

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

```
1 // A program to run many commands in parallel
2 // Lines that start with an ! are executed
3 int main(int argc, char** argv) {
4     if(argc!=2) { printf("Usage: %s commandfile\n", argv); exit(1); }
5     size_t capacity = 200;
6     char* buffer = malloc(capacity);
7     ssize_t bytes;
8     FILE *file = fopen(argv[1], "r");
9     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 ...

```
1 // dotwriter.c
2 int main() {
3     printf("My pid is %d\n", getpid() );
4     int i = 60;
5     while(--i) {
6         write(1, ".", 1);
7         sleep(1);
8     }
9     write(1, "Done!", 5);
10    return 0;
11 }
```

How can I send a signal from another program?

```
1 int main(int argc, char** argv) {
2     int signal = atoi(argv[1]);
3     pid_t pid = atoi( argv[2] );
4
5     if(signal && pid) _____
6     return 0;
7 }
```

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

```
1 void main() {
2     char result[20];
3     puts("You have 4 seconds");
4     while(1) {
5         puts("Secret backdoor NSA Password?");
6         char* rc = fgets( result, sizeof(result) , stdin);
7         if(*result='#') break;
8     }
9     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 demo Background processes and redirection (>) pipes (|)

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

#9 Spot the errors part 1

```
1 // Spot the errors part 1
2 double *a = malloc( sizeof(double*) );
3 double *b = a;
4 free(b); b = 0;
5 *a = (double) 0xbaadf00d;

6 char* result;
7 strcpy(result, "CrashMaybe");

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

#10 Spot the errors part 2

```
1 //Spot the errors part 2
2 char* f() {
3     char result[16];
4     strcat( result, "Hi");
5     int *a;
6     if( &a != NULL) { printf("Yes %d\n",42); }

7     struct link* first= malloc(sizeof(struct link*));
8     free(first)
9     if(first->next) free(first->next);
10    return result;
11 }
```