# My Project

Generated by Doxygen 1.8.11

# Contents

# Chapter 1

# File Index

## 1.1 File List

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

# Chapter 2

# File Documentation

## 2.1 ejercicio4a.c File Reference

fichero que contiene ej4a de la practica 1 de SOPER para el estudio de la existencia de hijos huerfanos.

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

Include dependency graph for ejercicio4a.c:

## 2.2 ejercicio4b.c File Reference

fichero que contiene ej4b de la practica 1 de SOPER para el estudio de la existencia de hijos huerfanos.

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

Include dependency graph for ejercicio4b.c:

### Macros

- #define **NUM_PROC** 3

### Functions

- int main ()

### 2.2.1 Detailed Description

fichero que contiene ej4b de la practica 1 de SOPER para el estudio de la existencia de hijos huerfanos.

**Author**

Lucia Asencio y Rodrigo de Pool

**Date**

28-2-2017

### 2.2.2 Function Documentation

#### 2.2.2.1 int main ( void )

Cada proceso genera un hijo, independientemente de si tiene ya o no, hasta que el padre alcanza NUM_PROC hijos. Cada hijo imprime pid y ppid. Para controlar huerfanos, cada padre imprime su pid. Ademas, cada proceso hace un wait antes de terminar la ejecucion.

## 2.3 ejercicio4SinHuerfanos.c File Reference

fichero que contiene una solucion alternativa para no tener procesos hijos huerfanos. Practica 1 de SOPER

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

Include dependency graph for ejercicio4SinHuerfanos.c:

### Macros

- #define **NUM_PROC** 3

### Functions

- int main ()

### 2.3.1 Detailed Description

fichero que contiene una solucion alternativa para no tener procesos hijos huerfanos. Practica 1 de SOPER

**Author**

Lucia Asencio y Rodrigo de Pool

**Date**

28-2-2017

### 2.3.2 Function Documentation

#### 2.3.2.1 int main ( void )

Se prueba la funcion fork generando hijos exponencialmente. Cada proceso hijo imprimira su informacion. A diferencia de los otros dos ejercicios en este nos encargamos de no dejar a ningun proceso zombie.

## 2.4 ejercicio5a.c File Reference

fichero que contiene ej5a de la practica 1 de SOPER.

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

Include dependency graph for ejercicio5a.c:

### Macros

- #define **NUM_PROC** 3

### Functions

- int main (void)

### 2.4.1 Detailed Description

fichero que contiene ej5a de la practica 1 de SOPER.

**Author**

Lucia Asencio y Rodrigo de Pool

**Date**

28-2-2017

### 2.4.2 Function Documentation

#### 2.4.2.1 int main ( void )

Un proceso padre genera un hijo que a su vez genera otro y asi NUM_PROC veces, cada proceso hijo imprimira su informacion.

## 2.5 ejercicio5b.c File Reference

fichero que contiene ej5b de la practica 1 de SOPER.

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

Include dependency graph for ejercicio5b.c:

### Macros

- #define **NUM_PROC** 3

### Functions

- int main (void)

### 2.5.1 Detailed Description

fichero que contiene ej5b de la practica 1 de SOPER.

**Author**

Lucia Asencio y Rodrigo de Pool

**Date**

28-2-2017

### 2.5.2 Function Documentation

#### 2.5.2.1 int main ( void )

Programa que crea tres procesos hijos procedentes de un padre e imprime por pantalla la informacion de cada proceso hijo.

## 2.6 ejercicio6.c File Reference

fichero que contiene ej6 de la practica 1 de SOPER para el estudio del uso de memoria entre procesos padre e hijo.

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

Include dependency graph for ejercicio6.c:

### Functions

- int main ()

### 2.6.1 Detailed Description

fichero que contiene ej6 de la practica 1 de SOPER para el estudio del uso de memoria entre procesos padre e hijo.

**Author**

Lucia Asencio y Rodrigo de Pool

**Date**

28-2-2017

### 2.6.2 Function Documentation

#### 2.6.2.1 int main ( void )

Funcion main: Este main reserva memoria antes de hacer un fork(). Tanto el padre como el hijo alteran el contenido de esta memoria, y se comprueba que ambos deben liberarla despues, y que uno no tiene acceso a la memoria del otro.

## 2.7 ejercicio8.c File Reference

fichero que contiene ej8 de la practica 1 de SOPER. Ejercicio con el objetivo de entender el funcionamiento de las funciones 'exec'.

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

Include dependency graph for ejercicio8.c:

### Functions

- void execute (char *func, char *flag)
- int main (int argc, char *argv[ ])

### 2.7.1 Detailed Description

fichero que contiene ej8 de la practica 1 de SOPER. Ejercicio con el objetivo de entender el funcionamiento de las funciones 'exec'.

**Author**

Lucia Asencio y Rodrigo de Pool

**Date**

28-2-2017

### 2.7.2 Function Documentation

#### 2.7.2.1 void execute ( char * *func,* char * *flag* )

Esta funcion se encarga de ejecutar un comando con la funcion indicada a traves del flag.

**Parameters**

| | |
|---|---|
| *func* | indica el nombre del comando a ejecutar o su path. |
| *flag* | indica que funcion se utilizara para ejecutar func. |

#### 2.7.2.2 int main ( int *argc,* char * *argv[ ]* )

Funcion principal que se encarga de crear procesos hijos para ejecutar los comandos pasados como argumentos. Nota importante: Si se quiere ejecutar un proceso dado solo su nombre debido a que esta en PATH, se tiene que ejecutar con el comando correcto (execvp, execlp).

**Parameters**

| | |
|---|---|
| *argv* | lista de comandos a ejecutar y de ultimo un indicador de la funcion que se quiere utilizar para ejecutar los comandos. |

## 2.8 ejercicio9.c File Reference

fichero que contiene ej9 de la practica 1 de SOPER para el estudio de la comunicacion entre procesos mediante pipes.

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

Include dependency graph for ejercicio9.c:

## Functions

- int main ()

### 2.8.1 Detailed Description

fichero que contiene ej9 de la practica 1 de SOPER para el estudio de la comunicacion entre procesos mediante pipes.

**Author**

Lucia Asencio y Rodrigo de Pool

**Date**

28-2-2017

### 2.8.2 Function Documentation

#### 2.8.2.1 int main ( void )

Funcion main del ejercicio 9 Pide por pantalla dos valores double que seran almacenados por el proceso padre. El proceso padre genera cuatro hijos, a los que envua los valores escaneados a traves de un pipe. Cada hijo realiza una operacion y envia el resultado al padre a traves de otro pipe. El padre imprime cada uno de los resultados extraidos del pipe.

# Index