Systems Programming Reference Card

int atexit(void (*func)(void))
```
#include <stdlib.h>
```

int close(int fd)
```
#include <unistd.h>
```

int dup(int oldfd)
```
#include <unistd.h>
```

int dup2(int oldfd, int newfd)
```
#include <unistd.h>
```

int execve(const char *path, char *const argv[], char *const envp[])
```
#include <unistd.h>
```

void exit(int status)
```
#include <stdlib.h>
```

FILE *fdopen(int fd, const char *mode)
```
#include <stdio.h>
```

pid_t fork(void)
```
#include <unistd.h>
#include <sys/types.h>
```

char *getenv(const char *name)
```
#include <stdlib.h>
```

pid_t getpgrp(void)
```
#include <unistd.h>
```

pid_t getpid(void)
```
#include <sys/types.h>
#include <unistd.h>
```

pid_t getppid(void)
```
#include <unistd.h>
#include <sys/types.h>
```

int isatty(int fd)
```
#include <unistd.h>
```

mode_t

`S_IRUSR` (read by owner), `S_IWUSR` (write by owner), `S_IROTH` (read by others), `S_IWOTH` (write by others)

int open(char *pathname, int flags, mode_t mode)
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
flags: O_RDONLY, O_WRONLY

void perror(const char *prefix)
#include <stdio.h>

ssize_t read(int fd, void *buffer, size_t count)
#include <unistd.h>

int setenv(const char *name, const char* value, int overwrite)
#include <stdlib.h>

int setpgid(pid_t pid, pid_t pgid)
#include <sys/types.h>
#include <unistd.h>

char *strerror(int n)
#include <string.h>

int tcsetattr(int fd, int options, const struct termios *attr)
#include <termios.h>
#include <unistd.h>

char *ttyname(int fd)
#include <unistd.h>

int unsetenv(const char *name)
#include <stdlib.h>

pid_t wait(int *status)
#include <sys/types.h>
#include <sys/wait.h>

pid_t waitpid(pid_t pid, int *status, int options)
#include <sys/types.h>
#include <sys/wait.h>

ssize_t write(int fd, const void *buffer, size_t count)
#include <unistd.h>