

CSC209 Section Fall 2002: Aid Sheet

C function prototypes and structs:

```
int chdir(const char *path);
int closedir(DIR *dir);
int execlp(const char *file, const char *arg, ...);
int execvp(const char *file, char *const argv[]);
int fclose(FILE *stream);
char *fgets(char *s, int n, FILE *stream);
FILE *fopen(const char *file, const char *mode);
pid_t fork(void);
int fprintf(FILE *stream, const char *format, ...);
char *getcwd(char *buf, size_t size);
char *getenv(const char *name);
int mkdir(const char *pathname, mode_t mode);
DIR *opendir(const char *name);
void perror(const char *s);
struct dirent *readdir(DIR *dir);
int sprintf(char *s, const char *format, ...)
int stat(const char *file_name, struct stat *buf);
char *strncat(char *dest, const char *src, size_t n);
int strncmp(const char *s1, const char *s2, size_t n);
char *strncpy(char *dest, const char *src, size_t n);
pid_t wait(int *status)
pid_t waitpid(pid_t pid, int *status, int options);

struct dirent {
    long d_ino;           /* inode number */
    off_t d_off;         /* offset to this dirent */
    unsigned short d_reclen; /* length of this d_name */
    char d_name [NAME_MAX+1]; /* file name (null-terminated) */
}

struct stat {
    dev_t      st_dev;      /* device */
    ino_t      st_ino;      /* inode */
    mode_t     st_mode;     /* protection */
    nlink_t    st_nlink;    /* number of hard links */
    uid_t      st_uid;      /* user ID of owner */
    gid_t      st_gid;      /* group ID of owner */
    dev_t      st_rdev;     /* device type (if inode device) */
    off_t      st_size;     /* total size, in bytes */
    unsigned long st_blksize; /* blocksize for filesystem I/O */
    unsigned long st_blocks; /* number of blocks allocated */
    time_t     st_atime;    /* time of last access */
    time_t     st_mtime;    /* time of last modification */
    time_t     st_ctime;    /* time of last change */
};
```

The following POSIX macros are defined to check the file type (m is the st_mode field of the stat struct):

```
S_ISLNK(m)  is it a symbolic link?
S_ISREG(m)  regular file?
S_ISDIR(m)  directory?
```

CSC209 Section Fall 2002: Aid Sheet

Shell and Perl comparison operators

Shell	Perl	Description
-d filename	-d filename	Exists as a directory
-f filename	-f filename	Exists as a regular file.
-r filename	-r filename	Exists as a readable file
-w filename	-w filename	Exists as a writable file.
-x filename	-x filename	Exists as an executable file.
-z string	string eq ""	True if empty string
str1 = str2	str1 eq str2	True if str1 equals str2
str1 != str2	str1 ne str2	True if str1 not equal to str2
int1 -eq int2	int1 == int2	True if int1 equals int2
-ne, -gt, -lt, -le	!=, >, >=, <, <=	For numbers
!=, >, >=, <, <=	ne, gt, lt, le	For strings
-a, -o	&&,	And, or.

Perl functions:

push(ARRAY, LIST)
 pop(ARRAY) -- returns LIST
 sort BLOCK LIST -- returns LIST
 defined(SCALAR) -- returns true if SCALAR is defined, false otherwise
 split(/PATTERN/, SCALAR) - returns LIST
 open(FILEHANDLE, EXPR) -- return true if filename given by EXPR is opened

Perl pattern matching:

\s = [\t\n\r\f]	space
\w = [a-zA-Z0-9_]	word
\d = [0-9]	digit
[]	one character of a set
[^]	any expect one of the set
+	one or more
*	zero or more
?	zero or one
.	any character

Meta characters that need to be escaped are \ | () [{ . \$ ^ ? +

Perl Special variables

@_	arguments to a subroutine
\$_	the default scalar variable
\$.	the current input line number
\$0	program name
\$\$	process id
#!	last system call error
@ARGV	command line arguments