My Project

Generated by Doxygen 1.7.6.1

Sun Nov 17 2013 19:01:47

Contents

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

File Index

2.1 File List

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

Chapter 3

Class Documentation

3.1 course Struct Reference

Public Attributes

- float **freq**
- char **name** [20]
- int **number_of_sems_floated**
- char **deps** [20]
- char **unis** [100]
- dept * **dep**
- vector< student * > **course_stud**
- bool **floated**
- faculty * **instructor**

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

- global1.h

3.2 dept Struct Reference

Public Attributes

- char **name** [10]
- float **numfac**
- float **studperyr**
- float **semdeptcourse**
- float **semnondeptcourses**
- vector< course * > **depcourses**
- vector< course * > **nondepcourses**

- vector< faculty * > **fac**
- vector< student * > **dep_stud**
- map< interest *, float > **interests**

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

- global1.h

3.3 facnode Struct Reference

This structure stores data of Faculty.

`#include <analyser.h>`

Public Attributes

- string name
- string surname
- string entryno
- string hostel
- string dept
- string univ
- int **roomNo**
- vector< string > interests
- vector< string > courses

3.3.1 Detailed Description

This structure stores data of Faculty.

3.3.2 Member Data Documentation

3.3.2.1 facnode::courses

Member 'courses' contains courses of the Faculty currently he/she is taking

3.3.2.2 facnode::dept

Member 'dept' contains Department of the Faculty

3.3.2.3 facnode::entryno

Member 'entryno' contains entry no(id) of the Faculty

3.3.2.4 facnode::hostel

Member 'hostel' contains house assigned to the Faculty.

3.3.2.5 facnode::interests

Member 'interests' contains the interests of the Faculty

3.3.2.6 facnode::name

Member 'name' contains name of Faculty

3.3.2.7 facnode::surname

Member 'surname' contains surname of Faculty.

3.3.2.8 facnode::univ

Member 'univ' contains University of of the Faculty

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

- analyser.h

3.4 faculty Struct Reference

Public Attributes

- int **employee_code**
- char * **name**
- char * **surname**
- dept * **department**
- UNIV * **uni**
- houses * **house**
- int **plotno**
- vector< interest * > **intrst**
- map< string, float > **set_of_interests**
- vector< course * > **courses_taken**

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

- global1.h

3.5 graph Class Reference

A graph class.

`#include <global1.h>`

Public Member Functions

- graph (int i)

  A constructor.
- ∼graph ()

  A constructor.
- void create_edge (student ∗v1, student ∗v2)

  A normal member taking two arguments and returning a void value.
- void del_edge (student ∗v1, student ∗v2)

  A normal member taking two arguments and returning a void value.
- int no_of_friends (student ∗v1)

  A normal member taking one argument and returning an int value.
- int ∗ list_of_friends (student ∗v2)

  A normal member taking one argument and returning an int array.
- bool already_friends (student ∗v1, student ∗v2)

  A normal member taking two arguments and returning a bool value.

Public Attributes

- int numvertices

  number of vertices variable.
- int numedges

  number of edges variable.
- bool ∗∗ **graphmatrix**

3.5.1 Detailed Description

A graph class.

class description.

3.5.2 Member Function Documentation

3.5.2.1 bool graph::already_friends (student ∗ *v1,* student ∗ *v2*)

A normal member taking two arguments and returning a bool value.

Parameters

v1	a student.
v2	a student. Checks whether the two people are already friends or not.

Returns

void

3.5.2.2 void graph::create_edge (student * *v1*, student * *v2*)

A normal member taking two arguments and returning a void value.

Parameters

v1	first student.
v2	second student. Adds student 1 to friendlist of 2 and vice-versa

Returns

void

3.5.2.3 void graph::del_edge (student * *v1*, student * *v2*)

A normal member taking two arguments and returning a void value.

Parameters

v1	first student.
v2	second student. Removes student v1 to friendlist of v2 and vice-versa

Returns

void

3.5.2.4 int* graph::list_of_friends (student * *v2*)

A normal member taking one argument and returning an int array.

Parameters

v1	a student. Gives all the friends of student v1

Returns

void

3.5.2.5 int graph::no_of_friends (student * *v1*)

A normal member taking one argument and returning an int value.

Parameters

v1	a student. Gives no. of friends of student v1

Returns

void

3.5.3 Member Data Documentation

3.5.3.1 int graph::numedges

number of edges variable.

gives estimate of no. of friends

3.5.3.2 int graph::numvertices

number of vertices variable.

would be same as no. of students

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

- global1.h
- graph.cpp
- main.cpp

3.6 hostels Struct Reference

This structure stores all the hostels.

`#include <global1.h>`

Public Attributes

- char name [30]
- int we [4]
- vector< student * > hostel_stud

3.6.1 Detailed Description

This structure stores all the hostels.

3.6.2 Member Data Documentation

3.6.2.1 hostels::hostel_stud

Member 'hostel_stud' contains list of students living in that hostel.

3.6.2.2 hostels::name

Member 'name' contains name of hostel.

3.6.2.3 hostels::we

Member 'we' denotes floor.

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

- global1.h

3.7 houses Struct Reference

This structure stores all the faculty houses.

`#include <global1.h>`

Public Attributes

- char name [30]
- int fill
- vector< faculty * > house_fac

3.7.1 Detailed Description

This structure stores all the faculty houses.

3.7.2 Member Data Documentation

3.7.2.1 houses::fill

Member 'fill' denotes no. of faculty in this house locality.

3.7.2.2 houses::house_fac

Member 'house_fac' contains list of faculty with that house locality.

3.7.2.3 houses::name

Member 'name' contains name of house

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

- global1.h

3.8 interest Struct Reference

This structure stores all the intersets.

`#include <global1.h>`

Public Attributes

- char name [90]
- vector< student * > interest_stud
- vector< faculty * > interest_fac

3.8.1 Detailed Description

This structure stores all the intersets.

3.8.2 Member Data Documentation

3.8.2.1 interest::interest_fac

Member 'interest_fac' contains list of faculty with that interest.

3.8.2.2 interest::interest_stud

Member 'interest_stud' contains list of students with that interest.

3.8.2.3 interest::name

Member 'name' contains name of interest

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

- global1.h

3.9 mymsgbuf Struct Reference

This structure stores messgae struct for message queue.

`#include <message_temp.h>`

Public Attributes

- long **mtype**
- int **thdid**
- int **waketime**

3.9.1 Detailed Description

This structure stores messgae struct for message queue.

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

- message_temp.h

3.10 pq_operand Class Reference

`#include <TimeKeeper.h>`

Public Member Functions

- bool operator() (pair< int, int > &n1, pair< int, int > &n2)

  Comparater taking two pair of ints and returning a bool value.

3.10.1 Detailed Description

A comparator class for prioirty queue to schedule tasks

3.10.2 Member Function Documentation

3.10.2.1 bool pq_operand::operator() (pair< int, int > & *n1*, pair< int, int > & *n2*) [inline]

Comparater taking two pair of ints and returning a bool value.

Parameters

n1	: A pointer to pair of integers
n2,:	A pointer to pair of integers

Returns

Bool

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

- TimeKeeper.h

3.11 student Struct Reference

Public Attributes

- int **Entry_No**
- int **Year**
- int **Batch**
- char * **name**
- char * **surname**
- dept * **dep**
- UNIV * **uni**
- hostels * **host**
- string **floor**
- int **room_no**
- unsigned long **location_in_allstudents**
- vector< interest * > **intrst**
- vector< course * > **sem_dep_courses_taken**
- vector< course * > **sem_non_dep_courses_taken**
- vector< student * > **friends**

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

- global1.h

3.12 studnode Struct Reference

This structure stores data of students.

`#include <analyser.h>`

Public Attributes

- string name
- string surname
- string hostel
- string dept
- string univ

- int batch
- int year
- int roomNo
- int entryno
- char floor
- vector< string > **interests**
- vector< string > courses
- vector< int > friends

3.12.1 Detailed Description

This structure stores data of students.

3.12.2 Member Data Documentation

3.12.2.1 studnode::batch

Member 'batch' contains batch of the student

3.12.2.2 studnode::courses

Member 'courses' contains courses of the student currently he/she is pursuing

3.12.2.3 studnode::dept

Member 'dept' contains Department of the student

3.12.2.4 studnode::entryno

Member 'entryno' contains entry no of the student

3.12.2.5 studnode::floor

Member 'floor' contains entry no of the student

3.12.2.6 studnode::friends

Member 'firends' contains friend list of the student

3.12.2.7 studnode::hostel

Member 'hostel' contains hostel assigned to the student.

3.12.2.8 studnode::name

Member 'name' contains name of Student

3.12.2.9 studnode::roomNo

Member 'roomNo' contains room no. in the hostel

3.12.2.10 studnode::surname

Member 'surname' contains surname of student.

3.12.2.11 studnode::univ

Member 'unic' contains University of of the student

3.12.2.12 studnode::year

Member 'year' contains the year of degree the student is pursuing

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

- analyser.h

3.13 UNIV Struct Reference

Public Attributes

- char **name** [100]
- vector< string > **hostel**
- map< string, hostels > **hostel_struct**
- vector< string > **houselocality**
- map< string, houses > **house_struct**
- float **friendshiprate**
- float **open**
- float **friendliness**
- map< string, dept > **depts**
- vector< course * > **allcourses**
- map< string, float > **interest1**

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

- global1.h

Chapter 4

File Documentation

4.1 analyser.h File Reference

A Documented file. Details.

```
#include <string>  #include <limits>  #include <vector> ×
#include <iostream> #include <fstream> #include <stdio.-
h> #include <stdlib.h> #include <string.h> #include <map> ×
```

Classes

- struct studnode

  This structure stores data of students.
- struct facnode

  This structure stores data of Faculty.

Functions

- int **shortpath** (int id1, int id2)
- int **floyd** ()
- void **shortest_path** (int i, int j)
- void **list_short_path** (int i, int j)
- int **importance** (int i)
- void **moreimp** (int i)
- void **largestshortestpath** ()
- bool **checkfriends** (int i, int j)
- void **clique** (int i)

4.1.1 Detailed Description

A Documented file. Details.

4.2 genCourse.h File Reference

Generates courses and allocates students to these courses.

```
#include "main_2.h"
```

Functions

- void floatCourses (int course_random)

  A function taking one arguments and returning void. It floats all courses in a given semester.
- void allocateCourses (int course_random)

  A function taking one arguments and returning void. It allocates floated all courses in a given semester.

4.2.1 Detailed Description

Generates courses and allocates students to these courses.

4.2.2 Function Documentation

4.2.2.1 void allocateCourses (int *course_random*)

A function taking one arguments and returning void. It allocates floated all courses in a given semester.

Parameters

course_random	Random Seed Integer

Returns

Void

4.2.2.2 void floatCourses (int *course_random*)

A function taking one arguments and returning void. It floats all courses in a given semester.

Parameters

course_random	Random Seed Integer

Returns

Void

4.3 generator.h File Reference

A Documented file. Details.

```
#include <pthread.h> #include <unistd.h> #include <iostream> ×
#include <sys/types.h> #include <errno.h> #include <stdio.-
h> #include <sys/wait.h> #include <sys/stat.h> #include
<fcntl.h> #include <string.h> #include <typeinfo> #include
<utility> #include <sys/mman.h> #include "graphml.h" ×
#include <queue> #include <vector> #include "genFriend.-
h" #include "main_2.h" #include "message_temp.h" #include
"genstudent.h" #include "genCourse.h" #include <stdlib.-
h> #include <map> #include "genfac.h" #include "Semaphore.-
h"
```

Functions

- void ∗ generateFaculty (void ∗args)

  A function taking one arguments and returning void ∗.
- void ∗ generateStudents (void ∗args)

  A function taking one arguments and returning void ∗.
- void ∗ generateCourses (void ∗args)

  A function taking one arguments and returning void ∗.
- void ∗ genFen (void ∗args)

  A function taking one arguments and returning void ∗.
- void ∗ readpp (void ∗args)

  A function taking one arguments and returning void ∗.

4.3.1 Detailed Description

A Documented file. Details.

4.3.2 Function Documentation

4.3.2.1 void∗ generateCourses (void ∗ *args*)

A function taking one arguments and returning void ∗.

It calls the generate courses function and sends alarm request to timekeeper

Parameters

args	given due to pthread_create.

Returns

void ∗ due to pthread.

4.3.2.2 void∗ generateFaculty (void ∗ *args*)

A function taking one arguments and returning void ∗.

It calls the generate faculty function and sends alarm request to timekeeper

Parameters

args	given due to pthread_create.

Returns

void ∗ due to pthread.

4.3.2.3 void∗ generateStudents (void ∗ *args*)

A function taking one arguments and returning void ∗.

It calls the generate Students function and sends alarm request to timekeeper

Parameters

args	given due to pthread_create.

Returns

void ∗ due to pthread.

4.3.2.4 void∗ genFen (void ∗ *args*)

A function taking one arguments and returning void ∗.

It calls the generate friends function and sends alarm request to timekeeper

Parameters

args	given due to pthread_create.

Returns

void ∗ due to pthread.

4.3.2.5 void∗ readpp (void ∗ *args*)

A function taking one arguments and returning void ∗.

It recieves message from timekeeper and signals the thread

Parameters

args	it takes parameter simulation run time.

Returns

void ∗ due to pthread.

4.4 genfac.h File Reference

Generates faculty nodes and assigns random interest to them Details.

`#include "global1.h" #include "main_2.h"`

Functions

- void randominterest_fac (faculty ∗fc, dept ∗department)

  A function taking two arguments and returning void. It assigns random interests to a faculty from the interests of his department.
- void genfacof1dept (struct UNIV ∗univer, struct dept ∗depart, int seed)

  A function taking three arguments and returning void. Generates faculty nodes.

4.4.1 Detailed Description

Generates faculty nodes and assigns random interest to them Details.

4.4.2 Function Documentation

4.4.2.1 void genfacof1dept (struct UNIV ∗ *univer,* struct dept ∗ *depart,* int *seed*)

A function taking three arguments and returning void. Generates faculty nodes.

Parameters

seed	Random Seed Integer
depart	Department for which faculty are generated
univ	University for which faculty are generated

Returns

Void

4.4.2.2 void randominterest_fac (faculty * *fc*, dept * *department*)

A function taking two arguments and returning void. It assigns random interests to a faculty from the interests of his department.

Parameters

fc	Faculty node
department	Department of the faculty

Returns

Void

4.5 genFriend.h File Reference

Generates friends - abides by all the probability clauses. Uses re-entrant random class from C++11 for random no. generation.

`#include <math.h> #include <string.h>`

Functions

- int genFriends (void *args)

  A function taking one arguments and returning an integer value.

4.5.1 Detailed Description

Generates friends - abides by all the probability clauses. Uses re-entrant random class from C++11 for random no. generation.

4.5.2 Function Documentation

4.5.2.1 int genFriends (void * *args*)

A function taking one arguments and returning an integer value.

Parameters

t	Used to calculate next time for genfriends function call .

Returns

wake up time (which is sent to timekeeper)

4.6 genstudent.h File Reference

Generates student nodes and assigns random interest to them. Delete students after 4 years.

```
#include "global1.h" #include "main_2.h" #include <cstdlib> ×
#include <iostream> #include <string> #include <stdio.-
h> #include <stdlib.h> #include <map> #include <utility> ×
#include <vector> #include <time.h> #include <algorithm> ×
```

Functions

- void randominterest_st (student ∗st, dept ∗department)

  A function taking two arguments and returning void. It assigns random interests to a student.
- void genstuof1dept (float stupy, struct UNIV ∗univer, struct dept ∗depart, int seed, int batch)

  A function taking five arguments and returning void. It generates student nodes.
- void increase_year ()

  A function taking zero arguments and returning void. It increase the current year by one.
- void kill ()

  A function taking zero arguments and returning void. It kill the students after every four year.
- void kill_all ()

  A function taking zero arguments and returning void. It is called for cleanup at the end.

4.6.1 Detailed Description

Generates student nodes and assigns random interest to them. Delete students after 4 years.

4.6.2 Function Documentation

4.6.2.1 void genstuof1dept (float *stupy,* struct UNIV ∗ *univer,* struct dept ∗ *depart,* int *seed,* int *batch*)

A function taking five arguments and returning void. It generates student nodes.

Parameters

univer	University of the student
depart	Department of the student
seed	Student random seed
batch	Current entry year for which students are being created
stupy	Student per year that need to be generated

Returns

Void

4.6.2.2 void increase_year ( )

A function taking zero arguments and returning void. It increase the current year by one.

Returns

Void

4.6.2.3 void kill ( )

A function taking zero arguments and returning void. It kill the students after every four year.

Returns

Void

4.6.2.4 void kill_all ( )

A function taking zero arguments and returning void. It is called for cleanup at the end.

Returns

Void

4.6.2.5 void randominterest_st (student * *st,* dept * *department*)

A function taking two arguments and returning void. It assigns random interests to a student.

Parameters

st	Student node
department	Department of the student

Returns

Void

4.7 global1.h File Reference

Contains all the structs, classes and global variables used all over the simulation.

```
#include <pthread.h> #include <unistd.h> #include <iostream> ×
#include <string> #include <stdio.h> #include <sys/types.-
h> #include <stdlib.h> #include <map> #include <utility> ×
#include <vector> #include "Semaphore.h" #include <cstdlib> ×
#include <time.h> #include <algorithm>
```

Classes

- struct interest

  This structure stores all the intersets.
- struct hostels

  This structure stores all the hostels.
- struct houses

  This structure stores all the faculty houses.
- struct course
- struct student
- struct faculty
- struct dept
- struct UNIV
- class graph

  A graph class.

Variables

- vector< string > stud_name

  Vector containing names and surnames of students .
- vector< string > **stud_surname**
- pthread_mutex_t facMutex

  Mutexes .
- pthread_mutex_t **StudMutex**
- pthread_mutex_t **CourseMutex**
- pthread_mutex_t **FriendMutex**
- pthread_mutex_t **courseaffac**
- pthread_mutex_t **courseafStud**
- pthread_mutex_t **friAfAll**
- pthread_cond_t facCond

  Conditional variables .

- pthread_cond_t **StudCond**
- pthread_cond_t **CourseCond**
- pthread_cond_t **FriendCond**
- pthread_cond_t **condcoraffac**
- pthread_cond_t **condcorafStud**
- pthread_cond_t **friAf**
- int facRANDOM

  Random seed for faculty,student,course and friend .
- int **studRANDOM**
- int **courseRANDOM**
- int **friendRANDOM**
- int **p**
- int runThdStud

  Vector containing names and surnames of students .
- int **runThdCourse**
- int **runThdFriend**
- int **alloafgen**
- int **runThdfac**
- int **alloafStud**
- key_t gentotime

  Vector containing names and surnames of students .
- key_t **timetogen**
- int entryno

  Id's of students and faculty .
- int **facultyid**
- int **numberofsems**
- vector< string > prof_name

  Vector containing names and surnames of faculty .
- vector< string > **prof_surname**
- vector< UNIV > universities

  Vector containing all the university nodes .
- vector< faculty * > allfaculty

  Vector containing all the faculty nodes.
- vector< student * > allstudents

  Vector containing all the student nodes.
- map< string, interest * > allinterest

  Vector containing all the interest.
- graph * **g**

4.7.1 Detailed Description

Contains all the structs, classes and global variables used all over the simulation.

4.7.2 Variable Documentation

4.7.2.1 map<string,interest*> allinterest

Vector containing all the interest.

Interests are globbal across universities.

4.7.2.2 vector<student*> allstudents

Vector containing all the student nodes.

This vector gets updated after every kill and generate.

4.7.2.3 int entryno

Id's of students and faculty .

Both faculty and students are given their Id's sequentially

4.7.2.4 pthread_cond_t facCond

Conditional variables .

For pausing threads

4.7.2.5 pthread_mutex_t facMutex

Mutexes .

To lock various symbols

4.7.2.6 int facRANDOM

Random seed for faculty,student,course and friend .

Same as the ones in input file.

4.7.2.7 key_t gentotime

Vector containing names and surnames of students .

Used in random assignment of names to students

4.7.2.8 vector<string> prof_name

Vector containing names and surnames of faculty .

Used in random assignment of names to faculty.

4.7.2.9 int runThdStud

Vector containing names and surnames of students .

Used in random assignment of names to students

4.7.2.10 vector< string > stud_name

Vector containing names and surnames of students .

Used in random assignment of names to students

4.8 graphml.h File Reference

Generates the graph.graphml file Called after the simulation run is complete.

Functions

- void graphconverter ()

  A function taking zero arguments and returning void. It processes on available data and generates the graphml file .

4.8.1 Detailed Description

Generates the graph.graphml file Called after the simulation run is complete.

4.8.2 Function Documentation

4.8.2.1 void graphconverter ( )

A function taking zero arguments and returning void. It processes on available data and generates the graphml file .

Returns

Void

4.9 main_2.h File Reference

File Parser.

```
#include <iostream> #include <string.h> #include <stdio.-
h> #include <stdlib.h> #include <map> #include <vector> ×
#include <time.h>
```

Functions

- int setEnviro ()

  A function taking no arguments and returning an integer value.

4.9.1 Detailed Description

File Parser.

4.9.2 Function Documentation

4.9.2.1 int setEnviro ( )

A function taking no arguments and returning an integer value.

This function parses the input file and stores all the data.

Returns

return value as such of no further use

4.10 message_temp.h File Reference

A Documented file. Details.

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

Classes

- struct mymsgbuf

  This structure stores messgae struct for message queue.

Functions

- int open_queue (key_t t)

  A function taking one arguments and returning an integer value.
- int send_msg (int t, struct mymsgbuf ∗s)

A function taking two arguments and returning an integer value.

- int rec_msg (int, struct mymsgbuf ∗, long)

  A function taking two arguments and returning an integer value.

4.10.1 Detailed Description

A Documented file. Details.

4.10.2 Function Documentation

4.10.2.1 int open_queue (key_t t)

A function taking one arguments and returning an integer value.

Parameters

t	Used as a key for message queue.

Returns

id of the message queue.

4.10.2.2 int rec_msg (int , struct mymsgbuf ∗ , long)

A function taking two arguments and returning an integer value.

Parameters

t	queue id to identify the message queue
s	struct pointer in which message is to recieved

Returns

integer

4.10.2.3 int send_msg (int *t*, struct mymsgbuf ∗ *s*)

A function taking two arguments and returning an integer value.

Parameters

t	queue id to identify the message queue
s	struct pointer which has to be sent over queue

Returns

1 if message sending is successful

4.11 Semaphore.h File Reference

A Documented file. Details.

```
#include <stdio.h> #include <stdlib.h> #include <unistd.-
h>  #include <errno.h>  #include <sys/types.h>  #include
<sys/ipc.h> #include <sys/sem.h>
```

Functions

- int open_sem (key_t t, int s)

  A function taking two arguments and returning an integer value.
- void getSem (int a)

  A function taking one argument and having void return type.
- void relSem (int d)

  A function taking one argument and having void return type.

4.11.1 Detailed Description

A Documented file. Details.

4.11.2 Function Documentation

4.11.2.1 void getSem (int *a*)

A function taking one argument and having void return type.

Parameters

a	semaphore id acquire lock over the semaphore by decreasing semaphore value.

Returns

void

4.11.2.2 int open_sem (key_t *t*, int *s*)

A function taking two arguments and returning an integer value.

Parameters

t	Used to generate a semaphore.
s	Number of semaphores in a semaphore array.

Returns

id of the semaphore

4.11.2.3 void relSem (int *d*)

A function taking one argument and having void return type.

Parameters

d	release lock over the semaphore by increasing semaphore value.

Returns

void