

# Advanced Programming

## COMS 3157

April 18, 2025

Patrick Shen

pts2125@columbia.edu

---

1. Q1? (2 marks)
  - (a) What is a signal?
  - (b) What is a signal handler?
2. Do the `signal()` and `sigaction()` methods pause the flow of code? (1 mark)
3. How would each of these signals be triggered? (4 marks)
  - (a) SIGFPE
  - (b) SIGINT
  - (c) SIGTSTP
  - (d) SIGCONT
4. Which two signals cannot be handled? (2 marks)
5. Briefly explain each argument in `sigaction(int signum, const struct sigaction *act, struct sigaction *oldact)` (3 marks)
  - (a) `int signum`
  - (b) `struct *act`
6. What do each field for in the `sigaction` struct? (4 marks)
  - (a) `void (*sa_handler)(int);`
  - (b) `void (*sa_sigaction)(int, siginfo_t *, void *);`
  - (c) `sigset_t sa_mask;`
  - (d) `int sa_flags;`

```
1  struct sigaction {
2      void      (*sa_handler)(int);
3      void      (*sa_sigaction)(int, siginfo_t *, void *);
4      sigset_t   sa_mask;
5      int       sa_flags;
6      void      (*sa_restorer)(void); // obsolete, ignore
7  };
```

## Listing 1: sigaction struct

7. Briefly explain what each function does for sa\_mask in the sigaction struct (3 marks)

- (a) int sigemptyset(sigset\_t \*set)
- (b) int sigaddset(sigset\_t \*set, int signum)
- (c) int sigfillset(sigset\_t \*set)

8. What does the function call memset() do here? (1 mark)

```

1  struct sigaction act;
2
3  memset (&act, '\0', sizeof(act));
4

```

## Listing 2: memset()

9. What does act = {0} do here? (1 mark)

```

1  struct sigaction act;
2
3  act = {0};
4

```

## Listing 3: act

10. Suppose a SIGTERM signal comes in. What is the output? (1 mark)

```

1  static void hd1 (int sig, siginfo_t *siginfo, void *context)
2  {
3      printf("SIGTERM receieved.");
4  }
5
6  ....
7
8  struct sigaction act;
9
10  memset (&act, '\0', sizeof(act));
11
12  act.sa_sigaction = &hd1;
13  act.sa_flags = SA_SIGINFO;
14
15  if (sigaction(SIGTERM, &act, NULL) < 0)
16  {
17      perror("sigaction");
18      return 1;
19  }
20

```

## Listing 4: simple example

11. What do the following keywords in C do? (2 marks)

- (a) volatile
- (b) sig\_atomic\_t

```
1 volatile sig_atomic_t signal_val = 0;
```

Listing 5: keywords

12. What does the raise(int iSig) function do? (1 mark)

13. What does the kill(pid\_t iPid, int iSig) function do? (1 mark)

14. What is the output for each of these commands? The code is stored in a executable named "sleep". (2 marks)

- (a) ./sleep 2 (Ctrl + C is not sent)
- (b) ./sleep 5 (Ctrl + C is sent 4 seconds in)

```
1 void catch_signal(int sig) {
2     got_signal = 1;
3 }
4
5 int main(int argc, char *argv[]) {
6     if (argc != 2) {
7         fprintf(stderr, "Usage: %s <seconds>\n", argv[0]);
8         return EXIT_FAILURE;
9     }
10
11     int max_snooze_secs = atoi(argv[1]);
12     if (max_snooze_secs <= 0) {
13         fprintf(stderr,
14             "Error: Invalid number of seconds '%s' for max snooze
time.\n",
15             argv[1]);
16         return EXIT_FAILURE;
17     }
18
19     struct sigaction action = {0};
20     action.sa_handler = catch_signal;
21     action.sa_flags = SA_RESTART;
22     if (sigaction(SIGINT, &action, NULL) == -1) {
23         perror("sigaction");
24         return EXIT_FAILURE;
25     }
26
27     int count = 0;
28     while (!got_signal && count < max_snooze_secs) {
29         sleep(1);
```

```

30         count++;
31     }
32     printf("Slept for %d of the %d seconds allowed.\n",
33         count, max_snooze_secs);
34
35     return EXIT_SUCCESS;
36 }
37
38

```

Listing 8: sleep() example

15. Answer the following questions about the alarm(int time) function. (2 marks)

- (a) What does the alarm(int time) function do?
- (b) What happens if the time argument is set to 0?

16. What is the output of this code? (1 mark)

```

1 static void myHandler(int iSig)
2 {
3     printf("In myHandler with argument %d\n", iSig);
4     alarm(2); /* Set another alarm */
5 }
6
7 int main(void)
8 {
9     signal(SIGALARM, myHandler);
10    alarm(2); /* Set an alarm */
11    printf("Entering an infinite loop\n");
12    for (;;)
13        ;
14    return 0;
15 }
16

```

Listing 9: alarm() example

17. What is the output for each of these scenarios? (1 mark)

- (a) you enter the number '4' after 4 seconds
- (b) you enter the number '7' after 7 seconds

```

1 static void myHandler(int iSig)
2 {
3     printf("\nSorry. You took too long.\n");
4     exit(EXIT_RETURN);
5 }
6
7 intn main(void)
8 {

```

```
9  int i;
10  signal(SIGALRM, myHandler);
11  printf("Enter a number: ");
12  alarm(5);
13  scanf("%d", &i);
14  alarm(0);
15  printf("You entered the number %d.\n", i);
16  return 0;
17 }
```

Listing 10: time bomb example