Structs & Alignment
CSE 351 Winter 2024

Instructor:
Justin Hsia

Teaching Assistants:
Adithi Raghavan
Aman Mohammed
Connie Chen
Eyoel Gebre
Jiawei Huang
Malak Zaki
Naama Amiel
Nathan Khuat
Nikolas McNamee
Pedro Amarante
Will Robertson

https://pixels.com/featured/1-computer-programmer-funny-c-class-joke-noirty-designs.html
Relevant Course Information

❖ HW11 due tonight, HW12 due Monday, HW13 due Wednesday

❖ Lab 2 due tonight
❖ Lab 3 released Monday (2/5) – a shorter lab, due Friday, 2/16

❖ Take-home Midterm (2/8–10)
  ▪ Instructions will be posted on Ed Discussion
  ▪ Gilligan’s Island Rule: discuss high-level concepts and give hints, but not solving the problems together
  ▪ We will be available on Ed Discussion (private posts only) and support hours to answer clarifying questions
Structs & Alignment
Lesson Summary

❖ Alignment

▪ Data of alignment requirement \((i.e., size) K\) is considered aligned if its address is a multiple of \(K\)

▪ Arrays have alignment requirement of an individual element, not the total size

❖ Structures

▪ Allocate bytes for fields in order declared by programmer – can make choices to minimize memory allocations

▪ Pad in middle to satisfy individual element alignment requirements \((K)\)
  • *Internal* fragmentation

▪ Pad at end to satisfy overall struct alignment requirement \((K_{\text{max}})\)
  • *External* fragmentation
Lesson Summary (2/2)

❖ Learning Objectives:
    ▪ Analyze the memory layout of a struct and minimize its impact on program memory usage.
    ▪ Create, access, and modify array and struct elements in C.

❖ What lingering questions do you have from the lesson?
    ▪ Chat with your neighbors about the lesson for a few minutes to come up with questions
Structs & Alignment – Practice
Polling Questions (1/2)

How much space does (in bytes) does an instance of struct `ll_node` take? \[16 \text{ B}\]

Which of the following statements are syntactically valid?

- [✓] n1.next = &n2;
- [✗] n2->data = 351;
- [✓] n1.next->data = 333;
- [✗] (&n2)->next->next.data = 451;
Minimize the size of the struct by re-ordering the fields:

```
struct old {
    int i;
    short s[3];
    char* c;
    float f;
};
```

```
struct new {
    int i;
    float f;
    char* c;
    short s[3];
};
```

- What is the minimum size of `struct new`?

- A. 22 bytes  
- B. 24 bytes  
- C. 28 bytes  
- D. 32 bytes
Homework Setup

❖ Struct in a struct?
  - It’s just another data type, with its own alignment requirement
  - Example:
    ```c
    struct outer {
      char c;
      struct inner {
        short s;
        int i;
      } in;
    };
    ```

  - Syntax:
    ```c
    struct inner {
      short s;
      int i;
    };
    ```

  - New alignment requirement:
    ```c
    struct inner {
      short s;
      int i;
    };
    ```
Struct Pointers

- Pointers store addresses, which all “look” the same
  - **Lab 0 Example**: struct instance Scores could be treated as array of ints of size 4 via pointer casting
  - A struct pointer doesn’t *have* to point to a declared instance of that struct type
- Different struct fields may or may not be meaningful, depending on what the pointer points to
  - This will be important for Lab 5!

```c
long get_a3(struct rec* r) {
    return r->a[3];
}
```

```
movl 12(%rdi), %rax
ret
```

Memory:
```
    "r->a[3]"
```
```
Group Work Time

✧ During this time, you are encouraged to work on the following:
  1) If desired, continue your discussion
  2) Work on the homework problems
  3) Work on the lab (if applicable)

✧ Resources:
  ▪ You can revisit the lesson material
  ▪ Work together in groups and help each other out
  ▪ Course staff will circle around to provide support