Pre-course review

CPU

- CPU fetches instructions from memory and stores processed data back to memory. CPU is made of
  - **Arithmetic Logic Unit (ALU)**: performs all arithmetic and logical operations in a computer
  - **Control Unit**: sends and times control signals. It coordinates instructions and data flow in the CPU
- CPU has 3 instruction cycle
  - **Fetch**: reads instruction to be operated on
  - **Decode**: decodes all instructions fetched from memory
  - **Execute**: executes instructions
- **Program Counter (PC)**: is a register that contains the address of the current instruction being executed
- Call stack maintained by
  - **Stack Pointer (SP)**: register that stores address of the last program request in stack. It always points to the top of the stack
  - **Frame Pointer (FP)**: always points to top of frame aka. top of the stack frame immediately below
- **Register**: holds data
- **Memory Address Register (MAR)**: stores memory address where data will be stored or fetched from

Memory Hierarchy

- Faster/Small Capacity/Expensive ------- Slower/Large Capacity/Cheap
- CPU Registers --- CPU Cache --- DRAM --- HDD

Memory

1. Code
2. Data
3. Stack: grows and shrinks as functions pushes and pops local variables. Stack variable only exist when the function that created them in running. Stack size limited, large array/struct should be stored in heap. Managed automatically. Grows down, executed LIFO
4. Heap: free floating region of memory. Content here can be executed by all functions. Need to manage ie allocate and free memory. Grows up

- 00000000 is address of null

Stack Frame of Function Call

- This goes through example covered in tutorial

C Review

- Pointers
  - &: memory address
  - *: value in address

- Only one value can be returned(can be struct)
    - If variable *passed by value* changes in it will not be reflected outside function
    - If variable *passed by reference* changes in it will be reflected outside function
- Arrays
- Pointer Arithmetic
- Pointers and Structs
- Memory
    - use *malloc* to allocate memory in heap. For kernel *kmalloc*
    - use *free* to free up allocated memory. For kernel *kfree*
    - if memory leak can be debugged using valgrind
- Error Messages
- Boolean Operators
    - &: AND
    - |: OR
    - ~: NOT
    - ^: XOR
- Bit Shifting
    - Right shift (x >> k): move the bits by one k times and add zero to the left side while dropping bits from the right side. Can be thought of as: x * (2^k)
    - Left shift (x << k): move the bits by one k times and add zero to the right side while dropping bits from the left side. Can be thought of as: x * ((1/2)^k)
- Function pointer holds address of a function
    - cleaner code
    - change one function pointer instead of changing all the explicit function call
    - Refer to Code 1

```
// Code 1
// example declaration of function pointer

void (*ptr)();
ptr = Function;
ptr();
```