Sistemas Operativos

Formulario de auto-evaluación

Modulo 2. Sesión 5. Llamadas al sistema para gestión y control de señales.

Nombre y apellidos:

David Sánchez Jiménez

a) Cuestionario de actitud frente al trabajo.

El tiempo que he dedicado a la preparación de la sesión antes de asistir al laboratorio ha sido de …… minutos.

1. He resuelto todas las dudas que tenía antes de iniciar la sesión de prácticas: …… (si/no). En caso de haber contestado “no”, indica los motivos por los que no las has resuelto:

2. Tengo que trabajar algo más los conceptos sobre:

3. Comentarios y sugerencias:

b) Cuestionario de conocimientos adquiridos.

Mi solución al **ejercicio 2** ha sido:

```
#include <errno.h>
#include <signal.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>

int contador[35] = {0};

static void sig_USR_hdlr(int sigNum) {
  contador[sigNum] += 1;
  printf("\nLa senial %d se ha recibido %d veces\n", sigNum, contador[sigNum]);
}

int main(int argc, char const *argv[]) {
  struct sigaction sig_USR_nact;
  if (setvbuf(stdout, NULL, _IONBF, 0)) {
    perror("\nError en setvbuf");
  }

  sig_USR_nact.sa_handler = sig_USR_hdlr;
  sigemptyset(&sig_USR_nact.sa_mask);
  sig_USR_nact.sa_flags = 0;

  printf("\nNo puedo capturar la senial 9");
  printf("\nNo puedo capturar la senial 19");
  printf("\nEsperando el envio de señales...");

  // SIGHUP
  if (sigaction(SIGHUP, &sig_USR_nact, NULL) < 0) {
    perror("\nError al intentar establecer el manejador de señal para SIGHUP");
    exit(-1);
  }

  // SIGINT
  if (sigaction(SIGINT, &sig_USR_nact, NULL) < 0) {
    perror("\nError al intentar establecer el manejador de señal para SIGINT");
    exit(-1);
  }

  // SIGQUIT
  if (sigaction(SIGQUIT, &sig_USR_nact, NULL) < 0) {
    perror("\nError al intentar establecer el manejador de señal para SIGQUIT");
    exit(-1);
  }

  // SIGILL
  if (sigaction(SIGILL, &sig_USR_nact, NULL) < 0) {
    perror("\nError al intentar establecer el manejador de señal para SIGILL");
    exit(-1);
  }
```

```
// SIGTRAP
if (sigaction(SIGTRAP, &sig_USR_nact, NULL) < 0) {
  perror("\nError al intentar establecer el manejador de señal para SIGTRAP");
  exit(-1);
}

// SIGABRT
if (sigaction(SIGABRT, &sig_USR_nact, NULL) < 0) {
  perror("\nError al intentar establecer el manejador de señal para SIGABRT");
  exit(-1);
}

// SIGBUS
if (sigaction(SIGBUS, &sig_USR_nact, NULL) < 0) {
  perror("\nError al intentar establecer el manejador de señal para SIGBUS");
  exit(-1);
}

// SIGFPE
if (sigaction(SIGFPE, &sig_USR_nact, NULL) < 0) {
  perror("\nError al intentar establecer el manejador de señal para SIGFPE");
  exit(-1);
}

// SIGUSR1
if (sigaction(SIGUSR1, &sig_USR_nact, NULL) < 0) {
  perror("\nError al intentar establecer el manejador de señal para SIGUSR1");
  exit(-1);
}

// SIGUSR2
if (sigaction(SIGUSR2, &sig_USR_nact, NULL) < 0) {
  perror("\nError al intentar establecer el manejador de señal para SIGUSR2");
  exit(-1);
}

// SIGSEGV
if (sigaction(SIGSEGV, &sig_USR_nact, NULL) < 0) {
  perror("\nError al intentar establecer el manejador de señal para SIGSEGV");
  exit(-1);
}

// SIGPIPE
if (sigaction(SIGPIPE, &sig_USR_nact, NULL) < 0) {
  perror("\nError al intentar establecer el manejador de señal para SIGPIPE");
  exit(-1);
}

// SIGALRM
if (sigaction(SIGALRM, &sig_USR_nact, NULL) < 0) {
  perror("\nError al intentar establecer el manejador de señal para SIGALRM");
  exit(-1);
}
```

```
// SIGTERM
if (sigaction(SIGTERM, &sig_USR_nact, NULL) < 0) {
  perror("\nError al intentar establecer el manejador de señal para SIGTERM");
  exit(-1);
}

// SIGSTKFLT
if (sigaction(SIGSTKFLT, &sig_USR_nact, NULL) < 0) {
  perror(
      "\nError al intentar establecer el manejador de señal para SIGSTKFLT");
  exit(-1);
}

// SIGCHLD
if (sigaction(SIGCHLD, &sig_USR_nact, NULL) < 0) {
  perror("\nError al intentar establecer el manejador de señal para SIGCHLD");
  exit(-1);
}

// SIGCONT
if (sigaction(SIGCONT, &sig_USR_nact, NULL) < 0) {
  perror("\nError al intentar establecer el manejador de señal para SIGCONT");
  exit(-1);
}

// SIGTSTP
if (sigaction(SIGTSTP, &sig_USR_nact, NULL) < 0) {
  perror("\nError al intentar establecer el manejador de señal para SIGTSTP");
  exit(-1);
}

// SIGTTIN
if (sigaction(SIGTTIN, &sig_USR_nact, NULL) < 0) {
  perror(
      "\nError al intentar establecer el manejador de señal para   SIGTTIN");
  exit(-1);
}

// SIGTTOU
if (sigaction(SIGTTOU, &sig_USR_nact, NULL) < 0) {
  perror("\nError al intentar establecer el manejador de señal para SIGTTOU");
  exit(-1);
}

// SIGURG
if (sigaction(SIGURG, &sig_USR_nact, NULL) < 0) {
  perror("\nError al intentar establecer el manejador de señal para SIGURG");
  exit(-1);
}

// SIGXCPU
if (sigaction(SIGXCPU, &sig_USR_nact, NULL) < 0) {
  perror("\nError al intentar establecer el manejador de señal para SIGXCPU");
```

```
    exit(-1);
  }

  // SIGXFSZ
  if (sigaction(SIGXFSZ, &sig_USR_nact, NULL) < 0) {
    perror("\nError al intentar establecer el manejador de señal para SIGXFSZ");
    exit(-1);
  }

  // SIGVTALRM
  if (sigaction(SIGVTALRM, &sig_USR_nact, NULL) < 0) {
    perror(
        "\nError al intentar establecer el manejador de señal para SIGVTALRM");
    exit(-1);
  }

  // SIGPROF
  if (sigaction(SIGPROF, &sig_USR_nact, NULL) < 0) {
    perror("\nError al intentar establecer el manejador de señal para SIGPROF");
    exit(-1);
  }

  // SIGWINCH
  if (sigaction(SIGWINCH, &sig_USR_nact, NULL) < 0) {
    perror(
        "\nError al intentar establecer el manejador de señal para SIGWINCH");
    exit(-1);
  }

  // SIGPOLL
  if (sigaction(SIGPOLL, &sig_USR_nact, NULL) < 0) {
    perror("\nError al intentar establecer el manejador de señal para SIGPOLL");
    exit(-1);
  }

  // SIGPWR
  if (sigaction(SIGPWR, &sig_USR_nact, NULL) < 0) {
    perror("\nError al intentar establecer el manejador de señal para SIGPWR");
    exit(-1);
  }

  // SIGSYS
  if (sigaction(SIGSYS, &sig_USR_nact, NULL) < 0) {
    perror("\nError al intentar establecer el manejador de señal para SIGSYS");
    exit(-1);
  }

  for (;;) {
  }
}
```

Mi solución a la **ejercicio 3** ha sido:

```
#include <signal.h>
#include <stdio.h>

int main(int argc, char const *argv[]) {
  sigset_t new_mask;
  sigemptyset(&new_mask); // Inicializamos la nueva mascara de señales
  sigfillset(&new_mask);  // Inicializamos la mascara con todas las señales
  sigdelset(&new_mask, SIGUSR1); // Borramos la señal SIGUSR1 del conjunto
  sigsuspend(&new_mask);         // Esperamos a la señal SIGUSR1
}
```

Mi solución a la **ejercicio 4** ha sido:

Se crea una máscara en la que se añade la señal SIGTERM y se aplica para bloquear dicha señal.
Se envía la orden sleep con una duración de 10 segundos en el cual si enviamos dicha señal este no reacciona debido a que esta está bloqueada.
Al despertar aplicando la mascara antigua por lo que se desbloquea la señal SIGTERM. Comprueba si hemos introducido dicha señal mirando si variable signal_recibida se encuentra activa y avisa por pantalla al usuario.