Managing Memory with Structures

Zachary Bennett
SOFTWARE ENGINEER

@z_bennett_ github.com/zbennett10

Structures

Custom Types

Build your own custom types to better structure programs

Dynamic

Allocate and manage memory for structures on the heap

Manage Memory

Create your own memory pool manager

Structures are user-defined types.

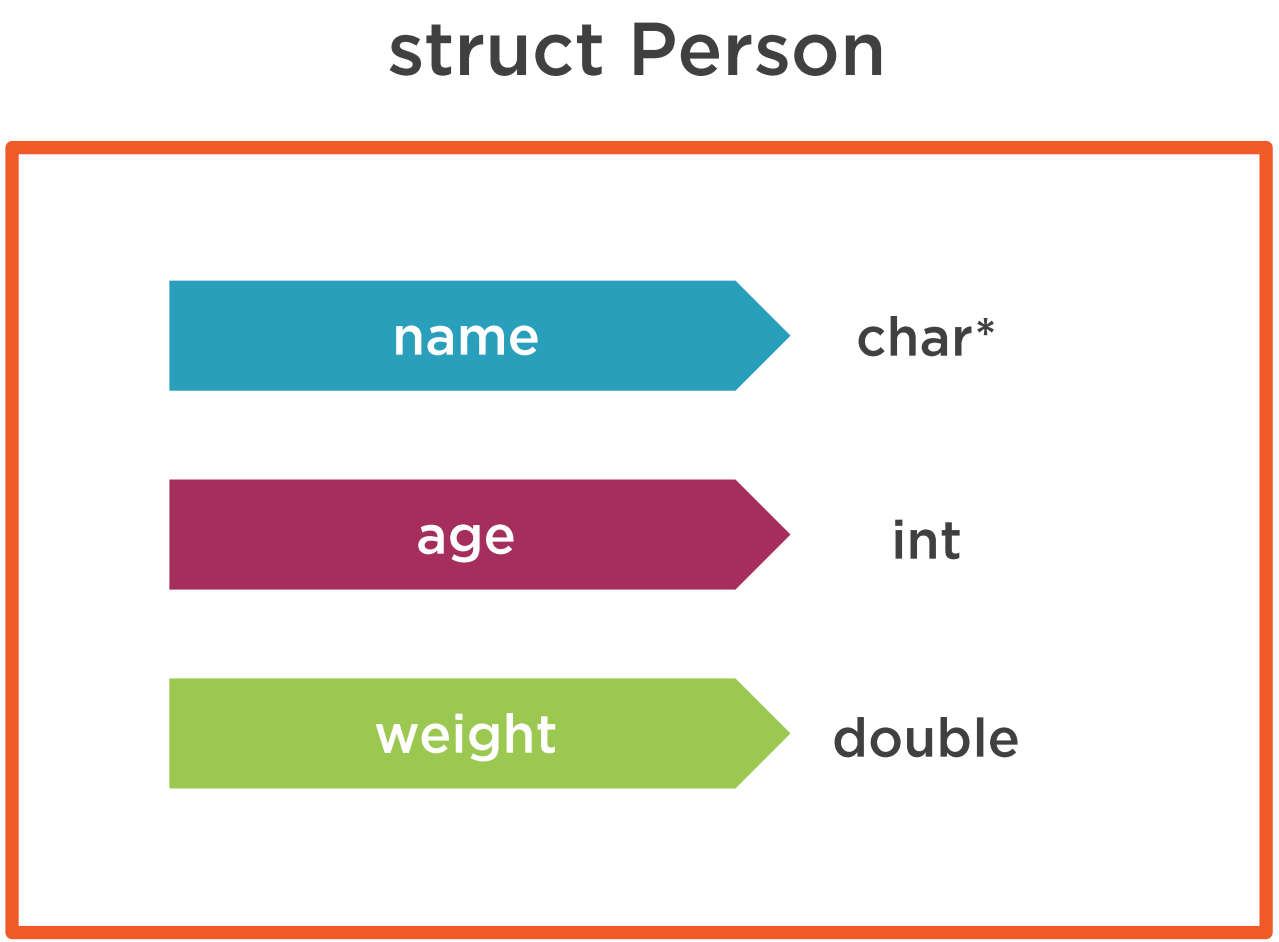
Custom data types

Composite - combines multiple data types

Useful for:

- Constructing objects
- Data hiding
- Writing business logic
- Memory management

Type Composition



Defining a Structure

```
struct Grade {
    char letter;
    int rank;
};
```

```
struct Grade {
    char letter;
    int rank;
} grade1, grade2;
```

```
typedef struct Grade {
    char letter;
    int rank;
} Grade;
```

Structure Initialization and Member Access

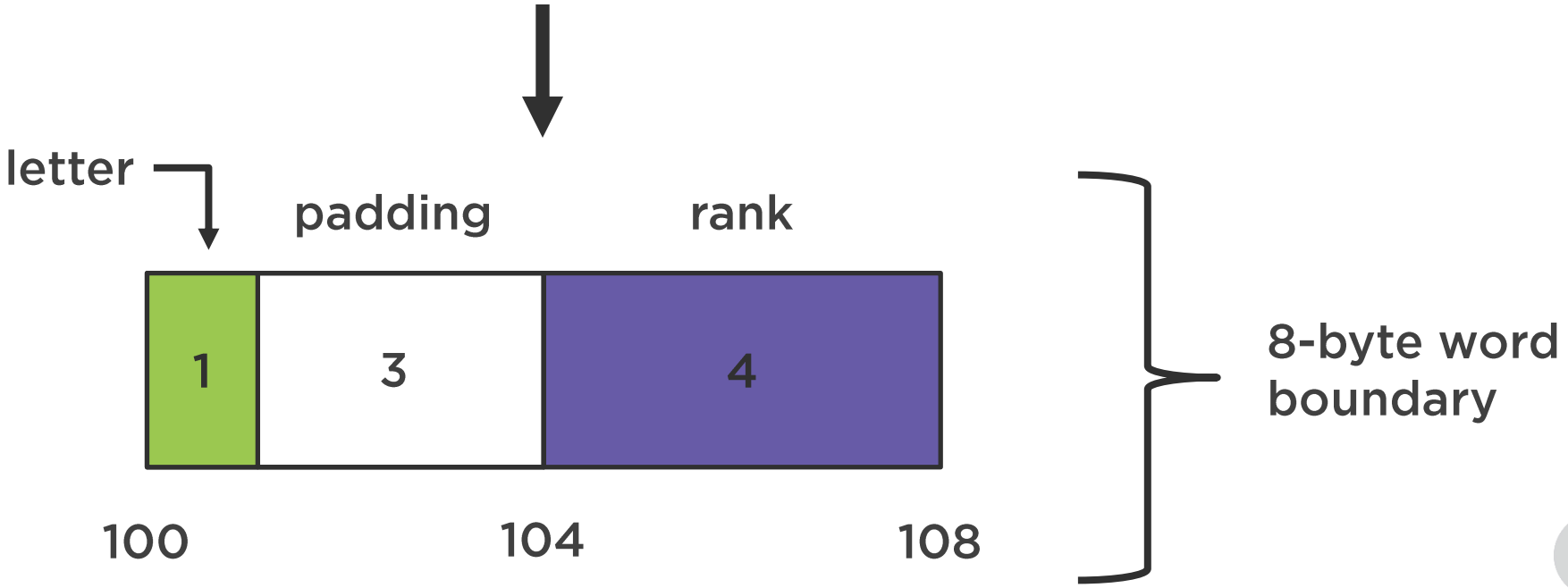
```
Grade student_grade;
student_grade.letter = 'B';
student_grade.rank = 86;
```

```
Grade student_grade = { 'A', 93 };
```

```
Grade student_grade = { .rank = 93, .letter = 'A' };
```

Structure Memory Layout

```
typedef struct Grade {
    char letter;
    int rank;
} Grade;
```



Demo

Declaring structures

Initializing structures

Accessing structure members

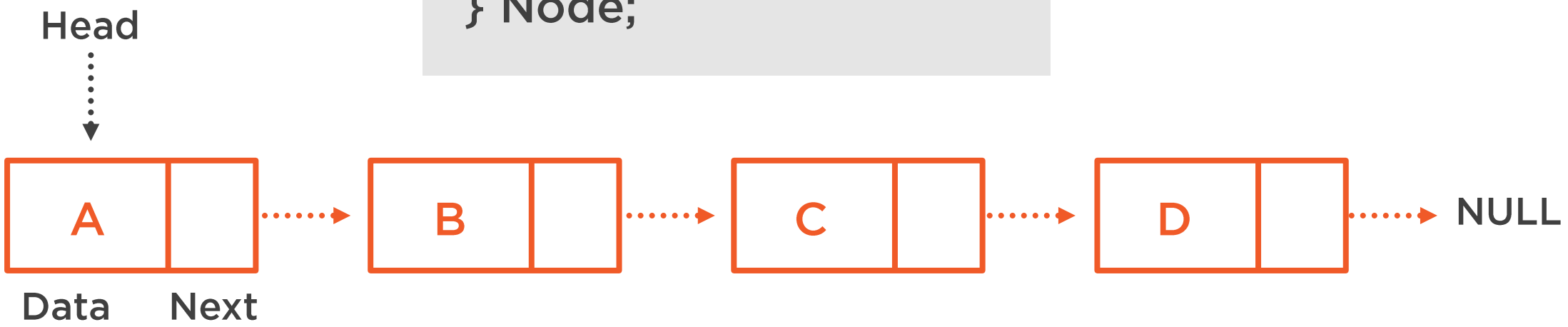
Inspect structure memory layout

Structures + Dynamic Memory Allocation =

Dynamic Data Structures

Dynamically Allocated Structures

```
typedef struct Node {
        void *data;
        struct Node *next;
} Node;
```



Dynamic Structure Syntax

```
Node *head = (Node*)malloc(sizeof(Node));
```

```
head->data
```

```
(*head).data
```

```
head->data = &my_int;
```

```
(*head).data = &my_int;
```

Demo

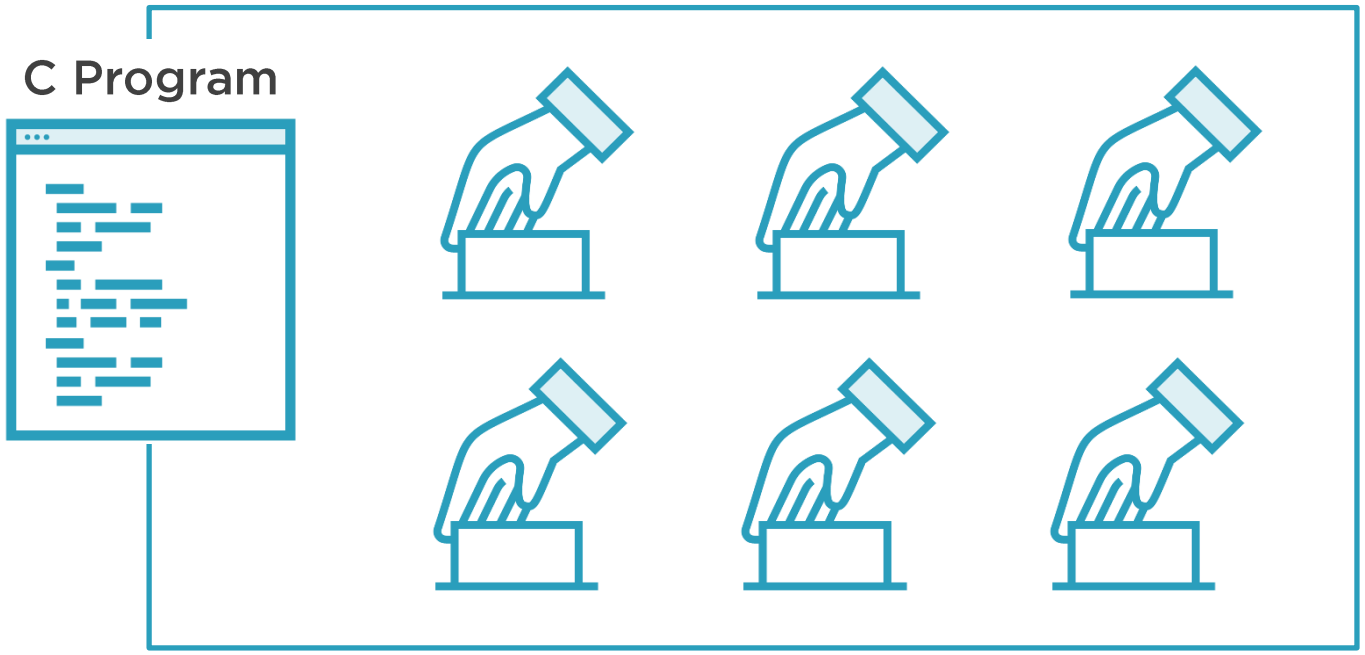
Define types for the linked list dynamic data structure

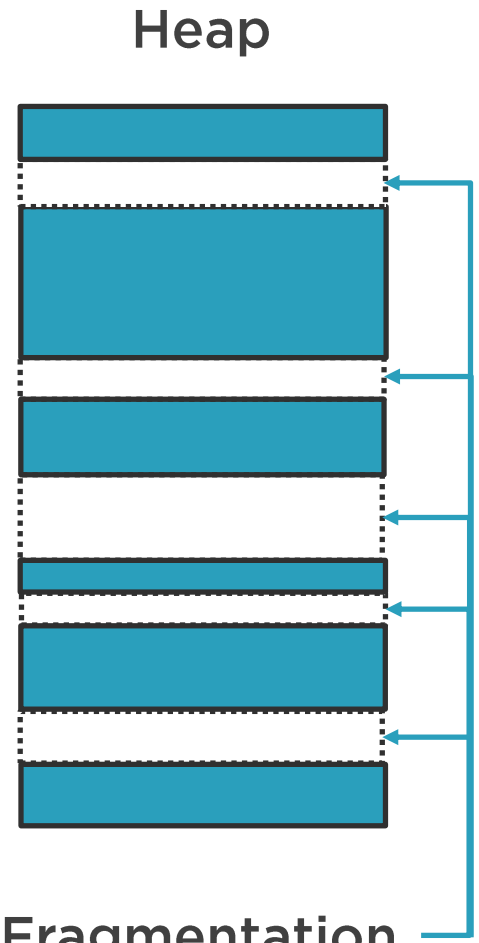
Dynamically allocate and free structure memory

Write and read values to and from structures using pointers

Become familiar with the linked list data structure

Memory Allocation Problems

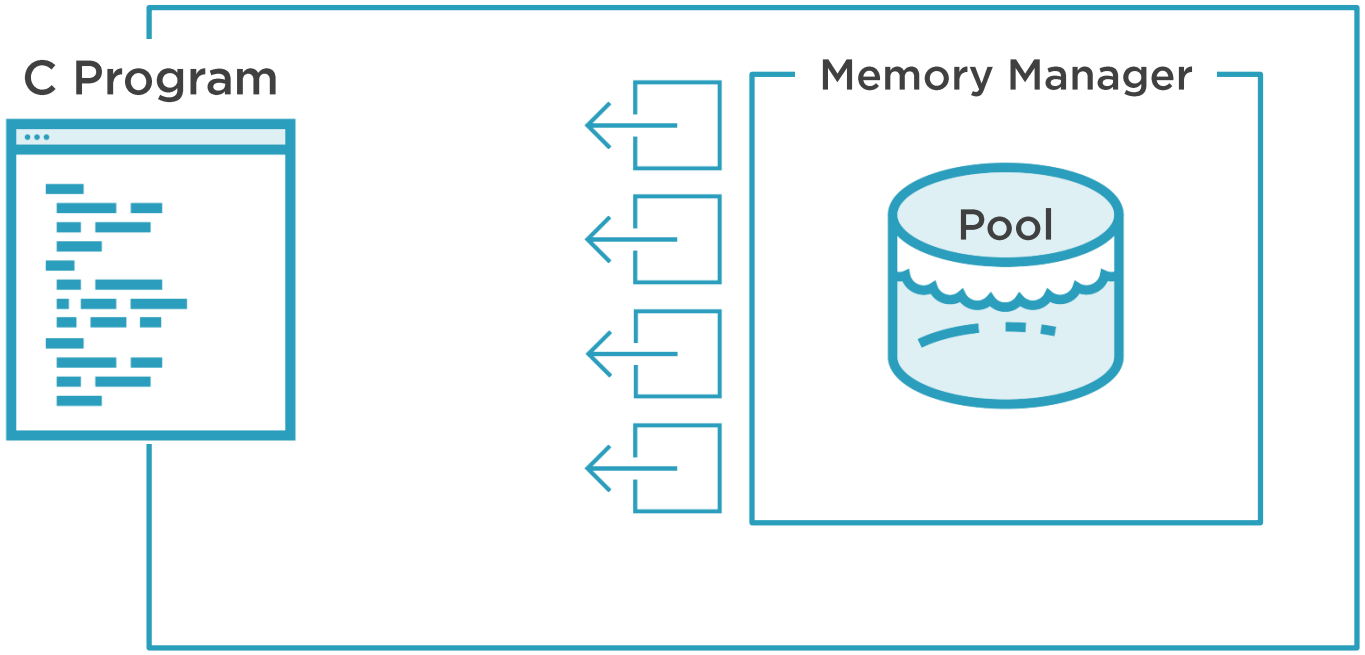


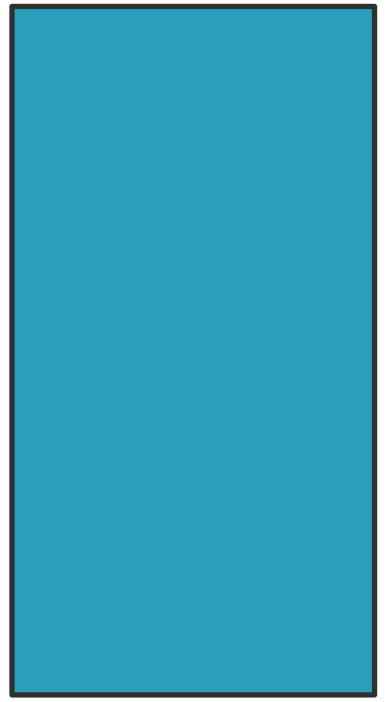


Lots of potentially slow system calls due to malloc

Fragmentation

Memory Pools



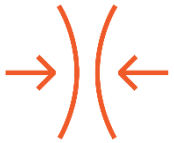


One system call to fetch memory!

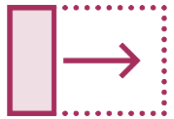
No Fragmentation

Why Memory Pools?

Performance issues

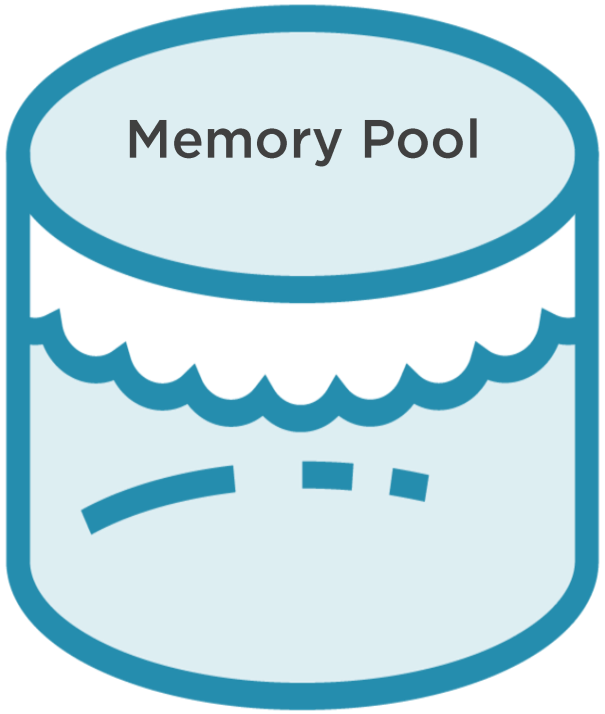


Tighter control over memory allocation



Less fragmentation when allocating lots of objects

Customized memory allocation



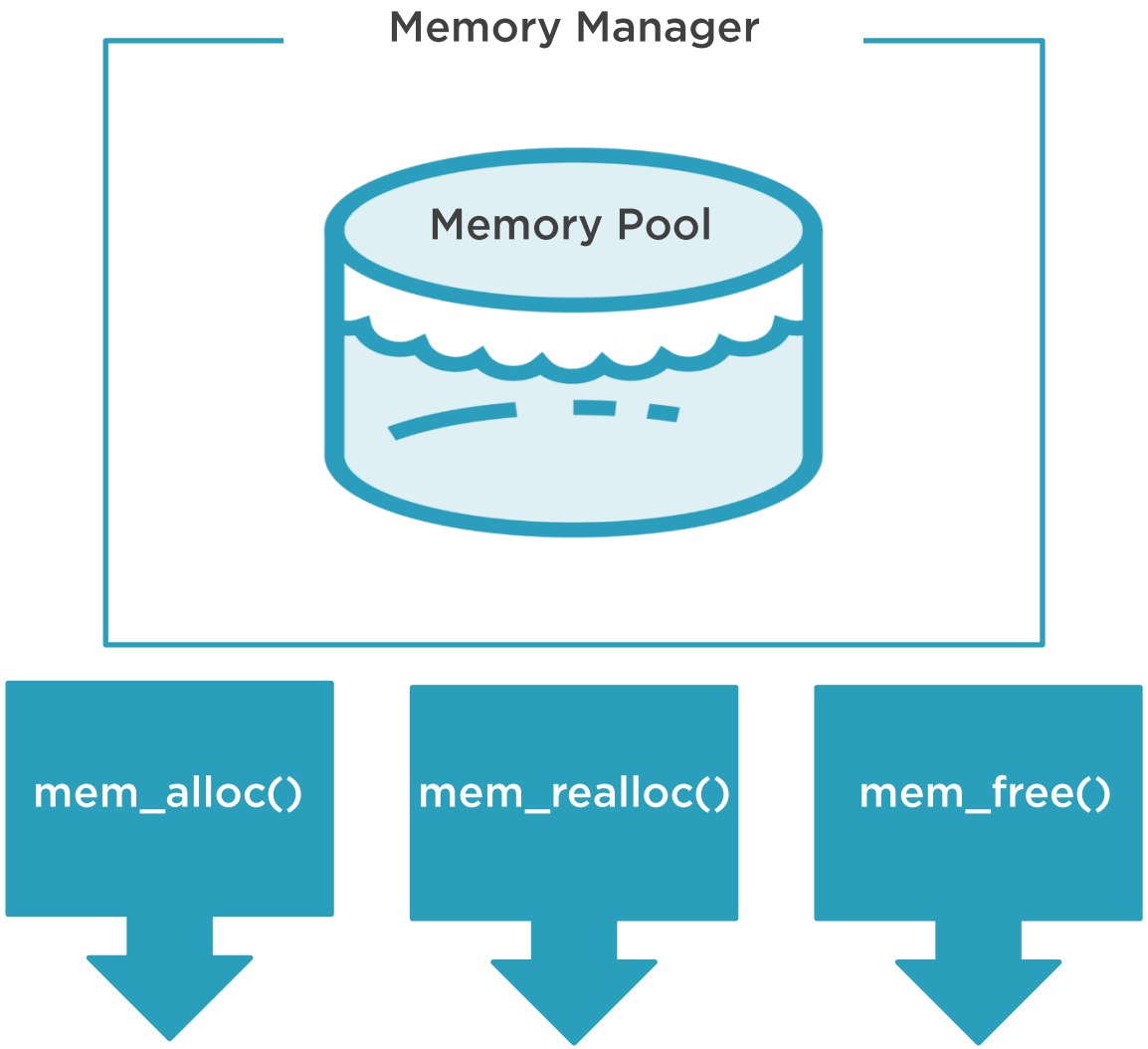
Fixed-size memory pool

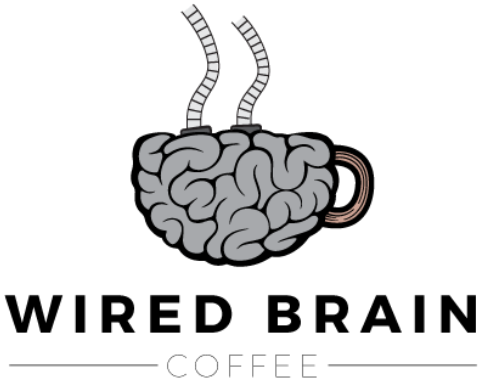
Contains five blocks of memory each of which contain 4,096 bytes

One call to malloc for 20,480 bytes

Can customize this to specific needs of app

Memory Managers





Wired Brain Coffee

Bad Performance - Slow Memory Allocation

Demo

- Diagnose calls to malloc causing slow performance
- Introduce fixed-size memory pool manager
- Replace calls to malloc with custom memory manager

Overview/ Summary

Structures as custom, composite data types

Structure syntax

Dynamically allocated structures

Memory pools and memory managers

Improving performance in Wired Brain Coffee's code base

Wired Brain Coffee bugs!

- Dynamic memory allocation
- Managing memory with pointers, arrays and structures
- Advanced memory management techniques using memory pools

”Thank you!” – Wired Brain Coffee