

## My Project

Generated by Doxygen 1.8.14



# Contents

<b>1</b>	<b>Class Index</b>	<b>1</b>
1.1	Class List . . . . .	1
<b>2</b>	<b>File Index</b>	<b>3</b>
2.1	File List . . . . .	3
<b>3</b>	<b>Class Documentation</b>	<b>5</b>
3.1	_estructura Struct Reference . . . . .	5
3.1.1	Detailed Description . . . . .	5
3.1.2	Member Data Documentation . . . . .	5
3.1.2.1	cadena . . . . .	5
3.1.2.2	entero . . . . .	6
3.2	param Struct Reference . . . . .	6
3.2.1	Detailed Description . . . . .	6
3.2.2	Member Data Documentation . . . . .	6
3.2.2.1	dim . . . . .	6
3.2.2.2	id . . . . .	6
3.2.2.3	matriz . . . . .	6
3.2.2.4	mult . . . . .	6

<b>4 File Documentation</b>	<b>7</b>
4.1 ejercicio12a.c File Reference	7
4.1.1 Detailed Description	8
4.1.2 Macro Definition Documentation	8
4.1.2.1 NUMERO_PROCESOS	8
4.1.3 Typedef Documentation	8
4.1.3.1 estructura	8
4.1.4 Function Documentation	8
4.1.4.1 calcula_primos()	8
4.1.4.2 main()	9
4.2 ejercicio12b.c File Reference	9
4.2.1 Detailed Description	10
4.2.2 Macro Definition Documentation	10
4.2.2.1 NUMERO_HILOS	10
4.2.3 Typedef Documentation	10
4.2.3.1 estructura	11
4.2.4 Function Documentation	11
4.2.4.1 calcula_primos()	11
4.2.4.2 main()	11
4.3 ejercicio13.c File Reference	12
4.3.1 Detailed Description	12
4.3.2 Macro Definition Documentation	13
4.3.2.1 ARRAY_SIZE	13
4.3.2.2 BUFFER_SIZE	13
4.3.3 Typedef Documentation	13
4.3.3.1 Param	13
4.3.4 Function Documentation	13
4.3.4.1 calcula_matriz()	13
4.3.4.2 main()	14
4.4 ejercicio4a.c File Reference	14

4.4.1	Detailed Description	14
4.4.2	Macro Definition Documentation	15
4.4.2.1	NUM_PROC	15
4.5	ejercicio4b.c File Reference	15
4.5.1	Detailed Description	15
4.5.2	Macro Definition Documentation	15
4.5.2.1	NUM_PROC	16
4.6	ejercicio5a.c File Reference	16
4.6.1	Detailed Description	16
4.6.2	Macro Definition Documentation	16
4.6.2.1	NUM_PROC	16
4.7	ejercicio5b.c File Reference	17
4.7.1	Detailed Description	17
4.7.2	Macro Definition Documentation	17
4.7.2.1	NUM_PROC	17
4.8	ejercicio6.c File Reference	18
4.8.1	Detailed Description	18
4.8.2	Macro Definition Documentation	18
4.8.2.1	MAX_CAD	18
4.9	ejercicio8.c File Reference	19
4.9.1	Detailed Description	19
4.9.2	Macro Definition Documentation	19
4.9.2.1	MAX_PATH	19
4.10	ejercicio9.c File Reference	20
4.10.1	Detailed Description	20
4.10.2	Macro Definition Documentation	20
4.10.2.1	BUFFER_SIZE	21
4.10.2.2	ESCRITURA	21
4.10.2.3	LECTURA	21
4.10.2.4	MENSAJE_SIZE	21
4.10.3	Function Documentation	21
4.10.3.1	factorial()	21
4.10.3.2	main()	22



# Chapter 1

## Class Index

### 1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">_estructura</a>	Estructura generada al principio de la ejecución . . . . .	<a href="#">5</a>
<a href="#">param</a>	Estructura de parámetros de un hilo . . . . .	<a href="#">6</a>





## Chapter 2

# File Index

### 2.1 File List

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

<a href="#">ejercicio12a.c</a>	Ejercicio para la comparación de tiempo: procesos . . . . .	7
<a href="#">ejercicio12b.c</a>	Ejercicio para la comparación de tiempo: hilos . . . . .	9
<a href="#">ejercicio13.c</a>	Ejercicio sobre paso de parámetros en funciones . . . . .	12
<a href="#">ejercicio4a.c</a>	Ejercicio 4a de la Práctica . . . . .	14
<a href="#">ejercicio4b.c</a>	Ejercicio 4b de la Práctica . . . . .	15
<a href="#">ejercicio5a.c</a>	Ejercicio 5a de la Práctica . . . . .	16
<a href="#">ejercicio5b.c</a>	Ejercicio 5b de la Práctica . . . . .	17
<a href="#">ejercicio6.c</a>	Ejercicio 6 de la Práctica . . . . .	18
<a href="#">ejercicio8.c</a>	Ejercicio 8 de la Práctica . . . . .	19
<a href="#">ejercicio9.c</a>	Ejercicio para la comunicacion entre procesos mediante tuberias . . . . .	20



## Chapter 3

# Class Documentation

### 3.1 `_estructura` Struct Reference

estructura generada al principio de la ejecución

#### Public Attributes

- char \* `cadena`
- int \* `entero`

#### 3.1.1 Detailed Description

estructura generada al principio de la ejecución

Esta estructura almacena una cadena de 100 caracteres y un entero. Se usa para comparar el uso de memoria entre los hilos y los procesos

#### 3.1.2 Member Data Documentation

##### 3.1.2.1 `cadena`

```
char * _estructura::cadena
```

Cadena de 100 caracteres

### 3.1.2.2 entero

```
int * _estructura::entero
```

Puntero a un entero

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

- [ejercicio12a.c](#)
- [ejercicio12b.c](#)

## 3.2 param Struct Reference

Estructura de parámetros de un hilo.

### Public Attributes

- int \*\* [matriz](#)
- int [mult](#)
- int [dim](#)
- int [id](#)

### 3.2.1 Detailed Description

Estructura de parámetros de un hilo.

Esta estructura almacena la información necesaria para los hilos en este ejercicio

### 3.2.2 Member Data Documentation

#### 3.2.2.1 dim

```
int param::dim
```

Dimension de las matrices

#### 3.2.2.2 id

```
int param::id
```

Identificador del hilo

#### 3.2.2.3 matriz

```
int** param::matriz
```

Matriz

#### 3.2.2.4 mult

```
int param::mult
```

Multiplicador

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

- [ejercicio13.c](#)

## Chapter 4

# File Documentation

### 4.1 ejercicio12a.c File Reference

Ejercicio para la comparación de tiempo: procesos.

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

#### Classes

- struct [\\_estructura](#)  
*estructura generada al principio de la ejecución*

#### Macros

- #define [NUMERO\\_PROCESOS](#) 100

#### Typedefs

- typedef struct [\\_estructura](#) [estructura](#)  
*estructura generada al principio de la ejecución*

#### Functions

- void [calcula\\_primos](#) (int N)  
*calcula los n primos numeros primos*
- int [main](#) (int argc, char \*\*argv)  
*Función principal del programa.*

### 4.1.1 Detailed Description

Ejercicio para la comparación de tiempo: procesos.

Este fichero, en conjunto con el [ejercicio12b.c](#), sirven para comparar la eficiencia de los threads comparados con los procesos

#### Author

Miguel Arconada Manteca y José Manuel Chacón Aguilera

#### Date

8-3-2018

### 4.1.2 Macro Definition Documentation

#### 4.1.2.1 NUMERO\_PROCESOS

```
#define NUMERO_PROCESOS 100
```

Numero total de procesos que queremos crear

### 4.1.3 Typedef Documentation

#### 4.1.3.1 estructura

```
typedef struct _estructura estructura
```

estructura generada al principio de la ejecución

Esta estructura almacena una cadena de 100 caracteres y un entero. Se usa para comparar el uso de memoria entre los hilos y los procesos

### 4.1.4 Function Documentation

#### 4.1.4.1 calcula\_primos()

```
void calcula_primos (  
    int N )
```

calcula los n primos numeros primos

Esta funcion calcula los n primero numeros primos

### Parameters

<i>N</i>	numero de primos a calcular
----------	-----------------------------

### Returns

void

#### 4.1.4.2 main()

```
int main (
    int argc,
    char ** argv )
```

Función principal del programa.

Este programa lanza 100 procesos simultáneamente, en los que se calculan los N primeros primos

### Parameters

<i>argc</i>	numero de parametros de entrada
<i>argv</i>	array de cada parametro. El primero es el nombre del programa, y el segundo es N

### Returns

0 si todo se ejecuta correctamente, y -1 en cualquier otro caso

## 4.2 ejercicio12b.c File Reference

Ejercicio para la comparación de tiempo: hilos.

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

### Classes

- struct [\\_estructura](#)

*estructura generada al principio de la ejecución*

## Macros

- `#define NUMERO_HILOS 100`

## Typedefs

- `typedef struct _estructura estructura`  
*estructura generada al principio de la ejecución*

## Functions

- `void * calcula_primos (void *arg)`  
*calcula los N primeros primos*
- `int main (int argc, char **argv)`  
*Función principal del programa.*

### 4.2.1 Detailed Description

Ejercicio para la comparación de tiempo: hilos.

Este fichero, en conjunto con el [ejercicio12b.c](#), sirven para comparar la eficiencia de los threads comparados con los procesos

#### Author

Miguel Arconada Manteca y José Manuel Chacón Aguilera

#### Date

8-3-2018

### 4.2.2 Macro Definition Documentation

#### 4.2.2.1 NUMERO\_HILOS

```
#define NUMERO_HILOS 100
```

Numero total de hilos que queremos crear

### 4.2.3 Typedef Documentation



#### 4.2.3.1 estructura

```
typedef struct _estructura estructura
```

estructura generada al principio de la ejecución

Esta estructura almacena una cadena de 100 caracteres y un entero. Se usa para comparar el uso de memoria entre los hilos y los procesos

### 4.2.4 Function Documentation

#### 4.2.4.1 calcula\_primos()

```
void * calcula_primos (  
    void * arg )
```

calcula los N primeros primos

calcula\_primos calcula los N primeros números primos

##### Parameters

<i>arg</i>	puntero a un entero que contiene N hecho casting a void*
------------	--

##### Returns

void

#### 4.2.4.2 main()

```
int main (  
    int argc,  
    char ** argv )
```

Función principal del programa.

Este programa lanza 100 hilos simultáneamente, en los que se calculan los N primeros primos

##### Parameters

<i>argc</i>	numero de parametros de entrada
<i>argv</i>	array de cada parametro. El primero es el nombre del programa, y el segundo es N

## Returns

0 si todo se ejecuta correctamente, y -1 en cualquier otro caso

## 4.3 ejercicio13.c File Reference

Ejercicio sobre paso de parámetros en funciones.

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

## Classes

- struct [param](#)  
*Estructura de parámetros de un hilo.*

## Macros

- #define [BUFFER\\_SIZE](#) 256
- #define [ARRAY\\_SIZE](#) 30

## Typedefs

- typedef struct [param](#) [Param](#)  
*Estructura de parámetros de un hilo.*

## Functions

- void \* [calcula\\_matriz](#) (void \*arg)  
*multiplica una matriz por un escalar*
- int [main](#) ()  
*Función principal del programa.*

### 4.3.1 Detailed Description

Ejercicio sobre paso de parámetros en funciones.

Este fichero contiene nuestra solución al ejercicio 13 enunciado en el pdf de la práctica

## Author

Miguel Arconada Manteca y José Manuel Chacón Aguilera

## Date

8-3-2018

## 4.3.2 Macro Definition Documentation

### 4.3.2.1 ARRAY\_SIZE

```
#define ARRAY_SIZE 30
```

Tamaño maximo de los buffers valores

### 4.3.2.2 BUFFER\_SIZE

```
#define BUFFER_SIZE 256
```

Tamaño maximo del buffer cadena\_aux

## 4.3.3 Typedef Documentation

### 4.3.3.1 Param

```
typedef struct param Param
```

Estructura de parámetros de un hilo.

Esta estructura almacena la información necesaria para los hilos en este ejercicio

## 4.3.4 Function Documentation

### 4.3.4.1 calcula\_matriz()

```
void * calcula_matriz (  
    void * arg )
```

multiplica una matriz por un escalar

calcula\_matriz multiplica una matriz por un escalar. Está almacenada en forma de puntero a función para ser ejecutada por varios threads simultáneamente.

#### Parameters

<i>arg</i>	estructura del tipo Param que contiene la matriz, el escalar, la dimension y un identificador del hilo
------------	--

**Returns**

void

**4.3.4.2 main()**

```
int main (
    void )
```

Función principal del programa.

Este programa pide por consola una dimensión, dos multiplicadores y dos matrices, y lanza dos hilos que multiplican cada matriz por los escalares, y que van imprimiendo cada línea según la calculan

**Parameters**

void	
------	--

**Returns**

0 si todo se ejecuta correctamente, y -1 en cualquier otro caso

## 4.4 ejercicio4a.c File Reference

Ejercicio 4a de la Práctica.

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>
```

**Macros**

- #define NUM\_PROC 6

**Functions**

- int **main** (void)

### 4.4.1 Detailed Description

Ejercicio 4a de la Práctica.

En este ejercicio vemos el ejemplo de ejecución de un programa que lanza procesos hijos.

**Author**

José Manuel Chacón Aguilera y Miguel Arconada Manteca

**Date**

8-3-2018

## 4.4.2 Macro Definition Documentation

### 4.4.2.1 NUM\_PROC

```
#define NUM_PROC 6
```

Número de iteraciones del bucle creador de procesos

## 4.5 ejercicio4b.c File Reference

Ejercicio 4b de la Práctica.

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

### Macros

- #define [NUM\\_PROC](#) 6

### Functions

- int **main** (void)

### 4.5.1 Detailed Description

Ejercicio 4b de la Práctica.

En este ejercicio vemos el ejemplo de ejecución de un programa que lanza procesos hijos.

#### Author

José Manuel Chacón Aguilera y Miguel Arconada Manteca

#### Date

8-3-2018

## 4.5.2 Macro Definition Documentation

#### 4.5.2.1 NUM\_PROC

```
#define NUM_PROC 6
```

Número de iteraciones del bucle creador de procesos

## 4.6 ejercicio5a.c File Reference

Ejercicio 5a de la Práctica.

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

### Macros

- `#define NUM_PROC 6`

### Functions

- `int main (void)`

#### 4.6.1 Detailed Description

Ejercicio 5a de la Práctica.

En este ejercicio vemos el ejemplo de ejecución de un programa que lanza procesos hijos.

#### Author

José Manuel Chacón Aguilera y Miguel Arconada Manteca

#### Date

8-3-2018

#### 4.6.2 Macro Definition Documentation

##### 4.6.2.1 NUM\_PROC

```
#define NUM_PROC 6
```

Número de iteraciones del bucle creador de procesos

## 4.7 ejercicio5b.c File Reference

Ejercicio 5b de la Práctica.

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

### Macros

- `#define NUM_PROC 6`

### Functions

- `int main (void)`

#### 4.7.1 Detailed Description

Ejercicio 5b de la Práctica.

En este ejercicio vemos el ejemplo de ejecución de un programa que lanza procesos hijos.

#### Author

José Manuel Chacón Aguilera y Miguel Arconada Manteca

#### Date

8-3-2018

#### 4.7.2 Macro Definition Documentation

##### 4.7.2.1 NUM\_PROC

```
#define NUM_PROC 6
```

Número de iteraciones del bucle creador de procesos

## 4.8 ejercicio6.c File Reference

Ejercicio 6 de la Práctica.

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

### Macros

- `#define MAX_CAD 128`

### Functions

- `int main ()`

#### 4.8.1 Detailed Description

Ejercicio 6 de la Práctica.

En este ejercicio combinamos cuestiones teóricas sobre reserva de memoria dinamica y ejecución del programa en procesos distintos

#### Author

José Manuel Chacón Aguilera y Miguel Arconada Manteca

#### Date

8-3-2018

#### 4.8.2 Macro Definition Documentation

##### 4.8.2.1 MAX\_CAD

```
#define MAX_CAD 128
```

Tamaño máximo de las cadenas inicializadas



## 4.9 ejercicio8.c File Reference

Ejercicio 8 de la Práctica.

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

### Macros

- `#define MAX_PATH 256`

### Functions

- `int main (int argc, char **argv)`

#### 4.9.1 Detailed Description

Ejercicio 8 de la Práctica.

En este ejercicio lanzamos tantos procesos hijo como especificados en los parámetros de entrada de la función `main` y además hacemos que estos procesos hijos ejecuten un código distinto al de este programa todo esto lo conseguimos con una combinación de las rutinas `fork()` y la familia de rutinas `exec()`.

#### Author

José Manuel Chacón Aguilera y Miguel Arconada Manteca

#### Date

8-3-2018

#### 4.9.2 Macro Definition Documentation

##### 4.9.2.1 MAX\_PATH

```
#define MAX_PATH 256
```

Tamaño máximo de la ruta

## 4.10 ejercicio9.c File Reference

Ejercicio para la comunicacion entre procesos mediante tuberias.

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

### Macros

- `#define LECTURA 0`
- `#define ESCRITURA 1`
- `#define BUFFER_SIZE 512`
- `#define MENSAJE_SIZE 512`

### Functions

- `int factorial (int n)`  
*calcula el factorial de un numero*
- `int main (int argc, char **argv)`  
*Función principal del programa.*

#### 4.10.1 Detailed Description

Ejercicio para la comunicacion entre procesos mediante tuberias.

En este fichero hacemos uso de las pipes para implementar la comunicacion entre un proceso padre y cuatro procesos hijos, a los cuales les manda dos enteros, y estos devuelven el resultado de una operacion con ellos

#### Author

Miguel Arconada Manteca y José Manuel Chacón Aguilera

#### Date

8-3-2018

#### 4.10.2 Macro Definition Documentation

#### 4.10.2.1 BUFFER\_SIZE

```
#define BUFFER_SIZE 512
```

Tamaño máximo de los buffers

#### 4.10.2.2 ESCRITURA

```
#define ESCRITURA 1
```

Macro que define la zona de escritura de la pipe

#### 4.10.2.3 LECTURA

```
#define LECTURA 0
```

Macro que define la zona de lectura de la pipe

#### 4.10.2.4 MENSAJE\_SIZE

```
#define MENSAJE_SIZE 512
```

Tamaño máximo de los mensajes

### 4.10.3 Function Documentation

#### 4.10.3.1 factorial()

```
int factorial (  
    int n )
```

calcula el factorial de un número

factorial calcula el factorial de un número pasado como argumento. Se calcula de forma recursiva, devolviendo el número multiplicado por el factorial del anterior, y con la condición de parada de que  $\text{factorial}(1) = 1$

##### Parameters

<i>arg</i>	estructura del tipo Param que contiene la matriz, el escalar, la dimensión y un identificador del hilo
------------	--

##### Returns

void

#### 4.10.3.2 main()

```
int main (
    int argc,
    char ** argv )
```

Función principal del programa.

Este programa crea cuatro procesos hijos que se comunican con el padre, e intercambian mensajes con numeros y el resultado de una operacion entre ellos diferente para cada hijo

##### Parameters

<i>argc</i>	y <i>argv</i>
-------------	---------------

##### Returns

0 si todo se ejecuta correctamente, y -1 en cualquier otro caso

# Index

- [\\_estructura](#), [5](#)
    - [cadena](#), [5](#)
    - [entero](#), [5](#)
- [ARRAY\\_SIZE](#)
  - [ejercicio13.c](#), [13](#)
- [BUFFER\\_SIZE](#)
  - [ejercicio13.c](#), [13](#)
  - [ejercicio9.c](#), [20](#)
- [cadena](#)
  - [\\_estructura](#), [5](#)
- [calcula\\_matriz](#)
  - [ejercicio13.c](#), [13](#)
- [calcula\\_primos](#)
  - [ejercicio12a.c](#), [8](#)
  - [ejercicio12b.c](#), [11](#)
- [dim](#)
  - [param](#), [6](#)
- [ESCRITURA](#)
  - [ejercicio9.c](#), [21](#)
- [ejercicio12a.c](#), [7](#)
  - [calcula\\_primos](#), [8](#)
  - [estructura](#), [8](#)
  - [main](#), [9](#)
  - [NUMERO\\_PROCESOS](#), [8](#)
- [ejercicio12b.c](#), [9](#)
  - [calcula\\_primos](#), [11](#)
  - [estructura](#), [10](#)
  - [main](#), [11](#)
  - [NUMERO\\_HILOS](#), [10](#)
- [ejercicio13.c](#), [12](#)
  - [ARRAY\\_SIZE](#), [13](#)
  - [BUFFER\\_SIZE](#), [13](#)
  - [calcula\\_matriz](#), [13](#)
  - [main](#), [14](#)
  - [Param](#), [13](#)
- [ejercicio4a.c](#), [14](#)
  - [NUM\\_PROC](#), [15](#)
- [ejercicio4b.c](#), [15](#)
  - [NUM\\_PROC](#), [15](#)
- [ejercicio5a.c](#), [16](#)
  - [NUM\\_PROC](#), [16](#)
- [ejercicio5b.c](#), [17](#)
  - [NUM\\_PROC](#), [17](#)
- [ejercicio6.c](#), [18](#)
  - [MAX\\_CAD](#), [18](#)
- [ejercicio8.c](#), [19](#)
  - [MAX\\_PATH](#), [19](#)
- [ejercicio9.c](#), [20](#)
  - [BUFFER\\_SIZE](#), [20](#)
  - [ESCRITURA](#), [21](#)
  - [factorial](#), [21](#)
  - [LECTURA](#), [21](#)
  - [MENSAJE\\_SIZE](#), [21](#)
  - [main](#), [21](#)
- [entero](#)
  - [\\_estructura](#), [5](#)
- [estructura](#)
  - [ejercicio12a.c](#), [8](#)
  - [ejercicio12b.c](#), [10](#)
- [factorial](#)
  - [ejercicio9.c](#), [21](#)
- [id](#)
  - [param](#), [6](#)
- [LECTURA](#)
  - [ejercicio9.c](#), [21](#)
- [MAX\\_CAD](#)
  - [ejercicio6.c](#), [18](#)
- [MAX\\_PATH](#)
  - [ejercicio8.c](#), [19](#)
- [MENSAJE\\_SIZE](#)
  - [ejercicio9.c](#), [21](#)
- [main](#)
  - [ejercicio12a.c](#), [9](#)
  - [ejercicio12b.c](#), [11](#)
  - [ejercicio13.c](#), [14](#)
  - [ejercicio9.c](#), [21](#)
- [matriz](#)
  - [param](#), [6](#)
- [mult](#)
  - [param](#), [6](#)
- [NUM\\_PROC](#)
  - [ejercicio4a.c](#), [15](#)
  - [ejercicio4b.c](#), [15](#)
  - [ejercicio5a.c](#), [16](#)
  - [ejercicio5b.c](#), [17](#)
- [NUMERO\\_HILOS](#)
  - [ejercicio12b.c](#), [10](#)
- [NUMERO\\_PROCESOS](#)
  - [ejercicio12a.c](#), [8](#)
- [Param](#)
  - [ejercicio13.c](#), [13](#)

param, [6](#)  
  dim, [6](#)  
  id, [6](#)  
  matriz, [6](#)  
  mult, [6](#)