Midterm Review

CSE 220: Systems Programming

Ethan Blanton & Carl Alphonce

Department of Computer Science and Engineering

University at Buffalo

Logistics

Your midterm will be in class on Wednesday during lecture.

You will need:

- Yourself
- A writing instrument
- Nothing else

If you are late, you will not be admitted to the room.

The exam is closed book, closed notes.

See Piazza @792

Introduction to CSE 220 and C

- C is a high level language used in systems programming.
- Architectural details are important in C.
- The C/POSIX model is:
  - A dedicated machine for each program
  - Sequential execution of program instructions
  - Data is stored in accessible, addressed memory
- We explored some trivial C programs.

Remember your required readings!

Variables, Strings, and Loops

- C is a typed language
- Every variable has a type
- Variable values must match the type
- Variables have scope, and cannot be used outside that scope
- Arrays are contiguous memory locations
- Array syntax uses []
- C strings are arrays of characters
- Every C string is terminated with a zero byte
- For loop syntax
- For loops are very flexible

Conditionals and Control Flow

- All nonzero values are true conditions in C.
- All Boolean expressions use 1 for true.
- The `bool` keyword holds only 0 or 1.
- C uses short-circuit evaluation of Boolean logic.
- Control flow is implemented with comparisons and jumps.
- Use blocks for if and else!

Memory and Pointers

- Memory locations are identified by addresses.
- Addresses are integers.
- Our system's memory is like one large array.
- POSIX processes appear to have their own dedicated memory.
- Pointers hold addresses and have types.
- Pointers and arrays are closely related, but not the same.

A Tour of Computer Systems

- Architectural details matter
  - Bus widths
  - Numeric properties
  - Performance details
- C and POSIX are just one possible system
- All systems have those details
- Software correctness can be critically important

Integers and Integer Representation

- The CPU and memory deal only in words
- Buses and registers have native word widths
- Integers have different:
  - Bit widths
  - Endianness
  - Sign representation
- Ones' and two's complement representation
- Bits also have to represent fractional values.

Memory Allocation

- The heap is where you manually allocate memory.
- The C standard library contains a flexible allocator.
- Heap allocations are sized by the programmer.
- C does not provide a way to query the size of a heap allocation.

Alignment, Padding, and Packing

- Integers, pointers, and floating point numbers are simple types.
- Arrays and structures are compound types.
- Structures can contain members of mixed type.
- Simple types must be aligned.
- Compound types must align for simple types.
- Allocation normally aligns to the largest requirement.
- Pointer arithmetic uses stride in computations.
- `void *` has a stride of 1.
- The `void *` type can be used for raw memory manipulation
- Casting `void *` to another type is convenient
- Math on `void *` is by byte

Bitwise Operations

- C can manipulate individual bits in memory.
- Bit operations can be subtle and tricky!
- Signedness matters.
- Bit manipulations can force endianness or other representations.

Process Anatomy

- POSIX programs are laid out in sections
- The stack grows downward
- Automatic variables are allocated on the stack
- Stack frames track function calls
- Items removed from the stack are not cleared
- Stack-allocated arguments are how C is call-by-value

License

To retrieve a copy of this material, or related materials, see `https://www.cse.buffalo.edu/~eblanton/`.