SOPER P2 - P10: Blanca Martín y Fernando Villar

Generado por Doxygen 1.8.8

Viernes, 17 de Marzo de 2017 11:21:00

Índice general

Capítulo 1

Índice de clases

1.1. Lista de clases

Lista de las clases, estructuras, uniones e interfaces con una breve descripción:

Capítulo 2

Indice de archivos

2.1. Lista de archivos

Lista de todos los archivos con descripciones breves:

Capítulo 3

Documentación de las clases

3.1. Referencia de la Estructura param

Parametros de entrada para la funcion mult_matrices().

Diagrama de colaboración para param:

Atributos públicos

- int id
- int mult
- int dim
- int * matrix
- int fila
- struct param * h

3.1.1. Descripción detallada

Parametros de entrada para la funcion mult_matrices().

Estructura que contendra todos los datos necesarios que deberan ser pasados a la funcion mult_matrices().

3.1.2. Documentación de los datos miembro

3.1.2.1. int param::dim

Tamaño de una fila/columna de la matriz

3.1.2.2. int param::fila

Fila actual

3.1.2.3. struct param* param::h

Estructura del otro hilo

3.1.2.4. int param::id

Identificador del hilo en el que se esta operando

3.1.2.5. int* param::matrix

Matriz a multiplicar

3.1.2.6. int param::mult

Entero por el que se multiplica la matriz

La documentación para esta estructura fue generada a partir del siguiente fichero:

- ejercicio4.c

Capítulo 4

Documentación de archivos

4.1. Referencia del Archivo ejercicio10.c

Sistemas Operativos: Practica 2, ejercicio 10.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <signal.h>
#include <unistd.h>
#include <time.h>
```

Dependencia gráfica adjunta para ejercicio10.c:

'defines'

- #define NUM_WRD 13
- #define MAX_CHAR 10
- #define NUM_STR 50
- #define SENTENCE {"EL", "PROCESO", "A", "ESCRIBE", "EN", "UN", "FICHERO", "HASTA", "QUE", "LEE", "LA", "CADENA", "FIN"}

Funciones

- int randInt (int range)

  Funcion que genera un numero aleatorio.
- void despertar (int sig)

  Manejador 1: despertar.
- void alarma (int sig)

  Manejador 1: alarma.
- void morir (int sig)

  Manejador 1: morir.
- int main ()

  Funcion main del ejercicio10.

4.1.1. Descripción detallada

Sistemas Operativos: Practica 2, ejercicio 10.

Grupo 2201, Pareja 10. En este modulo se ha implementado el codigo del decimo ejercicio de la segunda practica, que consiste en el manejo de comunicacion entre dos procesos a base de señales y utilizando mascaras y manejadores.

Autor

Blanca Martín (blanca.martins@estudiante.uam.es)
Fernando Villar (fernando.villarg@estudiante.uam.es)

Fecha

17-03-2017

4.1.2. Documentación de los 'defines'

4.1.2.1. #define MAX_CHAR 10

Maximo de caracteres para palabras de la frase

4.1.2.2. #define NUM_STR 50

Numero de palabras que lee el proceso B

4.1.2.3. #define NUM_WRD 13

Numero de palabras de la frase

4.1.2.4. #define SENTENCE {"EL", "PROCESO", "A", "ESCRIBE", "EN", "UN", "FICHERO", "HASTA", "QUE", "LEE", "LA", "CADENA", "FIN"}

Frase

4.1.3. Documentación de las funciones

4.1.3.1. void alarma (int *sig*)

Manejador 1: alarma.

Manejador que sirve para temporizar las lecturas del proceso B a traves de alarm() y SIGALRM.

Parámetros

sig,numero	de la señal

4.1.3.2. void despertar (int *sig*)

Manejador 1: despertar.

Manejador que despierta al proceso A despues de que haya leido FIN a traves de la señal SIGUSR1

Parámetros

sig,numero	de la señal

4.1.3.3. int main ( )

Funcion main del ejercicio10.

El programa consiste en la creacion de un proceso padre y uno hijo. El hijo escribe en un fichero palabras de SENTENCE hasta que escribe FIN, momento en el que se bloquea, con una mascara que le hara desbloquearse solo si llegan las señales SIGUSR1, SIGALRM o SIGTERM. Mientras tanto, B lee cada 5 segundos una palabra del fichero, y cuando lee FIN desbloquea el proceso A para volver a repetir la operacion hasta que B haya leido NUM_STR palabras, momento en el que B manda una señal para que muera A y luego B muere.

Devuelve

EXIT_SUCCESS si se han realizado correctamente todas las tareas, EXIT_FAILURE si se ha producido algun error al crear los manejadores, al crear la mascara, al hacer el fork() o al abrir/cerrar el fichero.

4.1.3.4. void morir (int *sig*)

Manejador 1: morir.

Manejador que mata al proceso A cuando el proceso B ha leido NUM_STR palabras.

Parámetros

sig,numero	de la señal

4.1.3.5. int randInt (int *range*)

Funcion que genera un numero aleatorio.

randInt genera un numero entero aleatorio entre 0 y range-1.

Parámetros

range,maximo	para el numero aleatorio

Devuelve

int, el numero aleatorio generado

4.2. Referencia del Archivo ejercicio3a.c

Sistemas Operativos: Practica 2, ejercicio 3a.

```
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <sys/time.h>
#include <unistd.h>
```

Dependencia gráfica adjunta para ejercicio3a.c:

'defines'

- #define MAX_PROC 100

Funciones

- int is_prime (int n)

  Comprueba si un numero es primo.
- int main (int argc, char ∗argv[])

  Funcion main del ejercicio3a.

4.2.1. Descripción detallada

Sistemas Operativos: Practica 2, ejercicio 3a.

Grupo 2201, Pareja 10. En este modulo se ha implementado el codigo del primer apartado del tercer ejercicio de la segunda practica, que consiste en la comparacion de tiempos entre hilos y procesos.

Autor

Blanca Martín (blanca.martins@estudiante.uam.es)
Fernando Villar (fernando.villarg@estudiante.uam.es)

Fecha

17-03-2017

4.2.2. Documentación de los 'defines'

4.2.2.1. #define MAX_PROC 100

Maximo de proccesos a crear

4.2.3. Documentación de las funciones

4.2.3.1. int is_prime (int *n*)

Comprueba si un numero es primo.

isPrime() recibe un entero, comprueba si este es primo y devuelve el resultado.

Parámetros

n,entero	que se quiere comprobar.

Devuelve

0 si el numero no es primo, 1 si el numero es primo.

4.2.3.2. int main (int *argc,* char ∗ *argv[ ]*)

Funcion main del ejercicio3a.

El programa consiste en la creacion de 100 procesos hijos en serie, cada uno de los cuales calculara N numeros primos. El proceso padre esperara a cada uno de sus hijos, y una vez hayan finalizado todos los hijos se imprimira por pantalla el tiempo total empleado en todos los calculos.

Parámetros

argc,numero	de argumentos
argv,array	de strings con los argumentos (numero de primos a calcular).

Devuelve

EXIT_SUCCESS si se han realizado correctamente todas las tareas, EXIT_FAILURE si se ha producido algun error al introducir los argumentos de entrada o al ejecutar la funcion fork().

4.3. Referencia del Archivo ejercicio3b.c

Sistemas Operativos: Practica 2, ejercicio 3b.

```
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <sys/time.h>
#include <unistd.h>
#include <pthread.h>
```

Dependencia gráfica adjunta para ejercicio3b.c:

'defines'

- #define MAX_HILOS 100

Funciones

- int is_prime (int n)

  Comprueba si un numero es primo.
- void * primos (void *n)

  Calcula los primeros n primos.
- int main (int argc, char *argv[])

  Funcion main del ejercicio3b.

4.3.1. Descripción detallada

Sistemas Operativos: Practica 2, ejercicio 3b.

Grupo 2201, Pareja 10. En este modulo se ha implementado el codigo del segundo apartado del tercer ejercicio de la segunda practica, que consiste en la comparacion de tiempos entre hilos y procesos.

Autor

Blanca Martín (blanca.martins@estudiante.uam.es)
Fernando Villar (fernando.villarg@estudiante.uam.es)

Fecha

17-03-2017

4.3.2. Documentación de los 'defines'

4.3.2.1. #define MAX_HILOS 100

Maximo de hilos a crear

4.3.3. Documentación de las funciones

4.3.3.1. int is_prime (int *n*)

Comprueba si un numero es primo.

isPrime() recibe un entero, comprueba si este es primo y devuelve el resultado.

Parámetros

n,entero	que se quiere comprobar.

Devuelve

0 si el numero no es primo, 1 si el numero es primo.

4.3.3.2. int main (int *argc,* char * *argv[]*)

Funcion main del ejercicio3b.

El programa consiste en la creacion de 100 hilos, cada uno de los cuales calculara N numeros primos. Una vez hayan finalizado todos los hilos se imprimira por pantalla el tiempo total empleado en todos los calculos.

Parámetros

argc,numero	de argumentos
argv,array	de strings con los argumentos (numero de primos a calcular).

Devuelve

EXIT_SUCCESS si se han realizado correctamente todas las tareas, EXIT_FAILURE si se ha producido algun error al introducir los argumentos de entrada o al ejecutar la funcion pthread_create().

4.3.3.3. void * primos (void * *n*)

Calcula los primeros n primos.

primos() recibe un entero que indica el numero de primos a generar y calcula los mismos.

Parámetros

n,numero	de primos a generar.

Devuelve

puntero a void del numero de primos a generar.

4.4. Referencia del Archivo ejercicio4.c

Sistemas Operativos: Practica 2, ejercicio 4.

```
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <sys/time.h>
#include <unistd.h>
#include <pthread.h>
```

Dependencia gráfica adjunta para ejercicio4.c:

Clases

- struct param

  Parametros de entrada para la funcion mult_matrices().

Funciones

- void * mult_matrices (void *parametros)

  Multiplica una matriz por un entero y muestra el resultado.
- int main ()

  Funcion main del ejercicio4.

4.4.1. Descripción detallada

Sistemas Operativos: Practica 2, ejercicio 4.

Grupo 2201, Pareja 10. En este modulo se ha implementado el codigo del cuarto ejercicio de la segunda practica, en el que se emplean hilos para la multiplicacion de dos matrices por un entero.

Autor

Blanca Martín (blanca.martins@estudiante.uam.es)
Fernando Villar (fernando.villarg@estudiante.uam.es)

Fecha

17-03-2017

4.4.2. Documentación de las funciones

4.4.2.1. int main ( )

Funcion main del ejercicio4.

El programa consiste en la creacion de 2 hilos, cada uno de los cuales multiplicara una matriz por un entero, leidos ambos por pantalla. Se iran imprimiendo por pantalla los resultados de la multiplicacion de cada fila, asi como por que fila va el otro hilo.

Devuelve

EXIT_SUCCESS si se han realizado correctamente todas las tareas, EXIT_FAILURE si se ha producido algun error al reservar memoria, al introducir los argumentos de entrada o al ejecutar la funcion pthread_create().

4.4.2.2. void * mult_matrices (void * *parametros*)

Multiplica una matriz por un entero y muestra el resultado.

mult_matrices() recibe un puntero a void que sera una estructura (param). Calculara la multiplicacion del entero por la matriz e imprimira el resultado por pantalla. Tambien mostrara el estado del otro hilo, mostrando la fila en la que se encuentra en sus calculos.

Parámetros

parametros,puntero	a void que contendra una estructura (param).

4.5. Referencia del Archivo ejercicio6.c

Sistemas Operativos: Practica 2, ejercicio 6.

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

Dependencia gráfica adjunta para ejercicio6.c:

Funciones

- int main ()

  Funcion main del ejercicio6.

4.5.1. Descripción detallada

Sistemas Operativos: Practica 2, ejercicio 6.

Grupo 2201, Pareja 10. En este modulo se ha implementado el codigo del sexto ejercicio de la segunda practica, en el que se emplean señales para la terminacion de procesos.

Autor

Blanca Martín (blanca.martins@estudiante.uam.es)
Fernando Villar (fernando.villarg@estudiante.uam.es)

Fecha

17-03-2017

4.5.2. Documentación de las funciones

4.5.2.1. int main ( )

Funcion main del ejercicio6.

El programa consiste en la creacion de un proceso hijo, en el que en un bucle continuo se van imprimiendo un mensaje por pantalla cada 5 segundos; y en el padre, tras 30 segundos, se manda una señal de terminacion al hijo.

Devuelve

EXIT_SUCCESS si se han realizado correctamente todas las tareas, EXIT_FAILURE si se ha producido algun error al usar el fork o el kill.

4.6. Referencia del Archivo ejercicio8.c

Sistemas Operativos: Practica 2, ejercicio 8.

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

Dependencia gráfica adjunta para ejercicio8.c:

Funciones

- void impr_tiempo ()

  Imprime fecha y hora.
- void manejador (int sig)

  Manejador principal.
- void manejador2 (int sig)

  Manejador secundario.
- int main (int argc, char ∗argv[])

  Funcion main del ejercicio8.

4.6.1. Descripción detallada

Sistemas Operativos: Practica 2, ejercicio 8.

Grupo 2201, Pareja 10. En este modulo se ha implementado el codigo del octavo ejercicio de la segunda practica, que consiste en el envío cíclico de señales entre procesos creados se forma secuencial.

Autor

Blanca Martín (blanca.martins@estudiante.uam.es)
Fernando Villar (fernando.villarg@estudiante.uam.es)

Fecha

17-03-2017

4.6.2. Documentación de las funciones

4.6.2.1. void impr_tiempo ( )

Imprime fecha y hora.

impr_tiempo() imprime la fecha y hora del momento en el que es llamada.

4.6.2.2. int main (int *argc,* char * *argv[]*)

Funcion main del ejercicio8.

El programa consiste en la creacion de n procesos secuencialmente. Una vez creados, el ultimo hijo mandara una señal SIGUSR1 al primer padre, este la mandara a su hijo, y asi sucesivamente, hasta que se llegue a un numero v de vueltas realizadas. Cuando esto ocurra, el primer padre mandara una señal SIGTERM a su hijo, que se propagara hasta llegar al ultimo hijo.

Parámetros

argc,numero	de argumentos
argv,array	de strings con los argumentos (numero de procesos y numero de vueltas).

Devuelve

EXIT_SUCCESS si se han realizado correctamente todas las tareas, EXIT_FAILURE si se ha producido algun error al introducir los argumentos de entrada, al asignar los manejadores o al ejecutar la funcion fork().

4.6.2.3. void manejador (int *sig*)

Manejador principal.

Este manejador se encarga de recoger SIGUSR1 o SIGTERM y actuar como se pide en el ejercicio, excluyendo la parte de terminar el proceso.

Parámetros

sig,numero	de la señal

4.6.2.4. void manejador2 (int *sig*)

Manejador secundario.

Manejador vacio. Sirve para cuando se han acabado las v vueltas y necesitamos que el ultimo hijo informe al padre sin que el padre imprima de nuevo por pantalla, antes de empezar a terminar el resto de procesos con SIGTERM.

Parámetros

sig,numero	de la señal

Índice alfabético