

Examen-Septiembre-2018-resuelto.pdf



Yassine_bhk



Fundamentos de Software de Comunicaciones



2º Grado en Ingeniería de las Tecnologías de Telecomunicación



Escuela Técnica Superior de Ingeniería de Telecomunicación
Universidad de Málaga



**TU ORDENADOR VIEJO
SI QUE NECESITA
DEJAR LA CARRERA
ESTE LA TERMINA**



MSI CREATOR M16

“No necesitas besos, si no volcanes de fuego en tu boca”.
¿De quién es la frase?

- a) William Shakespeare
- b) Federico García Lorca
- c) Nicholas Leister.

Culpa Mía

prime

VER AHORA

```
/* CLIENTE */
```

```
#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <stdio.h>
#include <time.h>
#include <stdlib.h>
#include <string.h>
#include <errno.h>
#include <sys/wait.h>
#include <sys/stat.h>
#include <font1.h>
#include <string.h>
#include <sys/select.h>
#include <sys/time.h>
#include <sys/sysinfo.h>
#include <signal.h>
```

```
ssize_t readn (int fd, void * datos, size_t n) {

    char * p = (char *) datos;
    int leidos;
    int intentando_leer = n;
    int total_leidos = 0;

    do {

        errno = 0;
        leidos = read (fd, p + total_leidos, intentando_leer);

        if (leidos > 0) {

            total_leidos += leidos;
            intentando_leer -= leidos;
        }
    } while (leidos > 0 && total_leidos < n || errno == EINTR);

    if (leidos < 0) {
```

WUOLAH

```

        return leidos;
    }

    else {

        return total_leidos;
    }

}

ssize_t writen (int fd, void * datos, size_t n) {

    char * p = (char *) datos;
    int leidos;
    int intentando_leer = n;
    int total_leidos = 0;

    do {

        errno = 0;
        leidos = write (fd, p + total_leidos, intentando_leer);

        if (leidos > 0) {

            total_leidos += leidos;
            intentando_leer -= leidos;
        }
    } while (leidos > 0 && total_leidos < n || errno == EINTR);

    if (leidos < 0) {

        return leidos;
    }

    else {

        return total_leidos;
    }

}

#define PUERTO 4950
#define TAM_MAX 512

```


Lenovo

TARDARÍAS MENOS EN COMPRAR
ESTE PORTÁTIL QUE TU ORDENADOR
EN REINICIARSE. AHÍ LO DEJO.

RTX4000

CUPÓN WUOLAH
-25% DTO.

Válido hasta el 30/06



CONFIGURA EL EQUIPO A MEDIDA EN [LENOVO.COM](https://lenovo.com)

legion pro 5i

Rendimiento de juego de última generación con Lenovo AI Engine+. Procesadores Intel® Core™ de 13.ª generación y la tarjeta gráfica NVIDIA® GeForce RTX™ 4000 para vivir la experiencia de juego definitiva. Almacenamiento SSD y almacenamiento DDR5 para disfrutar de tiempos de carga y almacenamiento más rápidos. Tecnología de refrigeración Legion ColdFront 5.0 para tener un control óptimo de la temperatura durante el juego. Teclado Legion TrueStrike y pantalla WQXGA de 40,64 cm (16") con frecuencia alta de actualización para sumergirte de lleno en el juego.



```

int main (int argc, char * argv []) {

    if (argc < 3) {

        printf ("Escribe el argumento porfavor");
        exit (1);
    }

    int fd_lectura = open (argv [2], O_RDONLY);
    printf ("nombre de fichero: %s\n", argv [2]);

    if (fd_lectura < 0) {

        perror ("Abriendo fichero");
        exit (1);
    }

    int sd = socket (AF_INET, SOCK_STREAM, 0);

    if (sd < 0) {

        perror ("Socket");
        exit (1);
    }

    uint32_t ip = inet_addr (argv [1]);

    struct sockaddr_in vinculo;
    memset (&vinculo, 0, sizeof (vinculo));
    memcpy (&vinculo.sin_addr, &ip, 4);
    vinculo.sin_family = AF_INET;
    vinculo.sin_port = htons (PUERTO);

    int resultado = connect (sd, (struct sockaddr *) &vinculo, sizeof (vinculo));

    if (resultado < 0) {

        perror ("Connect");
        close (sd);
        exit (1);
    }
}

```

```

printf ("Exito al conectar...\n");

uint8_t tam_fichero = strlen (argv[2]);
int escritos = writen (sd, &tam_fichero, 1);
printf ("Escribiendo tamaño...\n");

    if (escritos < 0) {

        perror ("Al escribir el tamaño del fichero");
        exit (1);
        close (sd);
    }

escritos = writen (sd, argv [2], tam_fichero);
printf ("Escribiendo nombre...\n");

    if (escritos < 0) {

        perror ("Al escribir el tamaño del fichero");
        exit (1);
        close (sd);
    }


int leidos;
char datos [32];
uint16_t num_letras_n;
uint16_t num_letras;
uint16_t num_numeros_n;
uint16_t num_numeros;
uint16_t letras [512];
uint16_t numeros [512];
int ejecucion = 0;

do {

    leidos = readn (fd_lectura, datos, 32);
    //printf ("\n\n datos: %s \n\n", datos);

    if (leidos < 0) {

        perror ("Al leer del fichero");
    }
}

```

“No necesitas besos, si no volcanes de fuego en tu boca”.
¿De quién es la frase?

- a) William Shakespeare
- b) Federico García Lorca
- c) Nicholas Leister.

Culpa Mía

prime

VER AHORA

```
        exit (1);
        close (sd);
    }

    escritos = writen (sd, datos, 32);

    if (escritos < 0) {

        perror ("Al leer del fichero");
        exit (1);
        close (sd);
    }

    int l = readn (sd, &num_letras_n, 2);

    l = readn (sd, &num_numeros_n, 2);
    printf ("numeros: %d", num_numeros);

    num_letras = ntohs (num_letras_n);
    printf ("letras: %d ", num_letras);
    num_numeros = ntohs (num_numeros_n);

    numeros [ejecucion] = num_numeros;
    printf (" array: %d ", numeros [ejecucion]);

    letras [ejecucion] = num_letras;
    printf (" %d \n", letras [ejecucion]);

    ejecucion++;

} while (leidos > 0);

printf ("No quedan más ficheros\n");

for (int i = 0; i < ejecucion; i++) {

    printf ("\nBloque "); printf ("%d", i); printf (":");
    printf (" %d ", numeros [i]);
    printf (" %d\n", letras [i]);

}
```

WUOLAH

```
close (sd);  
  
return 0;  
}
```



```

/* SERVIDOR */

#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <stdio.h>
#include <time.h>
#include <stdlib.h>
#include <string.h>
#include <errno.h>
#include <sys/wait.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <string.h>
#include <sys/select.h>
#include <sys/time.h>
#include <sys/sysinfo.h>
#include <signal.h>

ssize_t readn (int fd, void * buffer, size_t n) {

    char * p = (char *) buffer;
    int leidos;
    int intentando_leer = n;
    int total_leidos = 0;

    do {

        errno = 0;
        leidos = read (fd, p + total_leidos, intentando_leer);

        if (leidos > 0) {

            total_leidos += leidos;
            intentando_leer -= leidos;
        }
    } while (leidos > 0 && total_leidos < n || errno == EINTR);

    if (leidos < 0) {

```

```

        return leidos;
    }

    else {

        return total_leidos;
    }

}

ssize_t writen (int fd, void * buffer, size_t n) {

    char * p = (char *) buffer;
    int leidos;
    int intentando_leer = n;
    int total_leidos = 0;

    do {

        errno = 0;
        leidos = write (fd, p + total_leidos, intentando_leer);

        if (leidos > 0) {

            total_leidos += leidos;
            intentando_leer -= leidos;
        }
    } while (leidos > 0 && total_leidos < n || errno == EINTR);

    if (leidos < 0) {

        return leidos;
    }

    else {

        return total_leidos;
    }

}

#define PUERTO 4950
#define TAM_MAX 512

```

“No necesitas besos, si no volcanes de fuego en tu boca”.
¿De quién es la frase?

- a) William Shakespeare
- b) Federico García Lorca
- c) Nicholas Leister.

Culpa Mía

prime

VER AHORA

```
int fin = 0;

void manejadora (int s) {

    fin = 1;
}

int main () {

    int fd_escritura = open ("log.txt", O_WRONLY | O_APPEND | O_CREAT, 0644);

    signal (SIGINT, manejadora);

    if (fd_escritura < 0) {

        perror ("Error al acceder al fichero de escritura\n");
        exit (1);
    }

    int sd = socket (PF_INET, SOCK_STREAM, 0);

    if (sd < 0) {

        perror ("Socket");
        exit (1);
    }

    struct sockaddr_in vinculo;
    memset (&vinculo, 0, sizeof (vinculo));
    vinculo.sin_addr.s_addr = INADDR_ANY;
    vinculo.sin_family = AF_INET;
    vinculo.sin_port = htons (PUERTO);

    int resultado = bind (sd, (struct sockaddr *) &vinculo, sizeof (vinculo));

    if (resultado < 0) {

        perror ("Bind");
        close (sd);
        exit (1);
    }
}
```

WUOLAH

```

    }

    resultado = listen (sd, 10);

    if (resultado < 0) {

        perror ("Listen");
        close (sd);
        exit (1);
    }

    struct sockaddr_in dat_cliente;

    while (!fin) {

        printf ("\n\n.....Esperando cliente.....\n\n");

        socklen_t longitud = sizeof (dat_cliente);
        int csd = accept (sd, ( struct sockaddr *) &dat_cliente, &longitud);
        printf ("Exito al aceptar...\n");

        if (csd < 0) {

            perror ("Accept");
            close (sd);
            exit (1);
        }

        uint8_t tam_nom;
        char nombre [TAM_MAX];
        char datos [32];
        uint16_t num_letras = 0;
        uint16_t num_letras_n;
        uint16_t num_numero_n;
        uint16_t num_numeros = 0;
        uint16_t letras [512];
        uint16_t numeros [512];
        int ejecucion = 0;

        int leidos = readn (csd, &tam_nom, 1);
        //printf ("Leyendo tamaño..\n");

```

```

    if (leidos < 0) {

        perror ("Error al leer la longitud del fichero");
        close (csd);
        close (sd);
        exit (1);
    }

leidos = readn (csd, nombre, tam_nom);
//printf ("Leyendo nombre...\n");

    if (leidos < 0) {

        perror ("Error al leer el tamaño del fichero");
        close (csd);
        close (sd);
        exit (1);
    }
nombre [tam_nom] = '\0';
printf ("Datos provenientes del fichero: %s\n", nombre);

do {

    num_numeros = 0; num_letras = 0;
    leidos = readn (csd, datos, 32);
    datos [leidos] = '\0';
    printf ("\n\n...leyendo...\n\n");
    writen (1, datos, 32);

        if (leidos < 0) {

            perror ("Error al leer datos");
            close (csd);
            close (sd);
            exit (1);
        }

    for (int i = 0; i < leidos; i++) {

        if (datos [i] <= '9' && datos [i] >= '0') {

```



```

        num_numeros++;
    }

    if ((datos [i] <= 'z' && datos [i] >= 'a') || (datos [i] <= 'Z'
&& datos [i] >= 'A')) {

        num_letras++;

    }
}

//printf ("\nNum_numeros %d\n", num_numeros);
numeros [ejecucion] = num_numeros;
letras [ejecucion] = num_letras;

num_letras_n = htons (num_letras);
num_numero_n = htons (num_numeros);

int escritos = write (csd, &num_letras_n, sizeof (uint16_t));

if (escritos < 0) {

    perror ("Escribiendo");
    close (csd);
    close (sd);
    exit (1);
}

escritos = write (csd, &num_numero_n, sizeof (uint16_t));

if (escritos < 0) {

    perror ("Escribiendo");
    close (csd);
    close (sd);
    exit (1);
}

ejecucion++;

} while (leidos > 0);

close (csd);

```

“No necesitas besos, si no volcanes de fuego en tu boca”.
¿De quién es la frase?

- a) William Shakespeare
- b) Federico García Lorca
- c) Nicholas Leister.

Culpa Mía

prime

VER AHORA

```
printf ("He llegado a aqui\n");

int escritos = writen (fd_escritura, nombre, tam_nom);

if (escritos < 0) {

    perror ("Error al leer el tamaño del fichero");
    close (csd);
    exit (1);
}

for (int i = 0; i < ejecucion; i++) {

    writen (fd_escritura, " que pasa", 10);
    writen (fd_escritura, &numeros [i], sizeof (uint16_t));
    writen (fd_escritura, " ", 1);
    writen (fd_escritura, &letras [i], sizeof (uint16_t));
}

} // bucle while

printf ("\n FIN");
close (sd);
close (fd_escritura);

return 0;
}
```

WUOLAH

```

/* EJERCICIO 1*/

#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <stdio.h>
#include <time.h>
#include <stdlib.h>
#include <string.h>
#include <errno.h>
#include <sys/wait.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <string.h>
#include <sys/select.h>
#include <sys/time.h>
#include <sys/sysinfo.h>
#include <signal.h>

#define EST_ESPERA 0
#define EST_MOSTRAR_PROCESO 1
#define EST_MOSTRAR_MEMORIA 2

#define EV_SIGINT 0
#define EV_SIGUSR1 1
#define EV_SIGUSR2 2
#define EV_TIMEOUT 3

int pipe_fd [2];

ssize_t writen (int fd, void * buffer, size_t n) {

    char * p = (char *) buffer;
    int escritos;
    int intentado_escribir = n;
    int total_escritos = 0;

    do {

        errno = 0;

```

```

    escritos = write (fd, p + total_escritos, intentado_escribir);

    if (escritos > 0) {

        total_escritos += escritos;
        intentado_escribir -= escritos;
    }

} while (escritos > 0 && total_escritos < n || errno == EINTR);

if (escritos < 0) {

    return escritos;
}

else {

    return total_escritos;
}
}

ssize_t readn (int fd, void * buffer, size_t n) {

    char * p = (char *) buffer;
    int leidos;
    int intentado_leer = n;
    int total_leidos = 0;

    do {

        errno = 0;
        leidos = read (fd, p + total_leidos, intentado_leer);

        if (leidos > 0) {

            total_leidos += leidos;
            intentado_leer -= leidos;
        }

    } while (leidos > 0 && total_leidos < n || errno == EINTR);

    if (leidos < 0) {

```

```

        return leidos;
    }

    else {

        return total_leidos;
    }
}

void manejadora1 (int s) {

    signal (s, manejadora1);
    printf ("Manejadora sigusr1\n");
    uint8_t evento = EV_SIGUSR1;
    int escritos = writen (pipe_fd [1], &evento, 1);

}

void manejadora2 (int s) {

    signal (s, manejadora2);
    printf ("Manejadora sigusr2\n");
    uint8_t evento = EV_SIGUSR2;
    int escritos = writen (pipe_fd [1], &evento, 1);

}

void manejadora3 (int s) {

    signal (s, manejadora3);
    printf ("Manejadora ctrl\n");
    uint8_t evento = EV_SIGINT;
    int escritos = writen (pipe_fd [1], &evento, 1);

}

void manejadora4 (int s) {

    signal (s, manejadora4);
    printf ("Manejadora timeout\n");
    uint8_t evento = EV_TIMEOUT;
    int escritos = writen (pipe_fd [1], &evento, 1);

```


“No necesitas besos, si no volcanes de fuego en tu boca”.
¿De quién es la frase?

- a) William Shakespeare
- b) Federico García Lorca
- c) Nicholas Leister.

Culpa Mia

prime

VER AHORA

```
}

int espera_evento () {

    uint8_t evento;

    int leidos = readn (pipe_fd [0], &evento, 1);

    if (leidos > 0) {

        return evento;
    }

    else return -1;
}

int main () {

    pipe (pipe_fd);

    signal (SIGUSR1, manejadora1);
    signal (SIGUSR2, manejadora2);
    signal (SIGINT, manejadora3);
    signal (SIGALRM, manejadora4);

    int estado = EST_ESPERA;

    struct sysinfo info;
    sysinfo(&info);
    unsigned short dato;
    unsigned long dato2;

    struct itimerval timer;
    memset (&timer, 0, sizeof (timer));
    struct timeval tiempo = {5, 500000};

    while (1) {

        int evento = espera_evento ();
```

WUOLAH

```

if (evento < 0) {

    printf ("Evento no reconocido\n");
    break;
}

switch (estado) {

    case EST_ESPERA :

        switch (evento) {

            case EV_SIGINT :

                break;

            case EV_SIGUSR1 :

                printf ("Pasando al estado mostrar proceso...\n");
                estado = EST_MOSTRAR_PROCESO;
                timer.it_interval = tiempo;
                timer.it_value = tiempo;
                setitimer (ITIMER_REAL, &timer, 0);

                break;

            case EV_SIGUSR2 :

                printf ("Pasando al estado mostrar memoria...\n");
                estado = EST_MOSTRAR_MEMORIA;
                timer.it_interval = tiempo;
                timer.it_value = tiempo;
                setitimer (ITIMER_REAL, &timer, 0);

                break;

            default:

                printf ("Evento no esperado en estado espera\n");
                exit (1);
        }
    }
}

```

```

        break;
    }

break;

case EST_MOSTRAR_PROCESO :

    switch (evento) {

        case EV_SIGINT :

            printf ("Pasando al estado espera...\n");
            estado = EST_ESPERA;
            memset (&timer, 0, sizeof (timer));
            setitimer (ITIMER_REAL, &timer, 0);

            break;

        case EV_TIMEOUT :

            dato = info.procs;
            printf ("Número de procesos activos: %d\n", dato);

            break;

        case EV_SIGUSR2 :

            printf ("Pasando al estado mostrar memoria...\n");
            estado = EST_MOSTRAR_MEMORIA;
            timer.it_interval = tiempo;
            timer.it_value = tiempo;
            setitimer (ITIMER_REAL, &timer, 0);

            break;

        default:

            printf ("Evento no esperado en estado proceso\n");
            exit (1);

            break;
    }
}

```

```

break;

case EST_MOSTRAR_MEMORIA :

    switch (evento) {

        case EV_SIGINT :

            printf ("Pasando al estado espera..\n");
            estado = EST_ESPERA;
            memset (&timer, 0, sizeof (timer));
            setitimer (ITIMER_REAL, &timer, 0);

            break;

        case EV_SIGUSR1 :

            printf ("Pasando al estado mostrar proceso...\n");
            estado = EST_MOSTRAR_PROCESO;
            timer.it_interval = tiempo;
            timer.it_value = tiempo;
            setitimer (ITIMER_REAL, &timer, 0);

            break;

        case EV_TIMEOUT :

            dato2 = info.freeram;
            printf ("Memoria hábil: %d\n", dato);

            break;

        default:

            printf ("Evento no esperado en memoria\n");
            exit (1);

            break;

    }

break;

```

“No necesitas besos, si no volcanes de fuego en tu boca”.
¿De quién es la frase?

- a) William Shakespeare
- b) Federico García Lorca
- c) Nicholas Leister.

Culpa Mía

prime

VER AHORA

```
        default :  
  
            printf ("Estado no reconocido\n");  
            exit (1);  
  
        break;  
    }  
}  
  
return 0;  
}
```

WUOLAH

