

REMINDER: QUIZ 1 "C" IS AT CBTF (ENGR. GRANGER LIBRARY 57)  
THU-SUN. LIMITED SEATING - SCHEDULE YOUR 50 MIN SLOT

#1 Code review - how would you improve this code. Highlight every error you notice and then discuss the worst ones

```

01 // A program to run many commands in parallel
02 // Lines that start with an ! are executed
03 int main(int argc, char** argv) {
04     if(argc!=2) { printf("Usage: %s commandfile\n", argv); exit(1); }
05     size_t capacity = 200;
06     char* buffer = malloc(capacity);
07     ssize_t bytes;
08     FILE *file = fopen(argv[1], "r");
09     if(!file) { perror("Could not read file"); return 1;}
10     while( 1 ) {
11         bytes = getline(& buffer, & capacity , file );
12         buffer[bytes-1] = 0;
13         puts(buffer);
14         if( strcmp(buffer, "END") || bytes == -1) break;
15         if(*buffer == '!') {
16             if( ! fork() ) { execlp( "bash", buffer +1 , (char*) NULL); exit(1);}
17         }
18     }
19     return 0;
20 }
```

Line number : Comment or suggested fix

#2 What are POSIX signals?

#3 What are the two sources of signals?

#4 What are the most well known signals and what do they do?

SIGINT  
SIGSEGV  
SIGKILL

Demo.

First let's create an unsuspecting long running process ...

```

01 // dotwriter.c
02 int main() {
03     printf("My pid is %d\n", getpid() );
04     int i = 60;
05     while(--i) {
06         write(1, ".",1);
07         sleep(1);
08     }
09     write(1, "Done!",5);
10     return 0;
11 }
```

How can I send a signal from another program?

```

01 int main(int argc, char** argv) {
02     int signal = atoi(argv[1]);
03     pid_t pid = atoi( argv[2] );
04
05     if(signal && pid) _____
06     return 0;
07 }
```

How can I send a signal from the terminal?

#5 How would you modify the dotwriter program to send itself a SIGINT, after 5 dots?

## #6 Alarming signals

```
01 void main() {
02     char result[20];
03     puts("You have 4 seconds");
04     while(1) {
05         puts("Secret backdoor NSA Password?");
06         char* rc = fgets( result, sizeof(result) , stdin);
07         if(*result=='#') break;
08     }
09     puts("Congratulations. Connecting to NSA ...");
10     execlp("ssh", "ssh", "nsa-backdoor.net", (char*)NULL);
11     perror("Do you not have ssh installed?"); return 1;
12 }
```

## #7 Stopping and continuing programs

```
SIGSTOP
SIGCONT
```

## #8 Shell Job control, background processes and redirection (>) pipes (|)

```
&
ps
jobs
fg
bg
nohup dosomething.sh &
wc *.c > data.txt
```

```
01 #!/bin/bash
02 python analysis.py 1.dat &
03 python analysis.py 5.dat &
04 python analysis.py 8.dat &
05 wait
06 ...
```

## #9 Spot the errors part 1

```
01 // Spot the errors part 1
02 double *a = malloc( sizeof(double*) );
03 double *b = a;
04 free(b); b = 0;
05 *a = (double) 0xbaadf00d;

06 char* result;
07 strcpy(result, "CrashMaybe");

08 void* append(char** ptr, const char*mesg) {
09     if(!*ptr) ptr = malloc( strlen(mesg) );
10     strcat( *ptr, mesg);
11 }
```

## #10 Spot the errors part 2

```
01 //Spot the errors part 2
02 char* f() {
03     char result[16];
04     strcat( result, "Hi");
05     int *a;
06     if( &a != NULL) { printf("Yes %d\n",42); }

07     struct link* first= malloc(sizeof(struct link*));
08     free(first)
09     if(first->next) free(first->next);
10     return result;
11 }
```