# Lab 9 Slides

# Name-of-System-Call. Typical layout.

#include<Name-of-Needed-Include-File>

**void System-Call(type argument-passed-in);**

**This is the function prototype that is listed in the above include file. You don't need to type it in your code.**

Definition of the System Call

*Example*: of the call used in a line of code

# EXIT

```
#include<stdlib.h>

void exit(int status);
```

The exit call used by most programs including the parent.

These are defined in stdlib.h:

```
#define EXIT_SUCCESS 0
#define EXIT_FAILURE 1
```

***Example*: exit(EXIT_SUCCESS);**

# GETCWD

#include<unistd.h>

**char getcwd(char **cwdbuf*, size_t *size*);**

This is the function prototype and is embedded in the #include file.

Returns *cwdbuf* on success, or NULL on end.

A process can retrieve its current working directory using *getcwd()*

*Example:* **getcwd(path, length_of_array_for_path);**

# GETENV

#include<stdlib.h>

char *getenv(const char *name);

This is the function prototype and is embedded in the #include file.

Returns pointer to (value) string, or NULL if no such variable.

The *getenv()* function retrieves individual values from the process environment.

*Example*: **dir = getenv("HOME");**

# HOME

This is the initial directory into which the user is placed after logging in.

This field becomes the value of the HOME environment variable.

Faculty example: /gaia/home/faculty/bielr

Student example: /gaia/class/student/smithj

# CHDIR

```
#include<unistd.h>

int chdir(const char *pathame);
```
This is the function prototype and is embedded in the #include file.

Returns 0 on success, or -1 on error.

The *chdir()* system call changes the calling process's current working directory to the relative or absolute pathname specified in *pathname*.

*Example*: if (**chdir(dir) !=0)**

**perror ("Error changing directory\n");**

# Dealing with Errors

You have at least two choices:

- *Use a fprintf. Good for checking non-system errors.*
- *Example:* **fprintf(stderr, "No command \n");**

- *Use perror function. Good for system-call error checking.*
- *Example:* **perror("Error executing command\n");**

The book shows functions named:  errExit or usageErr or fatal

They will not work for us unless we include the appropriate code from the text book.

# perror System Call

```
#include <stdio.h>

void perror (const char *msg);
```

More information in the text on pages 48-50

Example: **perror("Error changing directory\n");**

# Error Examples

With a system call:

```
if(chdir(dir) != 0)
    perror("Error changing directory \n");
```

Without a system call (in lab10):

```
if (out_redir != 0) {
    fprintf(stderr, "Cannot output to more than one file \n");
        _exit(EXIT_FAILURE);
}
```

# Lab 9 Slides

The End