Systems Programming Reference Card

```
int kill(pid_t pid, int num)
#define _POSIX_SOURCE
#include <sys/types.h>
#include <signal.h>
```

```
int mkfifo(const char *name, mode_t mode)
#include <sys/stat.h>
```

```
int pause(void)
#include <unistd.h>
```

```
int pipe(int fd[2])
#include <unistd.h>
```

```
int pthread_cond_broadcast(pthread_cond_t *cv)
#include <pthread.h>
```

```
int pthread_cond_destroy(pthread_cond_t *cv)
#include <pthread.h>
```

```
int pthread_cond_init(pthread_cond_t *cv, const pthread_condattr_t *attr)
#include <pthread.h>
```

```
int pthread_cond_signal(pthread_cond_t *cv)
#include <pthread.h>
```

```
pthread_cond_t example = PTHREAD_COND_INITIALIZER
#include <pthread.h>
```

```
int pthread_cond_wait(pthread_cond_t *cv, pthread_mutex_t *mutex)
#include <pthread.h>
```

```
int pthread_create(pthread_t *thread, const pthread_attr_t *attr,
                   void *(*start)(void *), void *arg)
#include <pthread.h>
```

```
int pthread_detach(pthread_t thread)
#include <pthread.h>
```

```
int pthread_equal(pthread_t a, pthread_t b)
#include <pthread.h>
```

```
void pthread_exit(void *retval)
#include <pthread.h>
```

```
void *pthread_getspecific(pthread_key_t key)
#include <pthread.h>

int pthread_join(pthread_t thread, void **retval)
#include <pthread.h>

int pthread_key_create(pthread_key_t *key, void (*destructor)(void *)
#include <pthread.h>

int pthread_mutex_destroy(pthread_mutex_t *mutex)
#include <pthread.h>
-
int pthread_mutex_init(pthread_mutex_t *mutex,
                       const pthread_mutexattr_t *attr)
#include <pthread.h>

int pthread_mutex_lock(pthread_mutex_t *mutex)
#include <pthread.h>

pthread_mutex_t example = PTHREAD_MUTEX_INITIALIZER
#include <pthread.h>

int pthread_mutex_unlock(pthread_mutex_t *mutex)
#include <pthread.h>

int pthread_once(pthread_once_t *control, void (*init)(void))
#include <pthread.h>

pthread_t pthread_self(void)
#include <pthread.h>

int pthread_setspecific(pthread_key_t key, const void *value)
#include <pthread.h>

int sem_close(sem_t *sem)
#include <semaphore.h>

sem_t *sem_open(const char *name, int oflag, /*mode_t mode, unsigned value*/)
#include <semaphore.h>

int sem_post(sem_t *sem)
#include <semaphore.h>

int sem_trywait(sem_t *sem)
#include <semaphore.h>

int sem_unlink(const char *name)
#include <semaphore.h>
```

```
int sem_wait(sem_t *sem)
#include <semaphore.h>

int sigaction(int num, const struct sigaction *act, struct sigaction *oact)
#include <signal.h>

int sigaddset(sigset_t *set, int signo)
#include <signal.h>

int sigdelset(sigset_t *set, int signo)
#include <signal.h>

int sigemptyset(sigset_t *set)
#include <signal.h>

int sigfillset(sigset_t *set)
#include <signal.h>

void ( *signal(int num, void (*handler)(int)) ) (int)
#include <signal.h>

typedef void (*signalhandler_t)(int)
sighandler_t signal(int num, sighandler_t handler)
#include <signal.h>

int sigprocmask(int how, const sigset_t *set, sigset_t *oldest)
#include <signal.h>

int sigsuspend(const sigset_t *set)
#include <signal.h>

int unlink(const char *name)
#include <unistd.h>
```