

Escalonamento (FJSSP)

Generated by Doxygen 1.9.3

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 CelulaPlano Struct Reference	5
3.1.1 Detailed Description	5
3.2 list_conection Struct Reference	5
3.2.1 Detailed Description	6
3.3 list_job Struct Reference	6
3.3.1 Detailed Description	6
3.4 list_machine Struct Reference	6
3.4.1 Detailed Description	6
4 File Documentation	7
4.1 functions.c File Reference	7
4.1.1 Detailed Description	9
4.1.2 Function Documentation	9
4.1.2.1 avg_time()	9
4.1.2.2 change_operation()	10
4.1.2.3 count_op_ids()	10
4.1.2.4 exist_connection()	10
4.1.2.5 exist_machine()	11
4.1.2.6 exist_operation()	11
4.1.2.7 free_connections()	12
4.1.2.8 free_jobs()	12
4.1.2.9 free_machines()	12
4.1.2.10 get_new_job_id()	13
4.1.2.11 get_new_operation_id()	13
4.1.2.12 head_insert_connection()	14
4.1.2.13 head_insert_job()	14
4.1.2.14 head_insert_machine()	14
4.1.2.15 insert_operation()	15
4.1.2.16 interface_principal()	15
4.1.2.17 max_time()	16
4.1.2.18 menu_principal()	16
4.1.2.19 min_time()	17
4.1.2.20 read_connections()	17
4.1.2.21 read_jobs()	17
4.1.2.22 read_machines()	18
4.1.2.23 remove_job()	18

4.1.2.24 remove_operation()	18
4.1.2.25 show_connections()	19
4.1.2.26 show_jobs()	19
4.1.2.27 show_machines()	19
4.1.2.28 show_machines_by_opjob_id()	20
4.1.2.29 show_operations_by_job()	20
4.1.2.30 write_connections()	20
4.1.2.31 write_jobs()	21
4.1.2.32 write_machines()	21
4.2 functions.h	21
4.3 main.c File Reference	22
4.3.1 Detailed Description	23
4.3.2 Function Documentation	23
4.3.2.1 main()	23
4.4 structs.h File Reference	23
4.4.1 Detailed Description	24
4.5 structs.h	25
Index	27

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

CelulaPlano	Estrutura que representa uma celula da tabela de planeamento FJSSP	5
list_conection	Estrutura que contem a conexão das listas jobs,operacoes e maquinas	5
list_job	Estrutura que representa a lista dos jobs	6
list_machine	Estrutura que representa a lista das máquinas	6

Chapter 2

File Index

2.1 File List

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

functions.c		
	Funções do programa	7
functions.h	??
main.c		
	Solução para problema de escalonamento (Flexifile job shop problem)	22
structs.h		
	Estruturas do programa	23

Chapter 3

Data Structure Documentation

3.1 CelulaPlano Struct Reference

Estrutura que representa uma celula da tabela de planeamento FJSSP.

```
#include <structs.h>
```

Data Fields

- int **id_job**
- int **id_op**
- int **id_mac**
- int **t**

3.1.1 Detailed Description

Estrutura que representa uma celula da tabela de planeamento FJSSP.

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

- [structs.h](#)

3.2 list_conection Struct Reference

Estrutura que contem a conexão das listas jobs,operacoes e maquinas.

```
#include <structs.h>
```

Data Fields

- int **id_job**
- int **id_op**
- int **id_mac**
- int **time**
- struct [list_conection](#) * **next**
- struct [list_conection](#) * **previous**

3.2.1 Detailed Description

Estrutura que contem a conexão das listas jobs, operacoes e maquinas.

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

- [structs.h](#)

3.3 list_job Struct Reference

Estrutura que representa a lista dos jobs.

```
#include <structs.h>
```

Data Fields

- int **id_job**
- struct [list_job](#) * **next**
- struct [list_job](#) * **previous**

3.3.1 Detailed Description

Estrutura que representa a lista dos jobs.

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

- [structs.h](#)

3.4 list_machine Struct Reference

Estrutura que representa a lista das máquinas.

```
#include <structs.h>
```

Data Fields

- int **id_mac**
- struct [list_machine](#) * **next**
- struct [list_machine](#) * **previous**

3.4.1 Detailed Description

Estrutura que representa a lista das máquinas.

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

- [structs.h](#)

Chapter 4

File Documentation

4.1 functions.c File Reference

Funções do programa.

```
#include <stdio.h>
#include <stdlib.h>
#include "functions.h"
```

Functions

- [Machine](#) * [read_machines](#) ([Machine](#) *list_machines)
Leitura do ficheiro machines.txt e armazenamento em memória.
- [Machine](#) * [write_machines](#) ([Machine](#) *list_machines)
Escrita do ficheiro machines.txt e armazenamento em memória.
- [Job](#) * [read_jobs](#) ([Job](#) *list_jobs)
Leitura do ficheiro jobs.txt e armazenamento em memória.
- [Job](#) * [write_jobs](#) ([Job](#) *list_jobs)
Escrita do ficheiro opjobs.txt e armazenamento em memória.
- [Connection](#) * [read_connections](#) ([Connection](#) *list_connections)
Leitura do ficheiro connections.txt e armazenamento em memória.
- [Connection](#) * [write_connections](#) ([Connection](#) *list_connections)
Escrita do ficheiro connections.txt e armazenamento em memória.
- int [free_machines](#) ([Machine](#) *list_machines)
Desalocar memória ocupada pela lista das máquinas.
- int [free_jobs](#) ([Job](#) *list_job)
Desalocar memória ocupada pela lista de jobs.
- int [free_connections](#) ([Connection](#) *list_connections)
Desalocar memória ocupada pela lista de conexões.
- [Machine](#) * [head_insert_machine](#) ([Machine](#) *list_machines, [Machine](#) *aux)
Inserção á cabeça na lista das máquinas.
- [Job](#) * [head_insert_job](#) ([Job](#) *list_jobs, [Job](#) *aux)
Inserção á cabeça na lista dos jobs.
- [Connection](#) * [head_insert_connection](#) ([Connection](#) *list_connections, [Connection](#) *aux)

- Inserção á cabeça na lista das conexões.*

 - int **exist_machine** (**Machine** *list_machines, int id_machine)
 - Verifica se um id de uma maquina inserido existe na lista das maquinas.*
 - int **exist_operation** (**Connection** *list_connections, int id_operation, int id_job)
 - Verifica se um id de uma operação de um job inserido existe na lista das conexões.*
 - int **exist_connection** (**Connection** *list_connections, int id_machine, int id_operation, int id_job)
 - Verifica se uma combinação de id_mac, id_op e id_job existe na lista das conexões.*
 - int **get_new_operation_id** (**Connection** *list_connections, int id_job)
 - Função para encontrar o próximo id da lista das conexões a ser usado por uma nova operação a ser inserida na respetiva função.*
 - int **get_new_job_id** (**Job** *list_jobs)
 - Função para encontrar o próximo id da lista jobs a ser usado por um novo job a ser inserida na respetiva função.*
 - void **show_machines** (**Machine** *list_machines)
 - Listagem das máquinas na lista.*
 - void **show_machines_by_opjob_id** (**Connection** *list_connections, int id_operation, int id_job)
 - Mostra maquinas de uma operação de um job, pelo seu id de operação e id do job.*
 - void **show_jobs** (**Job** *list_jobs)
 - Listagem dos jobs.*
 - void **show_operations_by_job** (**Connection** *list_connections, int id_job)
 - Mostra os ids das operacoes pertencentes a um job passado por id.*
 - void **show_connections** (**Connection** *list_connections)
 - Mostra todas as conexões da lista.*
 - int **count_op_ids** (**Connection** *list_connections, int id_operation, int id_job)
 - Conta o numero de vezes que um id de operacao aparece num job.*
 - **Connection** * **insert_operation** (**Connection** *list_connections, **Machine** *list_machines, int id_job)
 - Inserção de uma operação na lista das operações e lista MacOp.*
 - **Connection** * **remove_operation** (**Connection** *list_connections, int id_operation, int id_job)
 - Remoção de uma operação da lista operações e respetivamente da lista intermedia MacOp.*
 - **Connection** * **change_operation** (**Connection** *list_connections, int id_operation, int id_job, int id_machine, int new_time)
 - Alteração de uma operação da lista operações Altera os tempos de máquinas que seleccione.*
 - float **avg_time** (**Connection** *list_connections, int id_operation, int id_job)
 - Determinação da quantidade média de unidades de tempo necessárias para completar uma operação de um job, considerando todas as alternativas possíveis.*
 - int **min_time** (**Connection** *list_connections, int id_job)
 - Determinação da quantidade mínima de unidades de tempo necessárias para completar o job.*
 - int **max_time** (**Connection** *list_connections, int id_job)
 - Determinação da quantidade máxima de unidades de tempo necessárias para completar o job.*
 - **Job** * **remove_job** (**Job** *list_jobs, **Connection** **list_connections, int id_job)
 - Remoção de um job Sucessidamente ira remover das restantes listas a ligações.*
 - void **interface_principal** ()
 - texto apresentado ao utilizador no Menu principal*
 - void **menu_principal** (**Job** **list_jobs, **Machine** **list_machines, **Connection** **list_connections)
 - Menu de opcoes.*

4.1.1 Detailed Description

Funções do programa.

Author

João Apresentação (a21152@alunos.ipca.pt)

Version

0.1

Date

2022-03-26

Copyright

Copyright (c) 2022

4.1.2 Function Documentation

4.1.2.1 avg_time()

```
float avg_time (
    Connection * list_connections,
    int id_operation,
    int id_job )
```

Determinação da quantidade média de unidades de tempo necessárias para completar uma operação de um job, considerando todas as alternativas possíveis.

Parameters

<i>list_connections</i>	lista das conexões
<i>id_operation</i>	id da operação selecionada
<i>id_job</i>	id do job selecionado

Returns

float

4.1.2.2 change_operation()

```
Connection * change_operation (
    Connection * list_connections,
    int id_operation,
    int id_job,
    int id_machine,
    int new_time )
```

Alteração de uma operação da lista operações Altera os tempos de máquinas que seleccione.

Parameters

<i>list_operations</i>	lista das operações
<i>list_macops</i>	lista intermedia entre as operações e as máquinas
<i>id_operation</i>	id da operação a ser alterada

4.1.2.3 count_op_ids()

```
int count_op_ids (
    Connection * list_connections,
    int id_operation,
    int id_job )
```

Conta o numero de vezes que um id de operacao aparece num job.

Parameters

<i>list_connections</i>	lista de conexoes
<i>id_operation</i>	id da operacao
<i>id_job</i>	id do job

Returns

int

4.1.2.4 exist_connection()

```
int exist_connection (
    Connection * list_connections,
    int id_machine,
    int id_operation,
    int id_job )
```

Verifica se uma combinação de id_mac, id_op e id_job existe na lista das conexões.

Parameters

<i>list_connections</i>	lista de conexões
<i>id_machine</i>	id da máquina
<i>id_operation</i>	id da operação
<i>id_job</i>	id do job

Returns

int

4.1.2.5 exist_machine()

```
int exist_machine (
    Machine * list_machines,
    int id_machine )
```

Verifica se um id de uma maquina inserido existe na lista das maquinas.

Parameters

<i>list_machines</i>	lista das máquinas
<i>id_machine</i>	id de uma máquina inserido

Returns

int

4.1.2.6 exist_operation()

```
int exist_operation (
    Connection * list_connections,
    int id_operation,
    int id_job )
```

Verifica se um id de uma operação de um job inserido existe na lista das conexões.

Parameters

<i>list_connections</i>	lista das conexões
<i>id_operation</i>	id da operação selecionado
<i>id_job</i>	id do job

Returns

int

4.1.2.7 free_connections()

```
int free_connections (
    Connection * list_connections )
```

Desalocar memória ocupada pela lista de conexões.

Parameters

<i>list_connections</i>	lista de conexões
-------------------------	-------------------

Returns

int

4.1.2.8 free_jobs()

```
int free_jobs (
    Job * list_job )
```

Desalocar memória ocupada pela lista de jobs.

Parameters

<i>list_job</i>	lista de jobs
-----------------	---------------

Returns

int

4.1.2.9 free_machines()

```
int free_machines (
    Machine * list_machines )
```

Desalocar memória ocupada pela lista das máquinas.

Parameters

<i>list_machines</i>	lista das máquinas
----------------------	--------------------

Returns

int

4.1.2.10 get_new_job_id()

```
int get_new_job_id (  
    Job * list_jobs )
```

Função para encontrar o próximo id da lista jobs a ser usado por um novo job a ser inserida na respetiva função.

Parameters

<i>list_jobs</i>	lista dos jobs
------------------	----------------

Returns

int

4.1.2.11 get_new_operation_id()

```
int get_new_operation_id (  
    Connection * list_connections,  
    int id_job )
```

Função para encontrar o próximo id da lista das conexões a ser usado por uma nova operação a ser inserida na respetiva função.

Parameters

<i>list_connections</i>	lista das conexões
<i>id_job</i>	id do job

Returns

int

4.1.2.12 head_insert_connection()

```
Connection * head_insert_connection (
    Connection * list_connections,
    Connection * aux )
```

Inserção á cabeça na lista das conexões.

Parameters

<i>list_connections</i>	Lista das conexões
<i>aux</i>	Lista auxiliar

Returns

Connection*

4.1.2.13 head_insert_job()

```
Job * head_insert_job (
    Job * list_jobs,
    Job * aux )
```

Inserção á cabeça na lista dos jobs.

Parameters

<i>list_jobs</i>	Lista dos jobs
<i>aux</i>	Lista auxiliar

Returns

Job*

4.1.2.14 head_insert_machine()

```
Machine * head_insert_machine (
    Machine * list_machines,
    Machine * aux )
```

Inserção á cabeça na lista das máquinas.

Parameters

<i>list_machines</i>	Lista das máquinas
<i>aux</i>	Lista auxiliar

Returns

Machine*

4.1.2.15 insert_operation()

```

Connection * insert_operation (
    Connection * list_connections,
    Machine * list_machines,
    int id_job )

```

Inserção de uma operação na lista das operações e lista MacOp.

Parameters

<i>list_operations</i>	lista das operações
<i>list_macops</i>	lista intermedia entre as operações e as máquinas
<i>list_machines</i>	lista das máquinas

4.1.2.16 interface_principal()

```
void interface_principal ( )
```

texto apresentado ao utilizador no Menu principal

int simulation(Connection *list_connections) { percorrer as listas verificar se maquinas são iguais se não forem -> ambas iniciam no instante 0 se forem -> verificar se tem tempos diferentes se m1 tiver mais tempo que m2 -> inicia no instante 0 job/op da m2 se forem iguais -> ??? enviar dados para a celula plano enviar tudo para html tabelar

```

int id_mac = 0, time;
CelulaPlano simulation[][];
Connection *head_connection, *head_connection2, *aux;

while (mac != get_last_mac())
{
    head_connection = *list_connections;
    while (head_connection)
    {
        if (head_connection->id_mac == id_mac)
        {
            head_connection2 = *list_connections;
            time = head_connection->time;
            while (head_connection2)
            {
                if (head_connection->id_op == head_connection2->id_op && head_connection2->time < time)
                {
                    aux = head_connection2;
                }

                head_connection2 = head_connection2->next;
            }
            simulation[aux->id_mac][t];
        }
    }
}

```

```

        }

        head_connection = head_connection->next;
    }

    id_mac++;
}

}

int Ocupa(int job, int oper, int maq, int t) { }
```

4.1.2.17 max_time()

```

int max_time (
    Connection * list_connections,
    int id_job )
```

Determinação da quantidade máxima de unidades de tempo necessárias para completar o job.

Parameters

<i>list_connections</i>	lista das conexões
<i>id_job</i>	id do job selecionado

Returns

int

4.1.2.18 menu_principal()

```

void menu_principal (
    Job ** list_jobs,
    Machine ** list_machines,
    Connection ** list_connections )
```

Menu de opcoes.

Parameters

<i>list_jobs</i>	lista de jobs
<i>list_operations</i>	lista de operacoes
<i>list_machines</i>	lista de maquinas
<i>list_connections</i>	lista de conexões

4.1.2.19 min_time()

```
int min_time (
    Connection * list_connections,
    int id_job )
```

Determinação da quantidade minima de unidades de tempo necessárias para completar o job.

Parameters

<i>list_connections</i>	lista das conexões
<i>id_job</i>	id do job selecionado

Returns

int

4.1.2.20 read_connections()

```
Connection * read_connections (
    Connection * list_connections )
```

Leitura do ficheiro connections.txt e armazenamento em memória.

Parameters

<i>list_connections</i>	Lista intermedia que contem jobs
-------------------------	----------------------------------

Returns

Connection*

4.1.2.21 read_jobs()

```
Job * read_jobs (
    Job * list_jobs )
```

Leitura do ficheiro jobs.txt e armazenamento em memória.

Parameters

<i>list_jobs</i>	Lista intermedia que contem jobs
------------------	----------------------------------

Returns

Job*

4.1.2.22 read_machines()

```
Machine * read_machines (
    Machine * list_machines )
```

Leitura do ficheiro machines.txt e armazenamento em memória.

Parameters

<i>list_machines</i>	Lista que contem todas as máquinas
----------------------	------------------------------------

Returns

Machine*

4.1.2.23 remove_job()

```
Job * remove_job (
    Job * list_jobs,
    Connection ** list_connections,
    int id_job )
```

Remoção de um job Sucessidamente ira remover das restantes listas a ligações.

Parameters

<i>list_jobs</i>	Lista de jobs
<i>list_opjobs</i>	Lista intermedia de jobs e operações
<i>list_operations</i>	Lista de operações
<i>list_macops</i>	Lista intermedia de máquinas e operações
<i>id_job</i>	Id do job a ser removido

Returns

Job*

4.1.2.24 remove_operation()

```
Connection * remove_operation (
    Connection * list_connections,
```

```
int id_operation,  
int id_job )
```

Remoção de uma operação da lista operações e respetivamente da lista intermedia MacOp.

Parameters

<i>list_operations</i>	lista das operações
<i>list_macops</i>	lista intermedia entre as operações e as máquinas
<i>id_operation</i>	id da operação a ser removida

4.1.2.25 show_connections()

```
void show_connections (  
    Connection * list_connections )
```

Mostra todas as conexões da lista.

Parameters

<i>list_connections</i>	Lista de conexões
-------------------------	-------------------

4.1.2.26 show_jobs()

```
void show_jobs (  
    Job * list_jobs )
```

Listagem dos jobs.

Parameters

<i>list_jobs</i>	Lista dos jobs
------------------	----------------

4.1.2.27 show_machines()

```
void show_machines (  
    Machine * list_machines )
```

Listagem das máquinas na lista.

Parameters

<i>list_machines</i>	Lista das máquinas
----------------------	--------------------

4.1.2.28 show_machines_by_opjob_id()

```
void show_machines_by_opjob_id (  
    Connection * list_connections,  
    int id_operation,  
    int id_job )
```

Mostra maquinas de uma operação de um job, pelo seu id de operação e id do job.

Parameters

<i>list_connections</i>	lista de conexões
<i>id_operation</i>	id da operação
<i>id_job</i>	id do job

4.1.2.29 show_operations_by_job()

```
void show_operations_by_job (  
    Connection * list_connections,  
    int id_job )
```

Mostra os ids das operacoes pertencentes a um job passado por id.

Parameters

<i>list_connections</i>	lista de conexões
<i>id_job</i>	id do job

4.1.2.30 write_connections()

```
Connection * write_connections (  
    Connection * list_connections )
```

Escrita do ficheiro connections.txt e armazenamento em memória.

Parameters

<i>list_connections</i>	
-------------------------	--

Returns

Connection*

4.1.2.31 write_jobs()

```
Job * write_jobs (
    Job * list_jobs )
```

Escrita do ficheiro opjobs.txt e armazenamento em memória.

Parameters

<i>list_opjobs</i>	
--------------------	--

Returns

OpJob*

4.1.2.32 write_machines()

```
Machine * write_machines (
    Machine * list_machines )
```

Escrita do ficheiro machines.txt e armazenamento em memória.

Parameters

<i>list_machines</i>	lista das maquinas
----------------------	--------------------

Returns

Machine*

4.2 functions.h

```
1
11 #ifndef FUNCTIONS
12 #define FUNCTIONS
13 #include "structs.h"
14
15 #pragma region MACHINES
16
17 Machine *read_machines(Machine *listMachine);
18 Machine *write_machines(Machine *listMachine);
19 int free_machines(Machine *list);
20 Machine *head_insert_machine(Machine *list, Machine *aux);
21 void show_machines(Machine *list);
22 int exist_machine(Machine *list_machines, int id_machine);
23
```

```

24 #pragma endregion
25
26 #pragma region OPERATIONS
27
28 Connection *remove_operation(Connection *list_connections, int id_operation, int id_job);
29 Connection *change_operation(Connection *list_connections, int id_operation, int id_job, int id_machine,
    int new_time);
30 Connection *insert_operation(Connection *list_connections, Machine *list_machines, int id_job);
31
32 #pragma endregion
33
34 #pragma region JOBS
35
36 int free_jobs(Job *list_job);
37 Job *read_jobs(Job *list_jobs);
38 Job *write_jobs(Job *list_jobs);
39 Job *head_insert_job(Job *list_jobs, Job *aux);
40 void show_jobs(Job *list_jobs);
41 Job *remove_job(Job *list_jobs, Connection **list_connections, int id_job);
42
43 #pragma endregion
44
45 #pragma region CONNECTIONS
46
47 Connection *read_connections(Connection *list_connections);
48 Connection *write_connections(Connection *list_connections);
49 int free_connections(Connection *list_connections);
50 Connection *head_insert_connection(Connection *list_connections, Connection *aux);
51 void show_connections(Connection *list_connections);
52 void show_operations_by_job(Connection *list_connections, int id_job);
53 void show_machines_by_opjob_id(Connection *list_connections, int id_operation, int id_job);
54 int exist_connection(Connection *list_connections, int id_machine, int id_operation, int id_job);
55 int exist_operation(Connection *list_connections, int id_operation, int id_job);
56
57 #pragma endregion
58
59 #pragma region INTERFACE
60
61 void interface_principal();
62 void menu_principal(Job **list_jobs, Machine **list_machines, Connection **list_connections);
63
64 #pragma endregion
65
66 #pragma region CALCULATIONS
67
68 int min_time(Connection *list_connections, int id_job);
69 int max_time(Connection *list_connections, int id_job);
70 float avg_time(Connection *list_connections, int id_operation, int id_job);
71 int count_op_ids(Connection *list_connections, int id_operation, int id_job);
72
73 #pragma endregion
74
75 #pragma region GET
76
77 int get_new_operation_id(Connection *list_connections, int id_job);
78 int get_new_job_id(Job *list_jobs);
79
80 #pragma endregion
81
82 #pragma region SIMULATION
83
84 int simulation(Connection *list_connections);
85 int Ocupa(int job, int oper, int mag, int t);
86
87 #pragma endregion
88
89 #endif

```

4.3 main.c File Reference

Solução para problema de escalonamento (Flexifile job shop problem)

```

#include <stdio.h>
#include <stdlib.h>
#include "functions.h"

```

Functions

- `int main ()`
Função main.

4.3.1 Detailed Description

Solução para problema de escalonamento (Flexifile job shop problem)

Author

João Apresentação (a21152@alunos.ipca.pt)

Version

0.1

Date

2022-03-26

Copyright

Copyright (c) 2022

4.3.2 Function Documentation

4.3.2.1 main()

```
int main ( )
```

Função main.

Returns

int

4.4 structs.h File Reference

Estruturas do programa.

Data Structures

- struct [list_machine](#)
Estrutura que representa a lista das máquinas.
- struct [list_job](#)
Estrutura que representa a lista dos jobs.
- struct [list_conection](#)
Estrutura que contem a conexão das listas jobs,operacoes e maquinas.
- struct [CelulaPlano](#)
Estrutura que representa uma celula da tabela de planeamento FJSSP.

Typedefs

- typedef struct [list_machine](#) **Machine**
Estrutura que representa a lista das máquinas.
- typedef struct [list_job](#) **Job**
Estrutura que representa a lista dos jobs.
- typedef struct [list_conection](#) **Connection**
Estrutura que contem a conexão das listas jobs,operacoes e maquinas.

4.4.1 Detailed Description

Estruturas do programa.

Author

João Apresentação (a21152@alunos.ipca.pt)

Version

0.1

Date

2022-03-26

Copyright

Copyright (c) 2022

4.5 structs.h

[Go to the documentation of this file.](#)

```
1
11 #ifndef STRUCTS
12 #define STRUCTS
13
18 typedef struct list_machine
19 {
20     int id_mac;
21     struct list_machine *next, *previous;
22 } Machine;
23
28 typedef struct list_job
29 {
30     int id_job;
31     struct list_job *next, *previous;
32 } Job;
33
38 typedef struct list_conection
39 {
40     int id_job, id_op, id_mac, time;
41     struct list_conection *next, *previous;
42 } Connection;
43
48 typedef struct
49 {
50     int id_job, id_op, id_mac, t;
51 } CelulaPlano;
52
53 #endif
```


Index

avg_time
 functions.c, 9

CelulaPlano, 5

change_operation
 functions.c, 9

count_op_ids
 functions.c, 10

exist_connection
 functions.c, 10

exist_machine
 functions.c, 11

exist_operation
 functions.c, 11

free_connections
 functions.c, 12

free_jobs
 functions.c, 12

free_machines
 functions.c, 12

functions.c, 7
 avg_time, 9
 change_operation, 9
 count_op_ids, 10
 exist_connection, 10
 exist_machine, 11
 exist_operation, 11
 free_connections, 12
 free_jobs, 12
 free_machines, 12
 get_new_job_id, 13
 get_new_operation_id, 13
 head_insert_connection, 13
 head_insert_job, 14
 head_insert_machine, 14
 insert_operation, 15
 interface_principal, 15
 max_time, 16
 menu_principal, 16
 min_time, 16
 read_connections, 17
 read_jobs, 17
 read_machines, 18
 remove_job, 18
 remove_operation, 18
 show_connections, 19
 show_jobs, 19
 show_machines, 19
 show_machines_by_opjob_id, 20
 show_operations_by_job, 20
 write_connections, 20
 write_jobs, 21
 write_machines, 21

get_new_job_id
 functions.c, 13

get_new_operation_id
 functions.c, 13

head_insert_connection
 functions.c, 13

head_insert_job
 functions.c, 14

head_insert_machine
 functions.c, 14

insert_operation
 functions.c, 15

interface_principal
 functions.c, 15

list_conection, 5

list_job, 6

list_machine, 6

main
 main.c, 23

main.c, 22
 main, 23

max_time
 functions.c, 16

menu_principal
 functions.c, 16

min_time
 functions.c, 16

read_connections
 functions.c, 17

read_jobs
 functions.c, 17

read_machines
 functions.c, 18

remove_job
 functions.c, 18

remove_operation
 functions.c, 18

show_connections
 functions.c, 19

show_jobs
 functions.c, [19](#)
show_machines
 functions.c, [19](#)
show_machines_by_opjob_id
 functions.c, [20](#)
show_operations_by_job
 functions.c, [20](#)
structs.h, [23](#)

write_connections
 functions.c, [20](#)
write_jobs
 functions.c, [21](#)
write_machines
 functions.c, [21](#)