

Practica 3

Generated by Doxygen 1.8.11

Contents

1	Data Structure Index	1
1.1	Data Structures	1
2	File Index	3
2.1	File List	3
3	Data Structure Documentation	5
3.1	Data Struct Reference	5
3.1.1	Detailed Description	5
3.2	info Struct Reference	5
3.2.1	Detailed Description	6
3.3	Mensaje Struct Reference	6
3.3.1	Detailed Description	6
4	File Documentation	7
4.1	cadena_montaje.c File Reference	7
4.2	ejercicio2.c File Reference	7
4.2.1	Detailed Description	8
4.2.2	Function Documentation	8
4.2.2.1	aleat_num(int inf, int sup)	8
4.2.2.2	main(void)	9
4.3	ejercicio2_solved.c File Reference	9
4.3.1	Detailed Description	10
4.3.2	Function Documentation	10
4.3.2.1	aleat_num(int inf, int sup)	10

4.3.2.2	<code>captura(int sennal)</code>	10
4.3.2.3	<code>hijo(int semid, info *inf, int i)</code>	10
4.3.2.4	<code>main(void)</code>	11
4.3.2.5	<code>padre(int semid, int n_hijos, info *inf)</code>	11
4.4	<code>ejercicio3.c</code> File Reference	11
4.4.1	Detailed Description	12
4.4.2	Function Documentation	12
4.4.2.1	<code>caracter(int i)</code>	12
4.4.2.2	<code>consumidor(int semid, Data *buffer)</code>	13
4.4.2.3	<code>main(void)</code>	13
4.4.2.4	<code>productor(int semid, Data *buffer)</code>	13
4.5	<code>ejercicio4.c</code> File Reference	13
4.5.1	Detailed Description	14
4.5.2	Function Documentation	14
4.5.2.1	<code>aleat_num(int inf, int sup)</code>	14
4.5.2.2	<code>hilo1(void *arg)</code>	15
4.5.2.3	<code>hilo2(void *arg)</code>	15
4.5.2.4	<code>main(void)</code>	15
4.6	<code>semaforos.c</code> File Reference	15
4.6.1	Detailed Description	16
4.6.2	Function Documentation	16
4.6.2.1	<code>Borrar_Semaforo(int semid)</code>	16
4.6.2.2	<code>Crear_Semaforo(key_t key, int size, int *semid)</code>	17
4.6.2.3	<code>Down_Semaforo(int id, int num_sem, int undo)</code>	18
4.6.2.4	<code>DownMultiple_Semaforo(int id, int size, int undo, int *active)</code>	18
4.6.2.5	<code>Inicializar_Semaforo(int semid, unsigned short *array)</code>	18
4.6.2.6	<code>Up_Semaforo(int id, int num_sem, int undo)</code>	19
4.6.2.7	<code>UpMultiple_Semaforo(int id, int size, int undo, int *active)</code>	19

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

Data	Estructura para la memoria compartida	5
info	Estructura para la memoria compartida	5
Mensaje	Estructura de los mensajes	6

Chapter 2

File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

cadena_montaje.c	
Cadena de montaje Ejercicio5	7
ejercicio2.c	
Ejercicio2	7
ejercicio2_solved.c	
Ejercicio2_solved	9
ejercicio3.c	
Ejercicio3	11
ejercicio4.c	
Ejercicio4	13
semaforos.c	
Biblioteca de semaforos	15
semaforos.h	??

Chapter 3

Data Structure Documentation

3.1 Data Struct Reference

Estructura para la memoria compartida.

Data Fields

- char **caracter**
- int **contador**

3.1.1 Detailed Description

Estructura para la memoria compartida.

The documentation for this struct was generated from the following file:

- [ejercicio3.c](#)

3.2 info Struct Reference

estructura para la memoria compartida

Data Fields

- char **nombre** [80]
- int **id**

3.2.1 Detailed Description

estructura para la memoria compartida

The documentation for this struct was generated from the following files:

- [ejercicio2.c](#)
- [ejercicio2_solved.c](#)

3.3 Mensaje Struct Reference

Estructura de los mensajes.

Data Fields

- long **mtime**
- int **acabado**
- char **info** [16]

3.3.1 Detailed Description

Estructura de los mensajes.

Compuesta por el tipo, la info y un int que indica si ha acabado

The documentation for this struct was generated from the following file:

- [cadena_montaje.c](#)

Chapter 4

File Documentation

4.1 cadena_montaje.c File Reference

Cadena de montaje Ejercicio5.

```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/msg.h>
#include <sys/wait.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

Include dependency graph for cadena_montaje.c:

4.2 ejercicio2.c File Reference

Ejercicio2.

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

Include dependency graph for ejercicio2.c:

Data Structures

- struct [info](#)

estructura para la memoria compartida

Macros

- `#define FILEKEY "/bin/cat"`
file para la clave de ftok
- `#define KEY 1300`
clave para el ftok
- `#define ALEAT aleat_num(1, 5)`
funcion aleatoria entre 1 y 5 para el sleep de los hijos

Functions

- `int aleat_num (int inf, int sup)`
Devuelve un numero aleatorio entre inf y sup En caso de pasar un numero negativo se cambia de signo En caso de que sup sea menor inf se permutan.
- `void captura (int sennal)`
Función capturadora de la señal.
- `int main (void)`
Main del Ejercicio2.

4.2.1 Detailed Description

Ejercicio2.

Author

Lucia Rivas Molina
Daniel Santo-Tomas Lopez

Date

14/04/2018

4.2.2 Function Documentation

4.2.2.1 `int aleat_num (int inf, int sup)`

Devuelve un numero aleatorio entre inf y sup En caso de pasar un numero negativo se cambia de signo En caso de que sup sea menor inf se permutan.

Parameters

<i>inf</i>	minimo numero aleatorio que puede salir
<i>sup</i>	maximo numero aleatorio que puede salir

Returns

el numero aleatorio

4.2.2.2 int main (void)

Main del Ejercicio2.

Returns

EXIT_FAILURE o EXIT_SUCCESS

4.3 ejercicio2_solved.c File Reference

Ejercicio2_solved.

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <string.h>
#include <errno.h>
#include <sys/shm.h>
#include <unistd.h>
#include <signal.h>
#include <sys/wait.h>
#include <time.h>
#include <stdlib.h>
#include "semaforos.h"
```

Include dependency graph for ejercicio2_solved.c:

Data Structures

- struct [info](#)
estructura para la memoria compartida

Macros

- #define [FILEKEY](#) "/bin/cat"
file para la clave de ftok
- #define [KEY](#) 1300
clave para el ftok
- #define [ALEAT](#) [aleat_num](#)(1, 5)
funcion aleatoria entre 1 y 5 para el sleep de los hijos

Functions

- int [aleat_num](#) (int inf, int sup)
Devuelve un numero aleatorio entre inf y sup En caso de pasar un numero negativo se cambia de signo En caso de que sup sea menor inf se permutan.
- void [captura](#) (int sennal)
Función capturadora de la señal.
- int [padre](#) (int semid, int n_hijos, [info](#) *inf)
Funcion que realiza el padre.
- int [hijo](#) (int semid, [info](#) *inf, int i)
Funcion que realiza el hijo.
- int [main](#) (void)
Main del Ejercicio2_solved.

4.3.1 Detailed Description

Ejercicio2_solved.

Author

Lucia Rivas Molina
Daniel Santo-Tomas Lopez

Date

14/04/2018

4.3.2 Function Documentation

4.3.2.1 `int aleat_num (int inf, int sup)`

Devuelve un numero aleatorio entre inf y sup En caso de pasar un numero negativo se cambia de signo En caso de que sup sea menor inf se permutan.

Parameters

<i>inf</i>	minimo numero aleatorio que puede salir
<i>sup</i>	maximo numero aleatorio que puede salir

Returns

el numero aleatorio

4.3.2.2 `void captura (int sennal)`

Función capturadora de la señal.

Parameters

<i>sennal</i>	: señal mandada En caso de mandar SIGINT el programa se encarga de borrar los semaforos y la memoria compartida
---------------	---

4.3.2.3 `int hijo (int semid, info * inf, int i)`

Funcion que realiza el hijo.

Parameters

<i>semid</i>	: semid de los semaforos
<i>i</i>	: numero de hijo que es
	<i>inf</i> : estructura de la memoria compartida

Returns

OK o ERROR

4.3.2.4 int main (void)

Main del Ejercicio2_solved.

Returns

EXIT_FAILURE o EXIT_SUCCESS

4.3.2.5 int padre (int semid, int n_hijos, info * inf)

Funcion que realiza el padre.

Parameters

<i>semid</i>	: semid de los semaforos
<i>n_hijos</i>	: numero de hijos que ha creado
	inf : estructura de la memoria compartida

Returns

OK o ERROR

4.4 ejercicio3.c File Reference

Ejercicio3.

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <string.h>
#include <errno.h>
#include <sys/shm.h>
#include <unistd.h>
#include <signal.h>
#include <sys/wait.h>
#include <time.h>
#include <stdlib.h>
#include "semaforos.h"
```

Include dependency graph for ejercicio3.c:

Data Structures

- struct [Data](#)

Estructura para la memoria compartida.

Macros

- `#define FILEKEY "/bin/cat"`
file para la clave
- `#define KEY 1300`
key para los semaforos y la memoria compartida
- `#define TAM 36`
tamaño del abecedario y los numeros

Functions

- `int productor (int semid, Data *buffer)`
Funcion del productor que produce una letra o numero y la guarda en memoria compartida.
- `int consumidor (int semid, Data *buffer)`
Funcion del consumidor que lee la memoria compartida y muestra el caracter por pantalla.
- `char caracter (int i)`
Funcion que convierte el caracter correspondiente al entero i pasado como parámetro, las letras o los números dependiendo del valor de la i.
- `int main (void)`
Main del ejercicio 3.

4.4.1 Detailed Description

Ejercicio3.

Author

Lucia Rivas Molina
Daniel Santo-Tomas Lopez

Date

14/04/2018

4.4.2 Function Documentation

4.4.2.1 `char caracter (int i)`

Funcion que convierte el caracter correspondiente al entero i pasado como parámetro, las letras o los números dependiendo del valor de la i.

Parameters

<i>i</i>	numero
----------	--------

Returns

-1 en caso de error o el caracter correspondiente a i

4.4.2.2 int consumidor (int *semid*, Data * *buffer*)

Funcion del consumidor que lee la memoria compartida y muestra el caracter por pantalla.

Parameters

<i>semid</i>	: id del semaforo
<i>buffer</i>	: la memoria compartida

Returns

OK o ERROR

4.4.2.3 int main (void)

Main del ejercicio 3.

Returns

EXIT_FAILURE o EXIT_SUCCESS

4.4.2.4 int productor (int *semid*, Data * *buffer*)

Funcion del productor que produce una letra o numero y la guarda en memoria compartida.

Parameters

<i>semid</i>	: id del semaforo
<i>buffer</i>	: la memoria compartida

Returns

OK o ERROR

4.5 ejercicio4.c File Reference

Ejercicio4.

```
#include <sys/mman.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <pthread.h>
#include <time.h>
```

Include dependency graph for ejercicio4.c:

Macros

- `#define INF 100`
infimo para el numero aleatorio creado en el fichero
- `#define MID 1000`
supremo para el aleatorio creado en el fichero e infimo para el numero de caracteres a escribir
- `#define SUP 2000`
supremo para el aleatorio de numero de caracteres a escribir

Functions

- `int aleat_num (int inf, int sup)`
Devuelve un numero aleatorio entre inf y sup En caso de pasar un numero negativo se cambia de signo En caso de que sup sea menor inf se permutan.
- `void * hilo1 (void *arg)`
Función que realiza el hilo 1 abriendo el fichero arg y llenandolo con una cantidad aleatoria de numeros aleatorios.
- `void * hilo2 (void *arg)`
Función que realiza el hilo 2 mapeando el fichero modificado por el hilo 1 y convirtiendo las comas en espacios.
- `int main (void)`
Main del ejercicio 4.

4.5.1 Detailed Description

Ejercicio4.

Author

Lucia Rivas Molina
Daniel Santo-Tomas Lopez

Date

14/04/2018

4.5.2 Function Documentation

4.5.2.1 `int aleat_num (int inf, int sup)`

Devuelve un numero aleatorio entre inf y sup En caso de pasar un numero negativo se cambia de signo En caso de que sup sea menor inf se permutan.

Parameters

<i>inf</i>	minimo numero aleatorio que puede salir
<i>sup</i>	maximo numero aleatorio que puede salir

Returns

el numero aleatorio

4.5.2.2 void * hilo1 (void * *arg*)

Función que realiza el hilo 1 abriendo el fichero *arg* y llenandolo con una cantidad aleatoria de numeros aleatorios.

Parameters

<i>arg</i>	: path del fichero a abrir
------------	----------------------------

4.5.2.3 void * hilo2 (void * *arg*)

Función que realiza el hilo 2 mapeando el fichero modificado por el hilo 1 y convirtiendo las comas en espacios.

Parameters

<i>arg</i>	: path del fichero a abrir
------------	----------------------------

4.5.2.4 int main (void)

Main del ejercicio 4.

Returns

EXIT_FAILURE o EXIT_SUCCESS

4.6 semaforos.c File Reference

Biblioteca de semaforos.

```
#include "semaforos.h"  
Include dependency graph for semaforos.c:
```

Functions

- int [Inicializar_Semaforo](#) (int semid, unsigned short *array)
Inicializa los semaforos indicados.
- int [Borrar_Semaforo](#) (int semid)
Elimina los semaforos indicados.
- int [Crear_Semaforo](#) (key_t key, int size, int *semid)
Crea los semaforos indicados.
- int [Down_Semaforo](#) (int id, int num_sem, int undo)
Hace un down de un semaforo.
- int [DownMultiple_Semaforo](#) (int id, int size, int undo, int *active)
Hace un down de un grupo de semaforos.
- int [Up_Semaforo](#) (int id, int num_sem, int undo)
Hace un up de un semaforo.
- int [UpMultiple_Semaforo](#) (int id, int size, int undo, int *active)
Hace un up de un grupo de semaforos.

4.6.1 Detailed Description

Biblioteca de semaforos.

Biblioteca de semaforos para las prácticas de SOPER

Author

Lucia Rivas Molina lucia.rivasmolina@estudiante.uam.es
Daniel Santo-Tomas Lopez daniel.santo-tomas@estudiante.uam.es

Date

20/03/2018

4.6.2 Function Documentation

4.6.2.1 int Borrar_Semaforo (int semid)

Elimina los semaforos indicados.

Parameters

<i>semid</i>	: identificador del grupo de semaforos
--------------	--

Returns

OK / ERROR

4.6.2.2 int Crear_Semaforo (key_t *key*, int *size*, int * *semid*)

Crea los semaforos indicados.

Parameters

<i>key</i>	: clave precompartida del semaforo
<i>size</i>	: numero de semaforos a crear
<i>semid</i>	: identificador del grupo de semaforos

Returns

OK / ERROR

4.6.2.3 int Down_Semaforo (int *id*, int *num_sem*, int *undo*)

Hace un down de un semaforo.

Parameters

<i>id</i>	: identificador del grupo de semaforos
<i>num_sem</i>	: numero del semaforo del cual hacemos el down
<i>undo</i>	: flags

Returns

OK / ERROR

4.6.2.4 int DownMultiple_Semaforo (int *id*, int *size*, int *undo*, int * *active*)

Hace un down de un grupo de semaforos.

Parameters

<i>id</i>	: identificador del grupo de semaforos
<i>size</i>	: numero de semaforos para hacer un down
<i>undo</i>	: flags
<i>active</i>	: semaforos involucrados

Returns

OK / ERROR

4.6.2.5 int Inicializar_Semaforo (int *semid*, unsigned short * *array*)

Inicializa los semaforos indicados.

Parameters

<i>semid</i>	: identificador del grupo de semaforos
<i>array</i>	: valores iniciales del semaforo

Returns

OK / ERROR

4.6.2.6 int Up_Semaforo (int *id*, int *num_sem*, int *undo*)

Hace un up de un semaforo.

Parameters

<i>id</i>	: identificador del grupo de semaforos
<i>num_sem</i>	: numero del semaforo del cual hacemos el down
<i>undo</i>	: flags

Returns

OK / ERROR

4.6.2.7 int UpMultiple_Semaforo (int *id*, int *size*, int *undo*, int * *active*)

Hace un up de un grupo de semaforos.

Parameters

<i>id</i>	: identificador del grupo de semaforos
<i>size</i>	: numero de semaforos para hacer un down
<i>undo</i>	: flags
<i>active</i>	: semaforos involucrados

Returns

OK / ERROR

Index

aleat_num
 ejercicio2.c, [8](#)
 ejercicio2_solved.c, [10](#)
 ejercicio4.c, [14](#)

Borrar_Semaforo
 semaforos.c, [16](#)

cadena_montaje.c, [7](#)

captura
 ejercicio2_solved.c, [10](#)

caracter
 ejercicio3.c, [12](#)

consumidor
 ejercicio3.c, [13](#)

Crear_Semaforo
 semaforos.c, [16](#)

Data, [5](#)

Down_Semaforo
 semaforos.c, [18](#)

DownMultiple_Semaforo
 semaforos.c, [18](#)

ejercicio2.c, [7](#)
 aleat_num, [8](#)
 main, [8](#)

ejercicio2_solved.c, [9](#)
 aleat_num, [10](#)
 captura, [10](#)
 hijo, [10](#)
 main, [11](#)
 padre, [11](#)

ejercicio3.c, [11](#)
 caracter, [12](#)
 consumidor, [13](#)
 main, [13](#)
 productor, [13](#)

ejercicio4.c, [13](#)
 aleat_num, [14](#)
 hilo1, [15](#)
 hilo2, [15](#)
 main, [15](#)

hijo
 ejercicio2_solved.c, [10](#)

hilo1
 ejercicio4.c, [15](#)

hilo2
 ejercicio4.c, [15](#)

info, [5](#)

Inicializar_Semaforo
 semaforos.c, [18](#)

main
 ejercicio2.c, [8](#)
 ejercicio2_solved.c, [11](#)
 ejercicio3.c, [13](#)
 ejercicio4.c, [15](#)

Mensaje, [6](#)

padre
 ejercicio2_solved.c, [11](#)

productor
 ejercicio3.c, [13](#)

semaforos.c, [15](#)
 Borrar_Semaforo, [16](#)
 Crear_Semaforo, [16](#)
 Down_Semaforo, [18](#)
 DownMultiple_Semaforo, [18](#)
 Inicializar_Semaforo, [18](#)
 Up_Semaforo, [19](#)
 UpMultiple_Semaforo, [19](#)

Up_Semaforo
 semaforos.c, [19](#)

UpMultiple_Semaforo
 semaforos.c, [19](#)