Principles of Computer Systems

4: Signals, Inter-Process Communication

- Processes give us protection
- Processes are isolated
- Separate address spaces in memory

Signals
- Remember kill?
- SIGTERM
- Ever asked yourself what happens when you press Ctrl-c?
- SIGINT
- Signals = Software Interrupts
- Signals notify a process of an event
  - OS receives event
  - Process is stopped immediately, signal is set
  - Signal handler in the process is called (if registered, else default handler)
  - Signal handler runs to completion
  - Process resumed where it was interrupted (or was terminated)

Signals
- Three different categories of signals:
  - Errors
    - Division by zero (SIGFPE), accessing an invalid memory address (SIGSEGV)
  - External Events
    - (asynchronous) I/O available (SIGIO), child has stopped (SIGCHLD)
  - Explicit Requests
    - kill the process (SIGKILL), interrupt process (SIGINT), etc.
    - SIGUSR1, SIGUSR2

- Signals can be synchronous or asynchronous
- (some) signals can be blocked (deferred) by the process
Signals

- Define a handler
  ```c
  void sighup_handler()
  {
      printf("the program has received a SIGHUP signal\n");
  }
  ```

- Register the handler
  ```c
  signal(SIGUP, sighup_handler);
  ```

- Block signal
  ```c
  int sigprocmask (int how, const sigset_t *set, sigset_t *oldset)
  ```

- When entering handler, the causing signal is automatically blocked