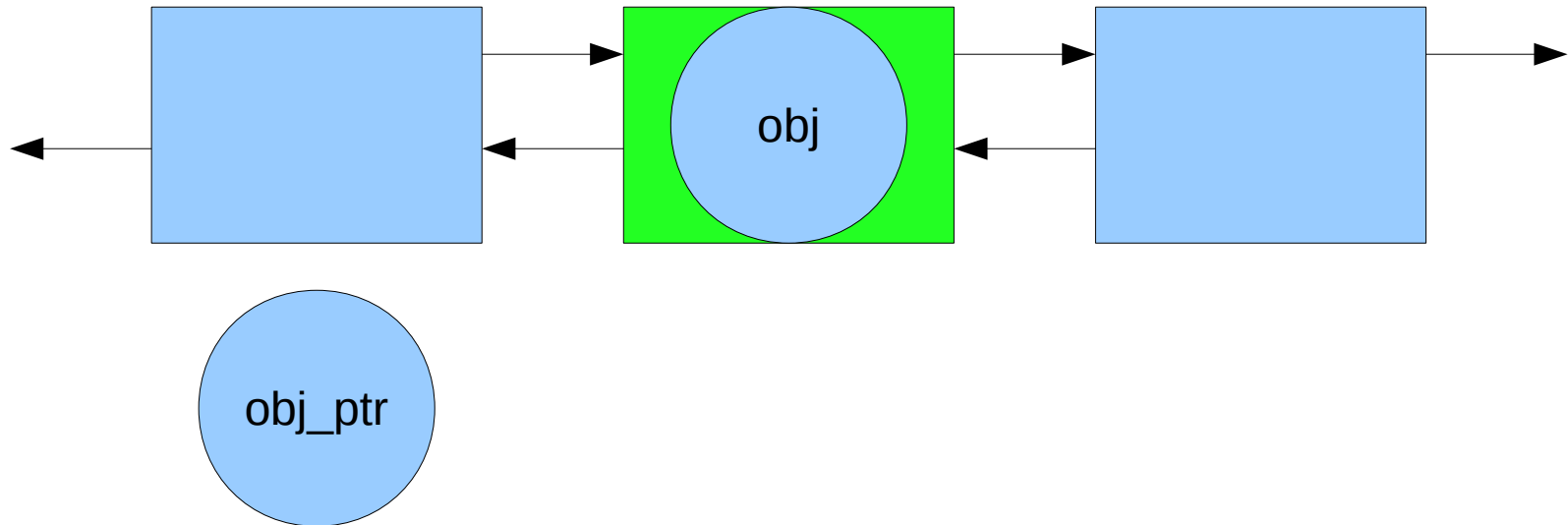
Kernel Linked List Illustrated

```
list_entry(pos, my_obj, list)
```



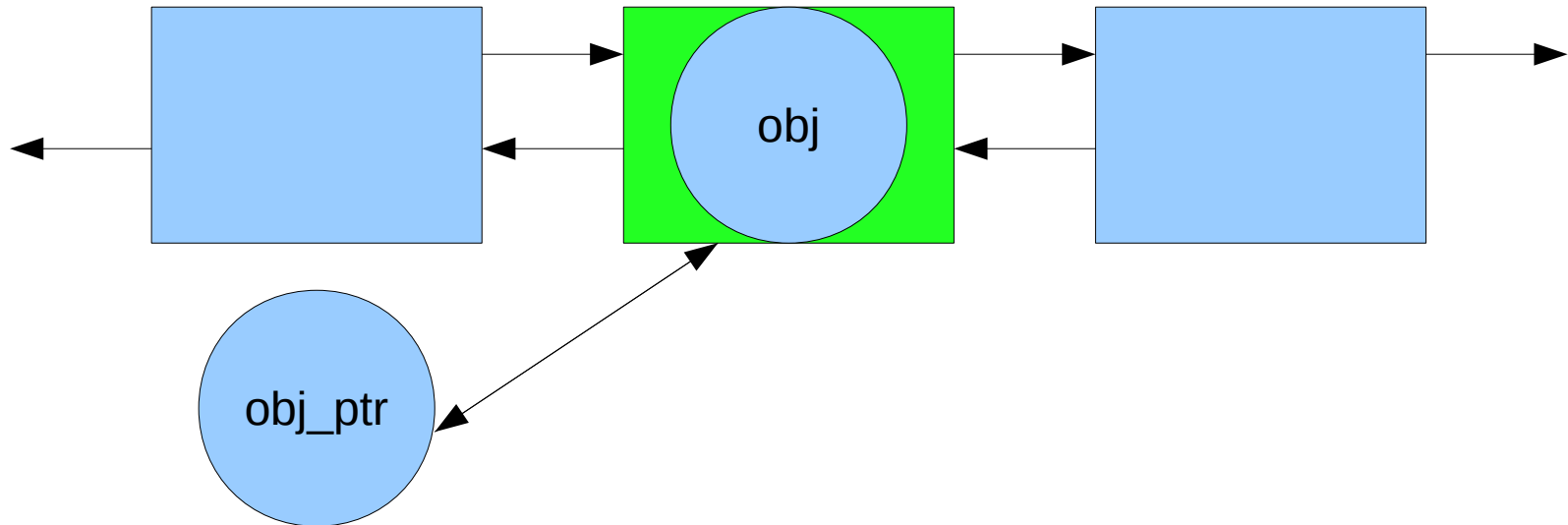
Kernel Linked List Illustrated

`list_entry(pos, my_obj, list)`



Kernel Linked List Illustrated

`list_entry(pos, my_obj, list)`



Next Time:

- Project 3