Chapter 14 Memory API

- The memory allocation interfaces in UNIX systems.

14.1 Types of Memory

- In a running C program there are two types of memory allocated.
  - Stack memory
    - allocations and deallocations are managed explicitely by the compiler for you.
    - Sometimes called automatic memory.
    - Declaring stack memory is as simple as declaring a variable in a program.
    - If you need something to live longer than a call invocation the stack is not where it should go.
  - Heap memory
    - Long lived memory where allocations and deallocations are explicitly handled by the programmer.
    - Presents more challenges to both users and systems.

14.2 malloc()

- The malloc() call is passed a size asking to make room for it on the heap.
  - On success gives a pointer to newly allocated memory.
  - On failure returns a NULL pointer.
- Requires the stdlib.h header file.

14.3 free()

- Used to deallocate memory allocated by malloc.

14.4 Common Errors

- There are a number of common errors when using malloc() and free().
- The following examples run and compile without complaint from teh compiler.
    - **Forgetting to Allocate Memory**
        - Many routine expect memory to be allocated before calling them.
        - strcpy() is an example of one.
            - Will most likely lead to a segmentation fault.
    - **Not Allocating Enough Memory**
        - Not allocating enough memory results in a buffer overflow.
        - In some cases the program may run fine, but leads to the existence of vulnerabilities.
    - **Forgetting to Initialize Allocated Memory**
        - If this happens the program will encounter an uninitialized read.
            - reads arbitrary data from the heap.
    - **Forgetting To Free Memory**
        - Known as a memory leak
        - Slowly leaking memory leads to running out of memory.
            - When you run out of memory the computer has to be restarted.
    - **Freeing Memory Before You Are Done With It**
        - Called a dangling pointer.
        - can crash the program or overwrite valid memory.
    - **Freeing Memory Repeatedly**
        - Called double free

        - The result is undefined.
    - **Calling free() Incorrectly**
        - If passed a value other than a pointer allocated with malloc bad things happen.

14.5 Underlying OS Support

- malloc() and free() are library calls not system calls.
- The malloc library manages space within virtual address space and built on top of system calls.
- System calls:
    - brk: used to change thel location of a programs break.
        - break is the location at the end of the heap.
        - Increases/decreases the size of the heap.
        - brk/sbrk should never be called directly.
    - mmap(): used to obtain memory from the OS directly.
        - using mmap can create anonymous memory in a program.
            - a region not associated with any file but is swap space.