Computer Programming 2

2024/04/10 Andy Hung

Slide here!

Outline

- HW04 solution
- Midterm Review
  - `malloc`
  - Linked List
  - Function Pointer

HW04

```c
#include <stdio.h>
#include <stdlib.h>

struct Program{
    int start;
    int end;
};
typedef struct Program Program;

int cmp(const void* a, const void* b) {
    Program* programA = (Program*)a;
    Program* programB = (Program*)b;
    return programA->end - programB->end;
}

int main() {
    int N;
    scanf("%d", &N);
```

Midterm Review

`malloc`

```c
// ptr = (cast-type*) malloc(byte-size)

// Create an int array of size 10
int* ptr = malloc(sizeof(int) * 10)
```

- Allocate memory in **heap**.
- Remember to free if you dont't need it anymore.

Midterm Review

Linked List

```c
typedef struct node Node;

struct node {
  int value;
  Node* nextPtr;
}
```

Midterm Review

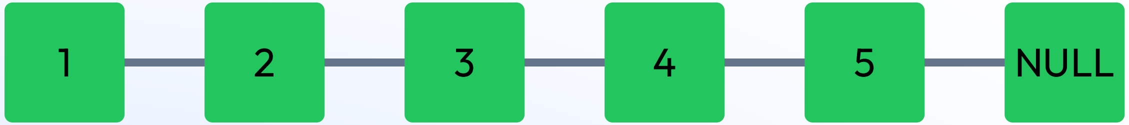
Linked List

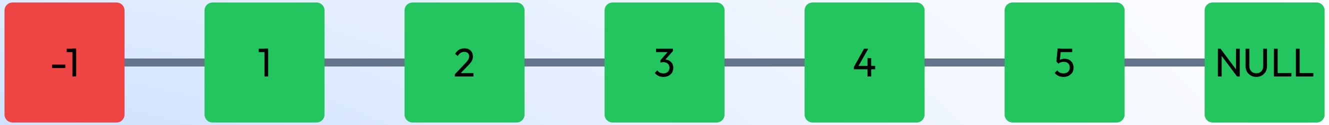
```c
typedef struct node Node;

struct node {
  int value;
  Node* nextPtr;
}
```

- Other Concept
- Insert
- Delete
- Remove

Midterm Review

Linked List - Other Concept

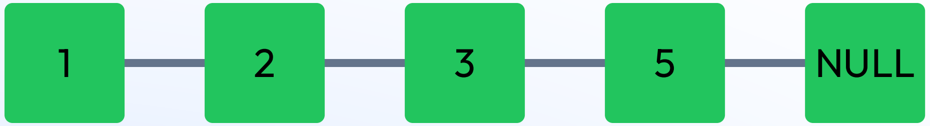


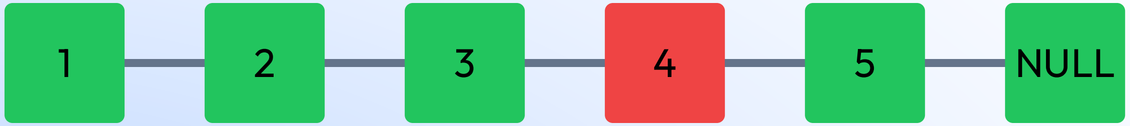


Use a dummy header to avoid strange pointer problem

Midterm Review

Linked List - Insert

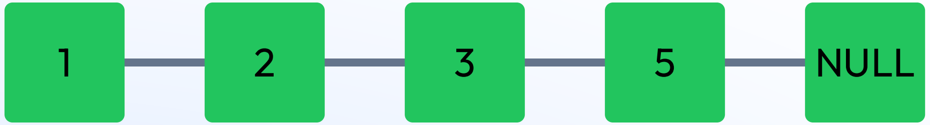


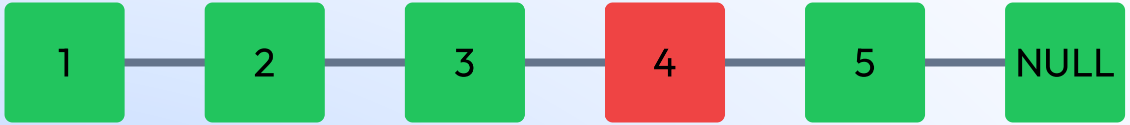


```c
Node* newPtr = malloc(sizeof(Node));
newPtr -> value = 0;
newPtr -> nextPtr = currPtr -> nextPtr;
currPtr -> nextPtr = newPtr;
```

Midterm Review

Linked List - Insert

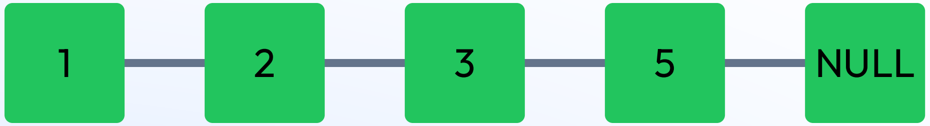


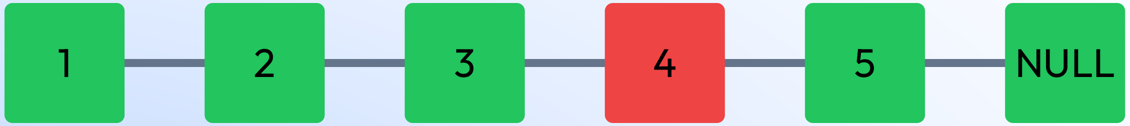


```c
Node* newPtr = malloc(sizeof(Node));
newPtr -> value = 0;
newPtr -> nextPtr = currPtr -> nextPtr;
currPtr -> nextPtr = newPtr;
```

Midterm Review

Linked List - Insert

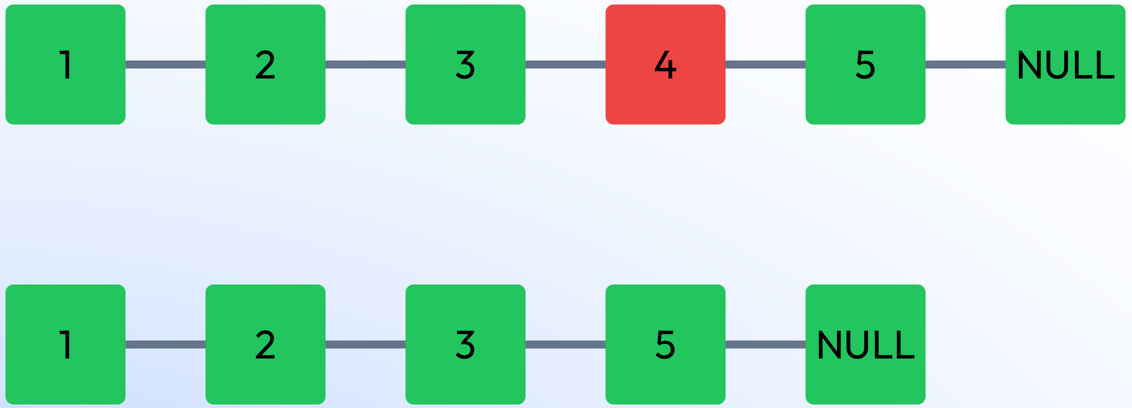




```c
Node* newPtr = malloc(sizeof(Node));
newPtr -> value = 0;
newPtr -> nextPtr = currPtr -> nextPtr;
currPtr -> nextPtr = newPtr;
```

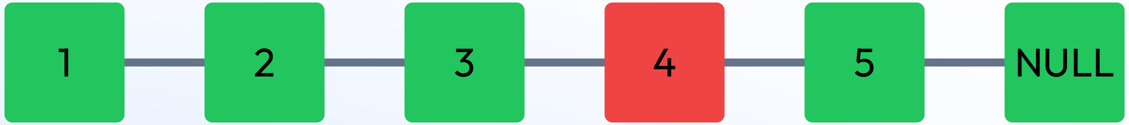
Midterm Review

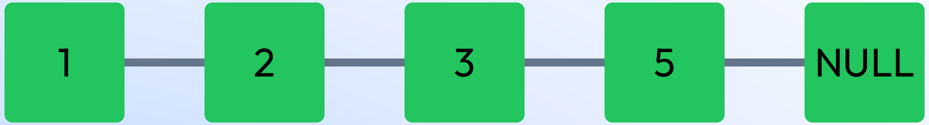
Linked List - Delete



Midterm Review

Linked List - Delete





```
Node* tmpPtr = currPtr -> nextPtr;
currPtr -> nextPtr = tmpPtr -> nextPtr;
free(tmpPtr);
```

Midterm Review

Linked List - Remove

```c
while(ptr != NULL) {
  Node* nextPtr = ptr -> nextPtr;
  free(ptr);
  ptr = nextPtr;
}
```

When using `malloc`, use `free` when not used.

Midterm Review

Function Pointer

```c
struct test {
  void (*test_function)();
};

void this_is_test() {
  printf("Hello world");
}

// ...
test->test_function = this_is_test;
```

An alternative solution of oop in C (very basic version)

Midterm Review

What else

- CP1 and CP2
- Slides
- Labs

Midterm Review

What else

- CP1 and CP2
- Slides
- Labs

Thanks for listening