DSA Project

Clique of people

Group: G612277

Brajan Miśkowicz
Adam Szałański
Łucja Pałkus

Chapter 1

User Documentation

The aim of this programming project is to manage a social network. This social network is formed by people that may be linked among each other if there is a friendship relationship between them.

If one runs program, menu options are displayed, where one can choose loading data from file with people, file with realtionships between people, displaying loaded data, saving current data to new files, deleting people (including all of their relationships) and deleting particular relationship.

One has to type file name without '.txt' extension if decides to load data from file with people or from file with relationships.
If one decides to display data from people or relationships files, data will be listed on screen.
One has to type file name without '.txt' extension when decides to save current data. If file has already existed, it will be overwritten.
One has to type person's ID if decides to delete person from data. All relationships that include this ID will be removed.
Deleting relationship between two people from data requires typing their IDs.

This program is protected from invalid input:
One cannot input string in menu options. Exception with message "Bad input" will appear on screen, but menu will reload.
Unexisting files cannot be loaded. Error "Such file does not exist" will be displayed, but program will not crash.
Unexisting person cannot be deleted. Exception "Person with that id does not exist" is thrown, but it doesn't interrupt the program.
Program doesn't crash if one tries to delete unexisting relationship.

Second version of the program allows users to retrieve friends of a person (given surname). If there are several people with the same surname then there is printed out a list of friends for each of those people. Moreover given a city once can retrieve all poeple born there, returning id and surname for each person. Having two dates program can return people born between them, sorted by birthplace, surname, name. This part retrieves peoples IDs from the provided file and finds them in the people net. Next takes their hometown and creates a list of people which birthplace matches that hometown. One of the possibilities in the menu is to build a list of classes, where classes contains users who likes the same collection of movies.

On the final version the program enables working on the social network and:
- checking if there can be made connection between two poeople in a maximum of six steps,
- returning the largest chain of different people linking two people (without duplicates),
- retrieving all the cliques of friends (crews) with more than 4 friends. A clique is a group of friends in which each person has friendship with each other.

Chapter 2

UML Diagram

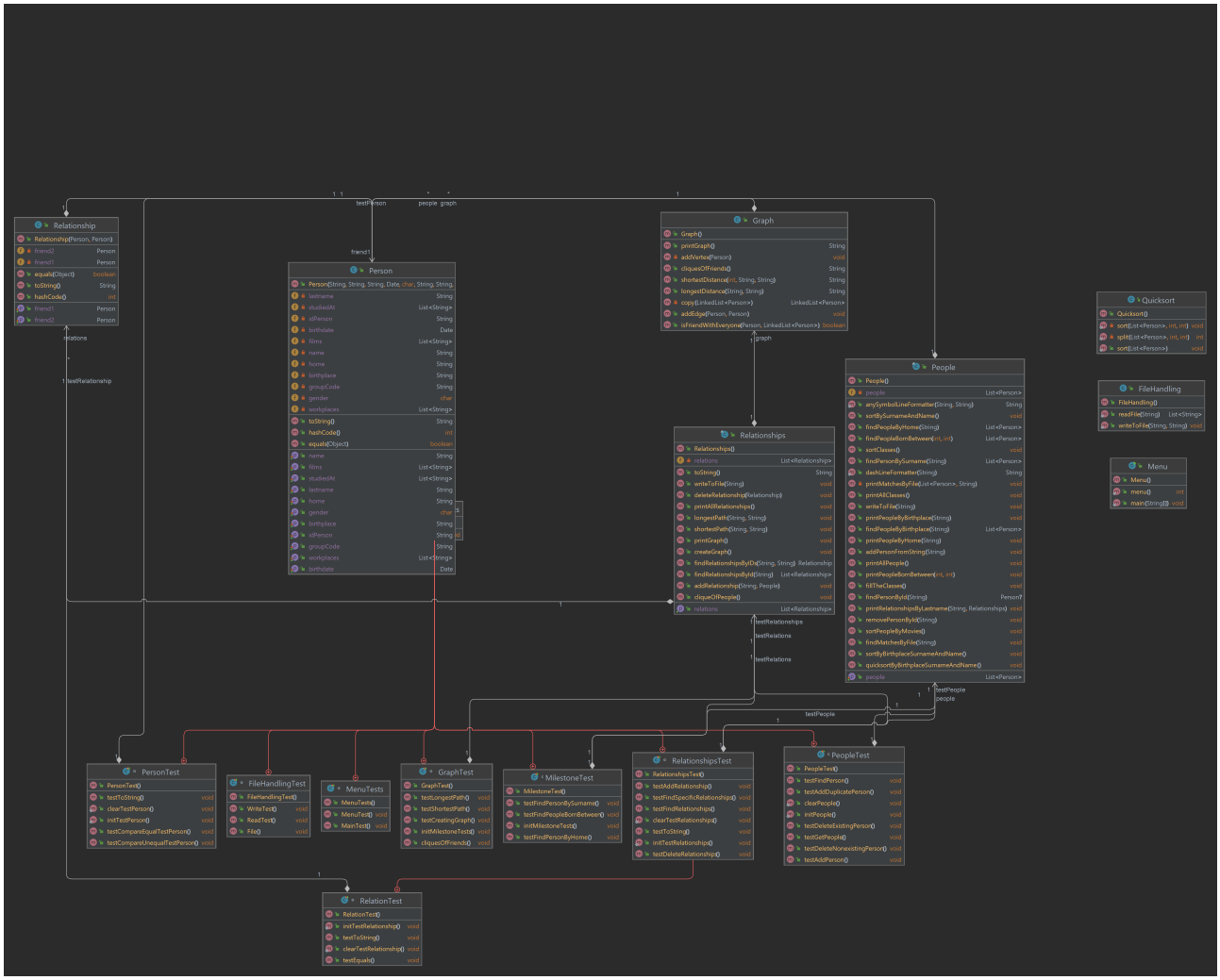


Figure 2.1 UML diagram of the classes

Chapter 3

Namespace Index

3.1 Package List

Here are the packages with brief descriptions (if available):

Chapter 4

Class Index

4.1 Class List

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

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

Chapter 6

Namespace Documentation

6.1 Package fileHandling

Classes

- class FileHandling

6.2 Package people

Classes

- class People
- class Person
- class Quicksort

6.3 Package relationships

Classes

- class Graph
- class Relationship
- class Relationships

Chapter 7

Class Documentation

7.1 fileHandling.FileHandling Class Reference

Static Public Member Functions

- static List< String > readFile (String fileName) throws FileNotFoundException
- static void writeToFile (String output, String fileName) throws IOException

7.1.1 Detailed Description

This class responsible for handling operations related to files It has classes which writes and rides text from/to files

7.1.2 Member Function Documentation

7.1.2.1 readFile()

```
static List< String > fileHandling.FileHandling.readFile (
            String fileName ) throws FileNotFoundException  [static]
```

This method read lines from the whole file.

Parameters

fileName	is String parameter with a file name

Returns

list of Strings

Here is the caller graph for this function:

7.1.2.2 writeToFile()

```
static void fileHandling.FileHandling.writeToFile (
            String output,
            String fileName ) throws IOException  [static]
```

This method write String to the file.

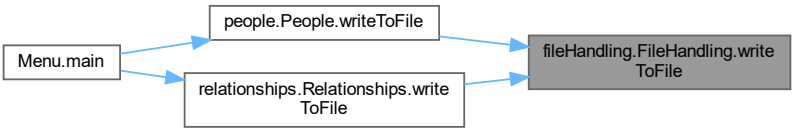
Parameters

output	String which will be written to file
fileName	String contains name of file where output will be written

Exceptions

IOException	This throws could occur when an I/O exception of some sort has occurred.

Here is the caller graph for this function:



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

- D:/Projekty/DSA_ehu_JavaProject/src/fileHandling/FileHandling.java

7.2 relationships.Graph Class Reference

Public Member Functions

- Graph ()
- void addEdge (Person source, Person destination)
- String printGraph ()
- String shortestDistance (int s, String origin, String dest)
- String longestDistance (String origin, String dest)
- String cliquesOfFriends ()
- boolean isFriendWithEveryone (Person person, LinkedList< Person > lists)

7.2.1 Detailed Description

This class implements a graph which represents the network. The nodes are the people in network, and edges are relationships between them Consists of list of relations and graph representing the network of relations @field graph is a Map with nodes and corresponding to them edges

7.2.2 Constructor & Destructor Documentation

7.2.2.1 Graph()

```
relationships.Graph.Graph ( )
```

Constructor for a Graph class.

7.2.3 Member Function Documentation

7.2.3.1 addEdge()

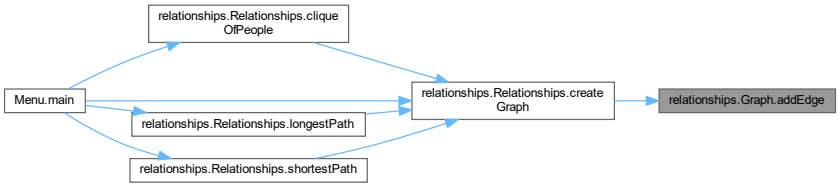
```
void relationships.Graph.addEdge (
            Person source,
            Person destination )
```

This function adds an edge between two given nodes and calls for creating nodes if they doesn't exist

Parameters

source	String id of first person
destination	String id of second person

Here is the caller graph for this function:

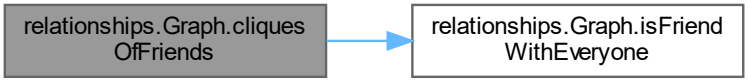


7.2.3.2 cliquesOfFriends()

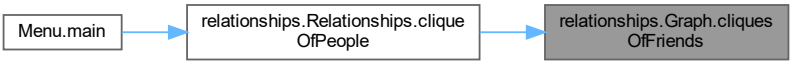
```
String relationships.Graph.cliquesOfFriends ( )
```

Creates the graph and prints cliques of people, where clique of people is a group of least 5 people where everyone has relationship with everyone

For every person it adds a linked list only with them into linked list of cliques of friends. Then for all linked list of clique it checks if the person is a friend with everyone, and if so it adds a new clique with this person added. At last, it prints every clique of friends with more than 4 participants. Here is the call graph for this function:



Here is the caller graph for this function:



7.2.3.3 isFriendWithEveryone()

```
boolean relationships.Graph.isFriendWithEveryone (
            Person person,
            LinkedList< Person > lists )
```

This method checks if a given person is a friend with everyone in given list Here is the caller graph for this function:

7.2.3.4 longestDistance()

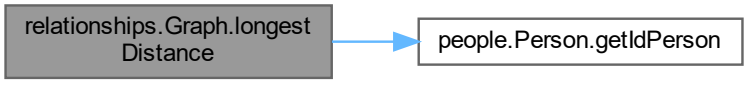
```
String relationships.Graph.longestDistance (
            String origin,
            String dest )
```

This function finds the longest path in graph between two people

Parameters

origin	String id of first person
dest	String id of second person

It creates a linked list to which it adds linked list with first person, and the second linked list that holds the longest path to destination the next step is to - in a loop - check if the last person in first linked list is designated person, and if so update the longest path linked list algorithm also adds a new linked list with added new node, where node is the friend of last person in first linked list, this step is repeated for every friend of the node and then the first linked list is removed - every linked list represents checked path. Here is the call graph for this function:



Here is the caller graph for this function:

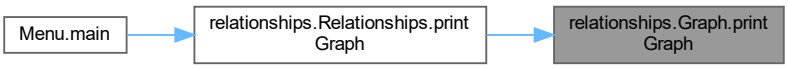
7.2.3.5 printGraph()

```
String relationships.Graph.printGraph ( )
```

Prints a graph as people and friends connected with them

Returns

Here is the caller graph for this function:



7.2.3.6 shortestDistance()

```
String relationships.Graph.shortestDistance (
            int s,
            String origin,
            String dest )
```

This function finds the shortest path in graph between two people under "s" graph nodes

Parameters

s	int, number of max nodes between searched people
origin	String id of first person
dest	String id of second person

It creates a linked list to which it adds linked list with first person, the next step is to add in a loop a new linked list with added new node, where node is the friend of last person in first linked list, this step is repeated for every friend of the node and then the first linked list is removed - every linked list represents checked path. The algorithm stops if the destination node is found under "s" steps and writes the path, and if the path is not found under 6 steps it informs about it. Here is the call graph for this function:

Here is the caller graph for this function:

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

- D:/Projekty/DSA_ehu_JavaProject/src/relationships/Graph.java

7.3 Menu Class Reference

Static Public Member Functions

- static void main (String[ ] args)
- static int menu ()

7.3.1 Detailed Description

This class is currently the main class of the projects. It creates the menu that allows user to insert People and Relationships and view them.

7.3.2 Member Function Documentation

7.3.2.1 main()

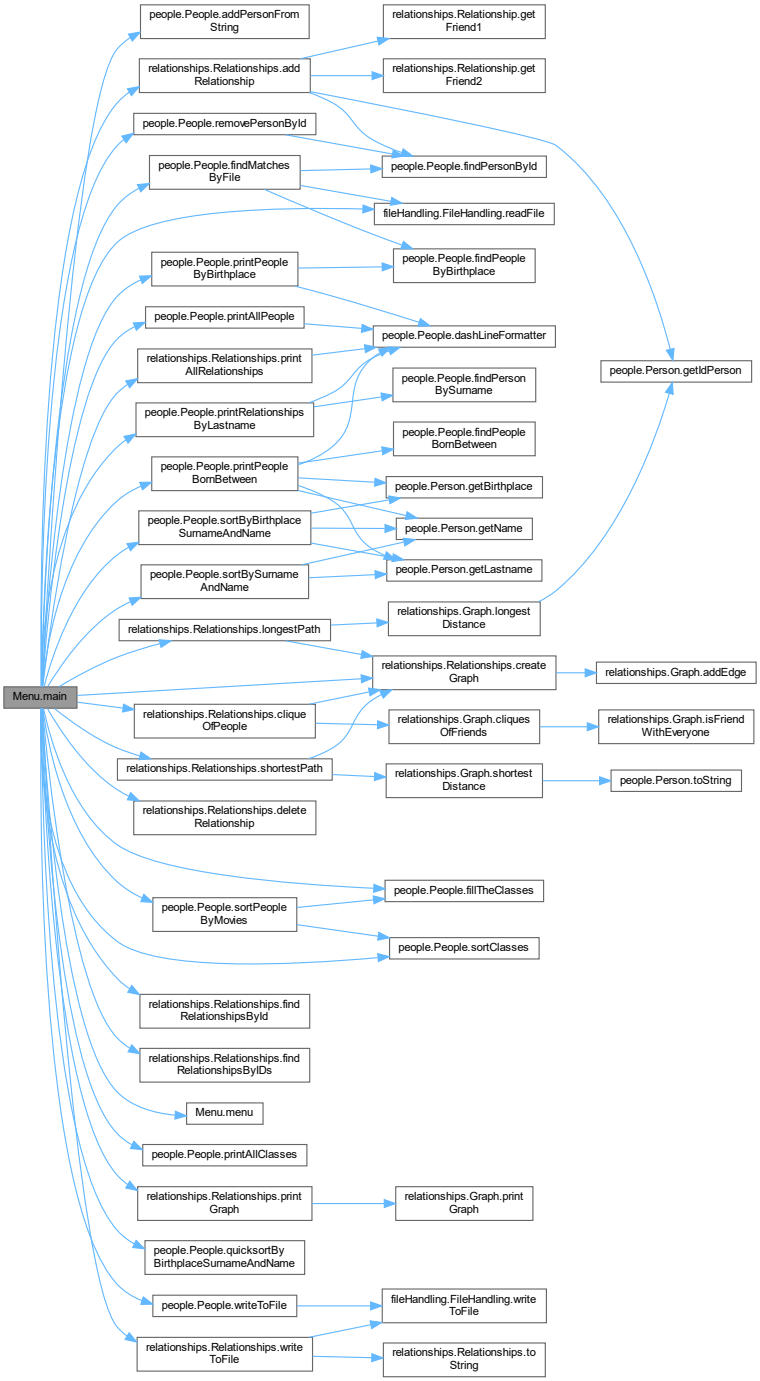
```
static void Menu.main (
            String[ ] args )  [static]
```

This method consist of calls to all functions related to the options presented in menu.

Parameters

args	

Here is the call graph for this function:



7.3.2.2 menu()

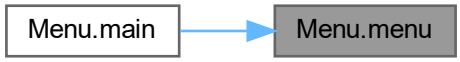
```
static int Menu.menu ( )  [static]
```

This method prints menu options of our programming project and reads input from the user.

Returns

integer

Here is the caller graph for this function:



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

- D:/Projekty/DSA_ehu_JavaProject/src/Menu.java

7.4 people.People Class Reference

Public Member Functions

- People ()
- List< Person > getPeople ()
- void setPeople (List< Person > people)
- void printAllPeople ()
- void addPersonFromString (String data) throws ParseException
- Person findPersonById (String id)
- List< Person > findPersonBySurname (String surname)
- void printRelationshipsByLastname (String lastname, Relationships relationships)
- List< Person > findPeopleByHome (String home)
- void printPeopleByHome (String home)
- List< Person > findPeopleByBirthplace (String birthplace)
- void printPeopleByBirthplace (String birthplace)
- List< Person > findPeopleBornBetween (int yearMin, int yearMax)
- void printPeopleBornBetween (int yearMin, int yearMax)
- void writeToFile (String filename) throws IOException
- void removePersonById (String id)
- void findMatchesByFile (String filename) throws IOException
- void sortBySurnameAndName ()
- void sortByBirthplaceSurnameAndName ()
- void quicksortByBirthplaceSurnameAndName ()
- void fillTheClasses ()
- void printAllClasses ()
- void sortClasses ()
- void sortPeopleByMovies ()

Static Public Member Functions

- static String dashLineFormatter (String line)
- static String anySymbolLineFormatter (String line, String symbol)

Public Attributes

- HashMap< List< String >, List< Person > > profiles = new HashMap<>()

Static Public Attributes

- static final String FIRST_LINE_PEOPLE = "idperson,name,lastname,birthdate,gender,birthplace,home,studiedat,workplaces,fil

7.4.1 Constructor & Destructor Documentation

7.4.1.1 People()

```
people.People.People ( )
```

Constructor for a People class.

7.4.2 Member Function Documentation

7.4.2.1 addPersonFromString()

```
void people.People.addPersonFromString (
            String data ) throws ParseException
```

This method check if the user ID is already in list of people, if not - add person from string to list of people

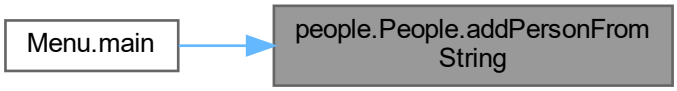
Parameters

data	is string containing data for creating a new object of Person class.

Exceptions

ParseException	Signals that an error has been reached unexpectedly while parsing.

Here is the caller graph for this function:



7.4.2.2 anySymbolLineFormatter()

```
static String people.People.anySymbolLineFormatter (
            String line,
            String symbol )  [static]
```

A little function that takes a line containing column names and returns a line of dashes adjusted to the length of passed line

Parameters

line	String with column names
symbol	String to be used as a delimiter line

Returns

String of passed symbol

7.4.2.3 dashLineFormatter()

```
static String people.People.dashLineFormatter (
            String line )  [static]
```

A little function that takes a line containing column names and returns a line of dashes adjusted to the length of passed line

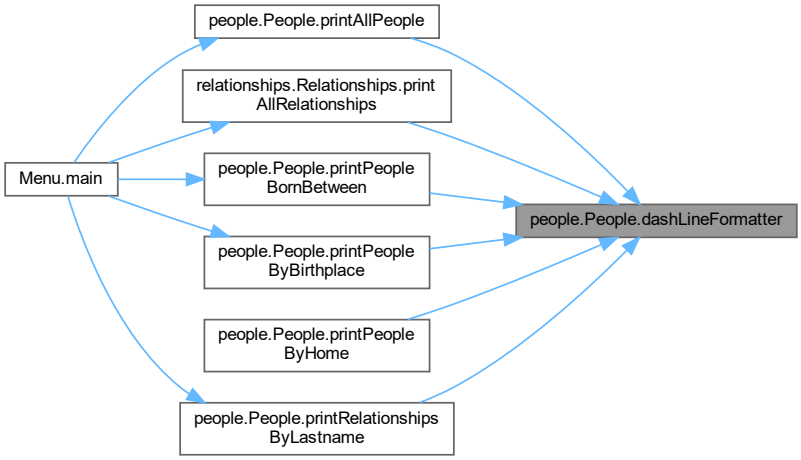
Parameters

line	String with column names

Returns

String of dashes

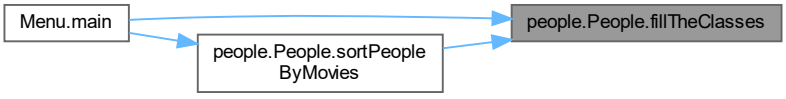
Here is the caller graph for this function:



7.4.2.4 fillTheClasses()

```
void people.People.fillTheClasses ( )
```

This function builds a classes of movies containing peoples who likes the same sets of movies Here is the caller graph for this function:



7.4.2.5 findMatchesByFile()

```
void people.People.findMatchesByFile (
            String filename ) throws IOException
```

The function takes the name of the .txt file, reads the contents of the file being the ID's of people and prints the lists of people whose birthplace is equal to hometown of people from file

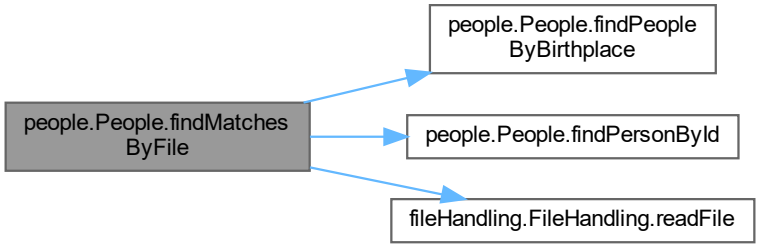
Parameters

filename	String - name of the file without extension

Exceptions

IOException	in case the file causes issues

Here is the call graph for this function:



Here is the caller graph for this function:

7.4.2.6 findPeopleBornBetween()

```
List< Person > people.People.findPeopleBornBetween (
            int yearMin,
            int yearMax )
```

This method finds a list of people born between passed years

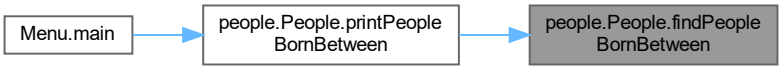
Parameters

yearMin	int
yearMax	int

Returns

List<Person> list of found people

Here is the caller graph for this function:



7.4.2.7 findPeopleByBirthplace()

```
List< Person > people.People.findPeopleByBirthplace (
            String birthplace )
```

This method finds a list of people with the passed birthplace

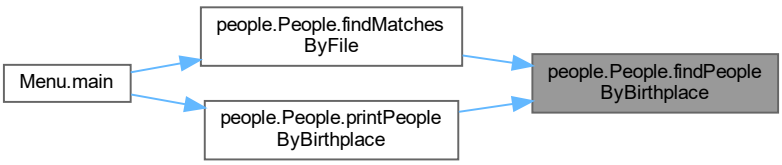
Parameters

birthplace	String

Returns

List<Person> list of found people

Here is the caller graph for this function:



7.4.2.8 findPeopleByHome()

```
List< Person > people.People.findPeopleByHome (
            String home )
```

This method finds a list of people with the passed hometown

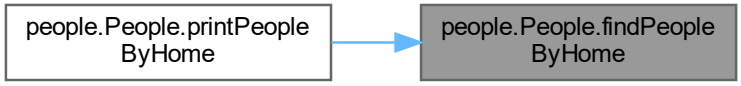
Parameters

home	String

Returns

List<Person> list of found people

Here is the caller graph for this function:



7.4.2.9 findPersonById()

```
Person people.People.findPersonById (
            String id )
```

This method finds Person object with corresponding ID in people list.

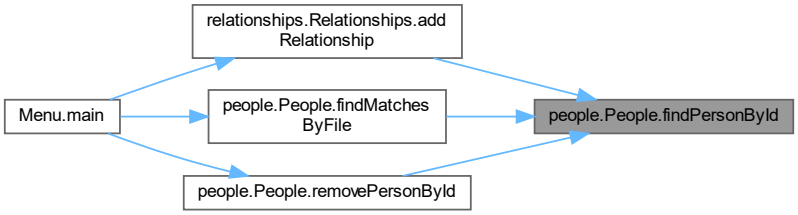
Parameters

id	string

Returns

Person object

Here is the caller graph for this function:



7.4.2.10 findPersonBySurname()

```
List< Person > people.People.findPersonBySurname (
            String surname )
```

This method finds List of Person objects with corresponding surname in people list.

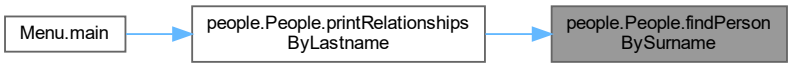
Parameters

surname	string

Returns

List<Person> object

Here is the caller graph for this function:



7.4.2.11 getPeople()

```
List< Person > people.People.getPeople ( )
```

Getter for a people field.

Returns

list of Person type objects.

7.4.2.12 printAllClasses()

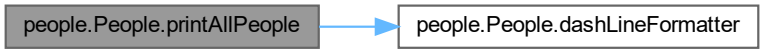
```
void people.People.printAllClasses ( )
```

This function prints all movies classes and all the people who like them Here is the caller graph for this function:

7.4.2.13 printAllPeople()

```
void people.People.printAllPeople ( )
```

This method prints out the list of all the people loaded to the program. Here is the call graph for this function:



Here is the caller graph for this function:

7.4.2.14 printPeopleBornBetween()

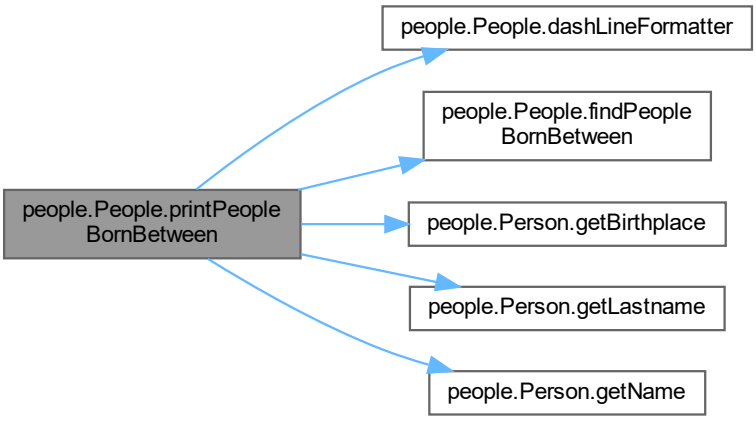
```
void people.People.printPeopleBornBetween (
            int yearMin,
            int yearMax )
```

This method prints people born between passed years

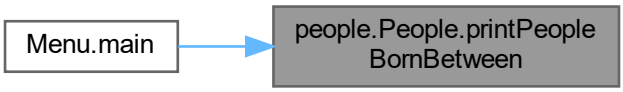
Parameters

yearMin	int
yearMax	int

Here is the call graph for this function:



Here is the caller graph for this function:



7.4.2.15 printPeopleByBirthplace()

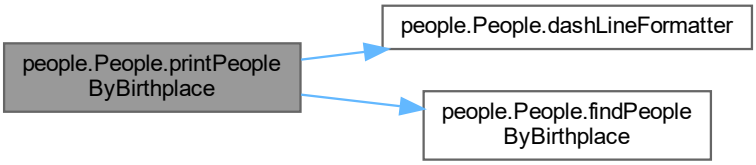
```
void people.People.printPeopleByBirthplace (
            String birthplace )
```

This method prints people with a given birthplace

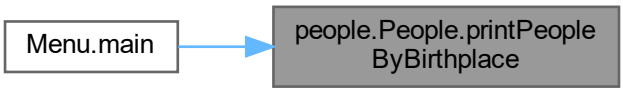
Parameters

birthplace	String

Here is the call graph for this function:



Here is the caller graph for this function:



7.4.2.16 printPeopleByHome()

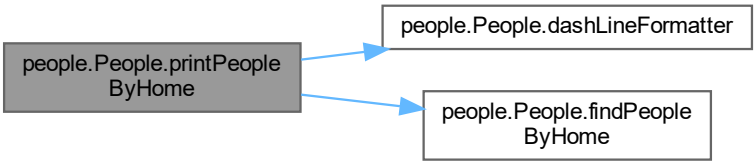
```
void people.People.printPeopleByHome (
            String home )
```

This method prints people with a given hometown

Parameters

home	String

Here is the call graph for this function:



7.4.2.17 printRelationshipsByLastname()

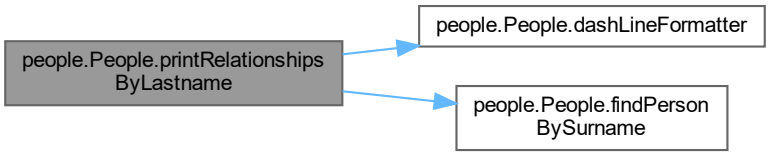
```
void people.People.printRelationshipsByLastname (
            String lastname,
            Relationships relationships )
```

This method prints relationships of people with a given lastname

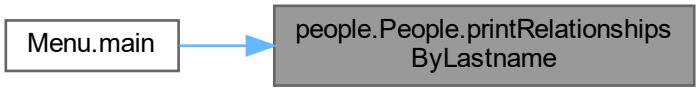
Parameters

lastname	String
relationships	Relationships

Here is the call graph for this function:



Here is the caller graph for this function:



7.4.2.18 quicksortByBirthplaceSurnameAndName()

```
void people.People.quicksortByBirthplaceSurnameAndName ( )
```

This function sorts people by birthplace surname and name with QuickSort Here is the caller graph for this function:

7.4.2.19 removePersonById()

```
void people.People.removePersonById (
            String id )
```

This method removes person with a certain id from the list of people If person with that id does not exist, prints out a message

Parameters

id	String

Here is the call graph for this function:

Here is the caller graph for this function:

7.4.2.20 setPeople()

```
void people.People.setPeople (
            List< Person > people )
```

Setter for a people field.

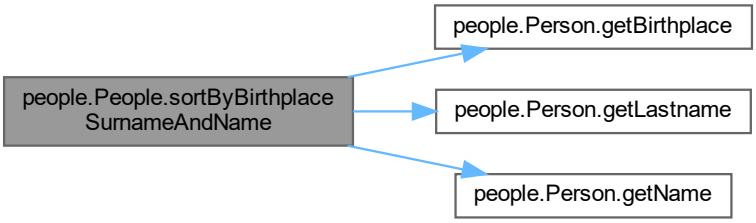
Parameters

people	is list of Person type objects.

7.4.2.21 sortByBirthplaceSurnameAndName()

```
void people.People.sortByBirthplaceSurnameAndName ( )
```

This function sorts people by birthplace surname and name with build-in sort Here is the call graph for this function:

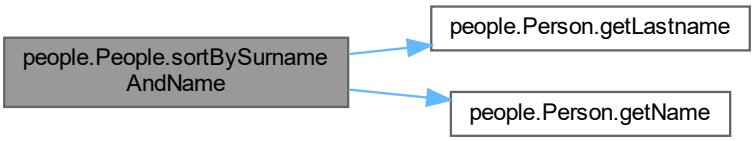


Here is the caller graph for this function:

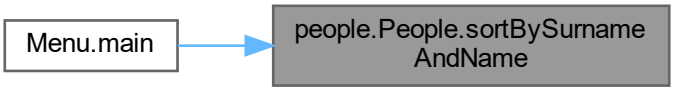
7.4.2.22 sortBySurnameAndName()

```
void people.People.sortBySurnameAndName ( )
```

This function sorts people by surname and name with build-in sort Here is the call graph for this function:



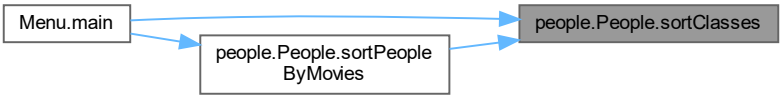
Here is the caller graph for this function:



7.4.2.23 sortClasses()

```
void people.People.sortClasses ( )
```

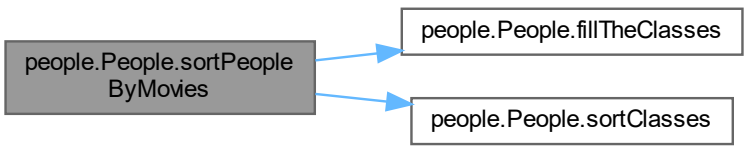
This function sorts the classes of movies by number of people who likes them Here is the caller graph for this function:



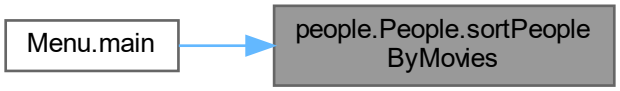
7.4.2.24 sortPeopleByMovies()

```
void people.People.sortPeopleByMovies ( )
```

This function sorts people by most common movies sets it creates the classes, sort them and then get sorted people from hashmap Here is the call graph for this function:



Here is the caller graph for this function:



7.4.2.25 writeToFile()

```
void people.People.writeToFile (
            String filename ) throws IOException
```

This method writes string to given file name.

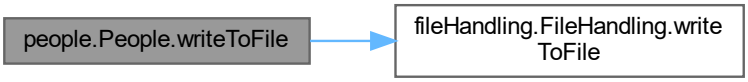
Parameters

filename	string

Exceptions

IOException	This throws could occur when writing to a file

Here is the call graph for this function:



Here is the caller graph for this function:

7.4.3 Member Data Documentation

7.4.3.1 FIRST_LINE_PEOPLE

```
final String people.People.FIRST_LINE_PEOPLE = "idperson,name,lastname,birthdate,gender,birthplace,home,studie
[static]
```

7.4.3.2 profiles

```
HashMap<List<String>, List<Person> > people.People.profiles = new HashMap<>()
```

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

- D:/Projekty/DSA_ehu_JavaProject/src/people/People.java

7.5 people.Person Class Reference

Public Member Functions

- boolean equals (Object o)
- int hashCode ()
- Person (String idPerson, String name, String lastname, Date birthdate, char gender, String birthplace, String home, List< String > studiedAt, List< String > workplaces, List< String > films, String groupCode)
- String getIdPerson ()
- void setIdPerson (String idPerson)
- String getName ()
- void setName (String name)
- String getLastname ()
- void setLastname (String lastname)
- Date getBirthdate ()
- void setBirthdate (Date birthdate)
- char getGender ()
- void setGender (char gender)
- String getBirthplace ()
- void setBirthplace (String birthplace)
- String getHome ()
- void setHome (String home)
- List< String > getStudiedAt ()
- void setStudiedAt (List< String > studiedAt)
- List< String > getWorkplaces ()
- void setWorkplaces (List< String > workplaces)
- List< String > getFilms ()
- void setFilms (List< String > films)
- String getGroupCode ()
- void setGroupCode (String groupCode)
- String toString ()

7.5.1 Constructor & Destructor Documentation

7.5.1.1 Person()

```
people.Person.Person (
            String idPerson,
            String name,
            String lastname,
            Date birthdate,
            char gender,
            String birthplace,
            String home,
            List< String > studiedAt,
            List< String > workplaces,
            List< String > films,
            String groupCode )
```

All-argument constructor for a Person class.

Parameters

idPerson	String contains user ID
name	String contains username
lastname	String contains lastname
birthdate	Date contains birthdate
gender	char contains 'M' as 'male' / 'F' as 'female'
birthplace	String contains birthplace
home	String contains name of hometown
studiedAt	list of Strings contains name of university
workplaces	list of Strings contains workplaces
films	list of Strings contains favourites movie titles
groupCode	String contains group code

7.5.2 Member Function Documentation

7.5.2.1 equals()

```
boolean people.Person.equals (
            Object o )
```

This method compares this object with object given in parameter. It overrides default method and allows us to use '==' operator.

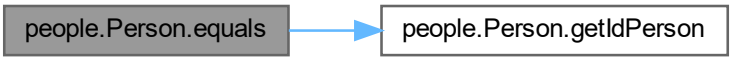
Parameters

o	is an object of undefined type

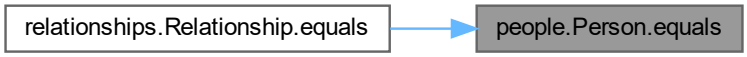
Returns

boolean

Here is the call graph for this function:



Here is the caller graph for this function:



7.5.2.2 getBirthdate()

```
Date people.Person.getBirthdate ( )
```

Getter for a birthdate field.

Returns

Date type object.

7.5.2.3 getBirthplace()

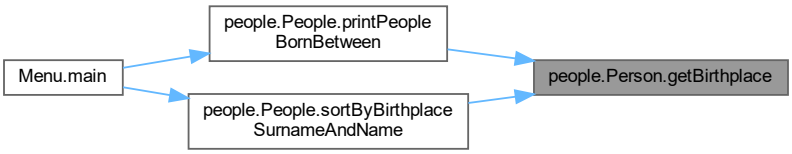
```
String people.Person.getBirthplace ( )
```

Getter for a birthplace field.

Returns

String type object.

Here is the caller graph for this function:



7.5.2.4 getFilms()

```
List< String > people.Person.getFilms ( )
```

Getter for a films field.

Returns

List of String type objects.

7.5.2.5 getGender()

```
char people.Person.getGender ( )
```

Getter for a gender field.

Returns

char type object.

7.5.2.6 getGroupCode()

```
String people.Person.getGroupCode ( )
```

Getter for a groupCode field.

Returns

String type object.

7.5.2.7 getHome()

```
String people.Person.getHome ( )
```

Getter for a getHome field.

Returns

String type object.

7.5.2.8 getIdPerson()

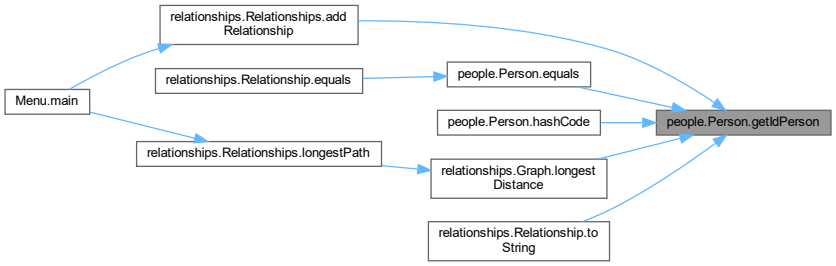
```
String people.Person.getIdPerson ( )
```

Getter for a idPerson field.

Returns

String type object.

Here is the caller graph for this function:



7.5.2.9 getLastname()

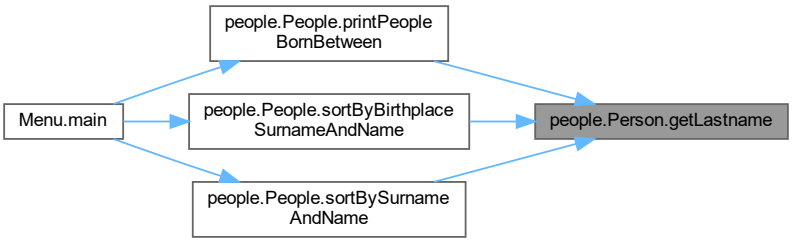
```
String people.Person.getLastname ( )
```

Getter for a lastname field.

Returns

String type object.

Here is the caller graph for this function:



7.5.2.10 getName()

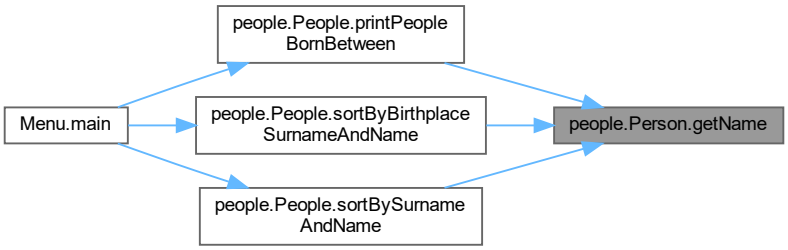
```
String people.Person.getName ( )
```

Getter for a name field.

Returns

String type object.

Here is the caller graph for this function:



7.5.2.11 getStudiedAt()

```
List< String > people.Person.getStudiedAt ( )
```

Getter for a studiedAt field.

Returns

List of String type objects.

7.5.2.12 getWorkplaces()

```
List< String > people.Person.getWorkplaces ( )
```

Getter for a workplaces field.

Returns

List of String type objects.

7.5.2.13 hashCode()

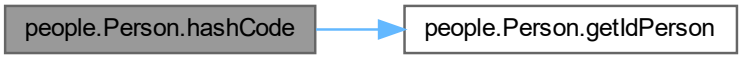
```
int people.Person.hashCode ( )
```

This method overrides default method and returns the integer hash code value of the object.

Returns

integer

Here is the call graph for this function:



7.5.2.14 setBirthdate()

```
void people.Person.setBirthdate (
            Date birthdate )
```

Setter for a birthdate field.

Parameters

birthdate	is a Date type object.

7.5.2.15 setBirthplace()

```
void people.Person.setBirthplace (
            String birthplace )
```

Setter for a birthplace field.

Parameters

birthplace	is a String type object.

7.5.2.16 setFilms()

```
void people.Person.setFilms (
            List< String > films )
```

Setter for a films field.

Parameters

films	is a list of String type object.

7.5.2.17 setGender()

```
void people.Person.setGender (
            char gender )
```

Setter for a gender field.

Parameters

gender	is a char type object.

7.5.2.18 setGroupCode()

```
void people.Person.setGroupCode (
            String groupCode )
```

Setter for a groupCode field.

Parameters

groupCode	is a String type object.

7.5.2.19 setHome()

```
void people.Person.setHome (
            String home )
```

Setter for a home field.

Parameters

home	is a String type object.

7.5.2.20 setIdPerson()

```
void people.Person.setIdPerson (
            String idPerson )
```

Setter for a idPerson field.

Parameters

idPerson	is a String type object.

7.5.2.21 setLastname()

```
void people.Person.setLastname (
            String lastname )
```

Setter for a lastname field.

Parameters

lastname	is a String type object.

7.5.2.22 setName()

```
void people.Person.setName (
            String name )
```

Setter for a name field.

Parameters

name	is a String type object.

7.5.2.23 setStudiedAt()

```
void people.Person.setStudiedAt (
            List< String > studiedAt )
```

Setter for a studiedAt field.

Parameters

studied↩ At	is a list of String type object.

7.5.2.24 setWorkplaces()

```
void people.Person.setWorkplaces (
            List< String > workplaces )
```

Setter for a workplaces field.

Parameters

workplaces	is a list of String type object.

7.5.2.25 toString()

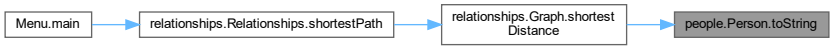
```
String people.Person.toString ( )
```

This method overrides default method and returns a string representation of this object.

Returns

string

Here is the caller graph for this function:



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

- D:/Projekty/DSA_ehu_JavaProject/src/people/Person.java

7.6 people.Quicksort Class Reference

Static Public Member Functions

- static< T extends Comparable< T > void sort (List< Person > list)

7.6.1 Member Function Documentation

7.6.1.1 sort()

```
static< T extends Comparable< T > void people.Quicksort.sort (
            List< Person > list )  [static]
```

first method of sorting

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

- D:/Projekty/DSA_ehu_JavaProject/src/people/Quicksort.java

7.7 relationships.Relationship Class Reference

Public Member Functions

- String toString ()
- boolean equals (Object o)
- int hashCode ()
- Relationship (Person friend1, Person friend2)
- Person getFriend1 ()
- void setFriend1 (Person friend1)
- Person getFriend2 ()
- void setFriend2 (Person friend2)

7.7.1 Detailed Description

This class represents a single relationship between two people. Consists of two objects of type Person. @field friend1 an absolute URL giving the base location of the image @field friend2 the location of the image, relative to the url argument

7.7.2 Constructor & Destructor Documentation

7.7.2.1 Relationship()

```
relationships.Relationship.Relationship (
            Person friend1,
            Person friend2 )
```

Constructor for a Relationships class.

Parameters

friend1	is parameter of Person type.
friend2	is parameter of Person type.

7.7.3 Member Function Documentation

7.7.3.1 equals()

```
boolean relationships.Relationship.equals (
            Object o )
```

This method compares this object with object given in parameter. It overrides default method and allows us to use '==' operator.

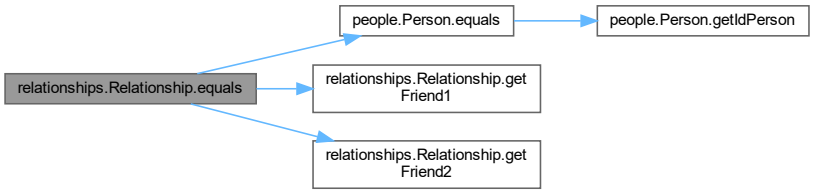
Parameters

o	is an object of undefined type

Returns

boolean

Here is the call graph for this function:



7.7.3.2 getFriend1()

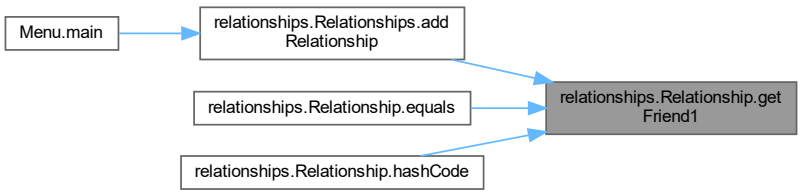
```
Person relationships.Relationship.getFriend1 ( )
```

Getter for a friend1 field.

Returns

object of Person type.

Here is the caller graph for this function:



7.7.3.3 getFriend2()

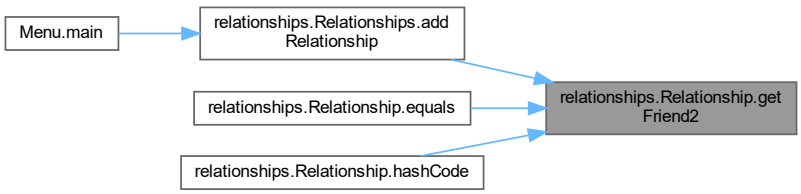
```
Person relationships.Relationship.getFriend2 ( )
```

Getter for a friend2 field.

Returns

object of Person type.

Here is the caller graph for this function:



7.7.3.4 hashCode()

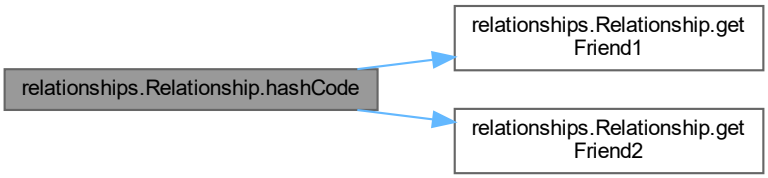
```
int relationships.Relationship.hashCode ( )
```

This method overrides default method and returns the integer hash code value of the object.

Returns

integer

Here is the call graph for this function:



7.7.3.5 setFriend1()

```
void relationships.Relationship.setFriend1 (
            Person friend1 )
```

Setter for a friend1 field.

Parameters

friend1	is parameter of Person type.

7.7.3.6 setFriend2()

```
void relationships.Relationship.setFriend2 (
            Person friend2 )
```

Setter for a friend2 field.

Parameters

friend2	is parameter of Person type.

7.7.3.7 toString()

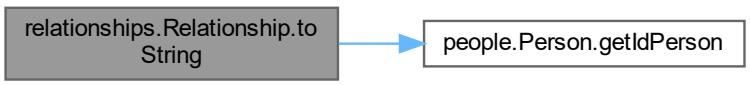
```
String relationships.Relationship.toString ( )
```

This method returns a string representation of this object.

Returns

string

Here is the call graph for this function:

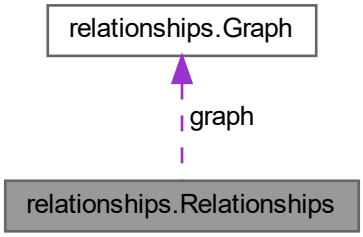


The documentation for this class was generated from the following file:

- D:/Projekty/DSA_ehu_JavaProject/src/relationships/Relationship.java

7.8 relationships.Relationships Class Reference

Collaboration diagram for relationships.Relationships:



Public Member Functions

- Relationships ()
- List< Relationship > getRelations ()
- void addRelationship (String list, People ppl)
- String toString ()
- void writeToFile (String filename) throws IOException
- List< Relationship > findRelationshipsById (String id)
- void deleteRelationship (Relationship r)
- Relationship findRelationshipsByIDs (String id1, String id2)
- void printAllRelationships ()
- void createGraph ()
- void printGraph ()
- void shortestPath (String id1, String id2)
- void longestPath (String id1, String id2)
- void cliqueOfPeople ()

Public Attributes

- Graph graph

Static Public Attributes

- static final String FIRST_LINE_RELATIONSHIPS = "friend1,friend2\n"

7.8.1 Detailed Description

This class represents a relationships in the network Consists of list of relations and graph representing the network of relations @field relations is a List of objects of type Relationship @field graph is an object of type Graph, it consists of nodes of people, and edges represents relationships

7.8.2 Constructor & Destructor Documentation

7.8.2.1 Relationships()

```
relationships.Relationships.Relationships ( )
```

Constructor for a Relationships class.

7.8.3 Member Function Documentation

7.8.3.1 addRelationship()

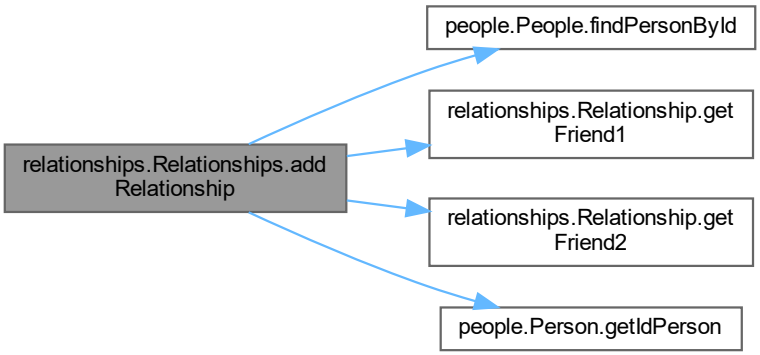
```
void relationships.Relationships.addRelationship (
            String list,
            People ppl )
```

This method check if relationship for two people exists. If not - create relationship between two people and add this relationship to list of relationships.

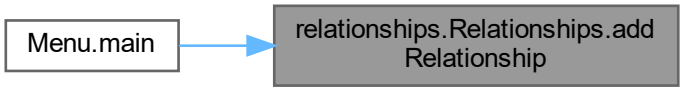
Parameters

list	This parameter is string list which consists two user IDs.
ppl	This parameter is People type object which have list of all users

Here is the call graph for this function:



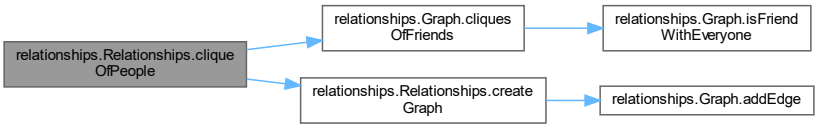
Here is the caller graph for this function:



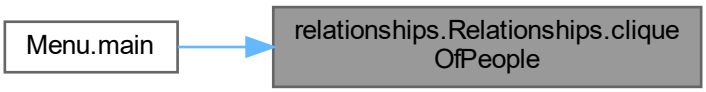
7.8.3.2 cliqueOfPeople()

```
void relationships.Relationships.cliqueOfPeople ( )
```

Calls for creating the graph and printing cliques of people clique of people is a group of least 5 people where everyone has relationship with everyone Here is the call graph for this function:



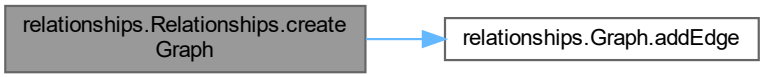
Here is the caller graph for this function:



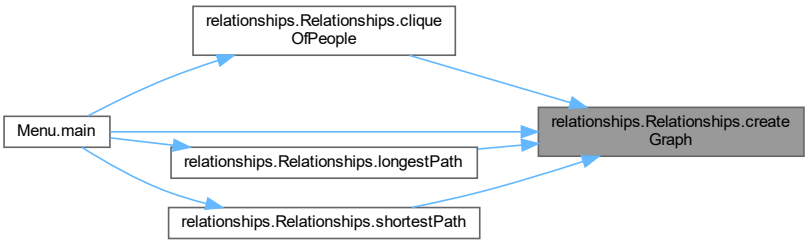
7.8.3.3 createGraph()

```
void relationships.Relationships.createGraph ( )
```

Method that for every relationship calls for function that adds edges and nodes to graph Here is the call graph for this function:



Here is the caller graph for this function:



7.8.3.4 deleteRelationship()

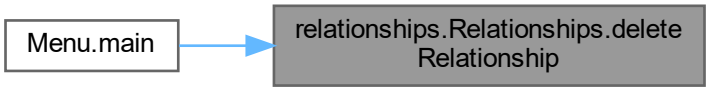
```
void relationships.Relationships.deleteRelationship (
            Relationship r )
```

Removes passed relationship from the list of relationships

Parameters

r	Relation

Here is the caller graph for this function:



7.8.3.5 findRelationshipsById()

```
List< Relationship > relationships.Relationships.findRelationshipsById (
            String id )
```

This method finds every relationship containing passed id

Parameters

id	String

Returns

List<Relationship> List of relationships containing that id

Here is the caller graph for this function:

7.8.3.6 findRelationshipsByIDs()

```
Relationship relationships.Relationships.findRelationshipsByIDs (
            String id1,
            String id2 )
```

This method finds relationship containing passed IDs

Parameters

id1	String
id2	String

Returns

Relationship containing passed IDs

Here is the caller graph for this function:

7.8.3.7 getRelations()

```
List< Relationship > relationships.Relationships.getRelations ( )
```

Getter for a relations field.

Returns

list of Relationship objects

7.8.3.8 longestPath()

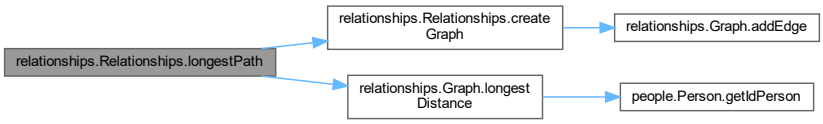
```
void relationships.Relationships.longestPath (
            String id1,
            String id2 )
```

Calls for creating the graph and finding longest path in graph between two people

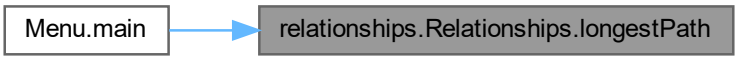
Parameters

id1	String id of first person
id2	String id of second person

Here is the call graph for this function:



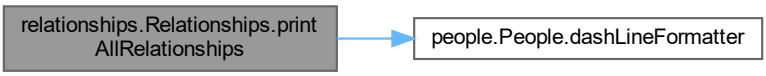
Here is the caller graph for this function:



7.8.3.9 printAllRelationships()

```
void relationships.Relationships.printAllRelationships ( )
```

This method prints out every relationship stored in the program Here is the call graph for this function:

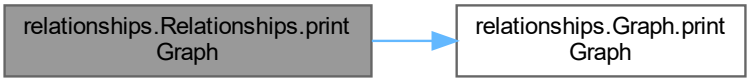


Here is the caller graph for this function:

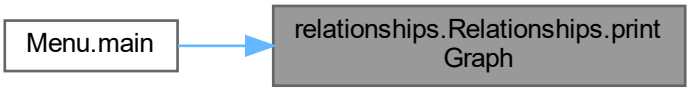
7.8.3.10 printGraph()

```
void relationships.Relationships.printGraph ( )
```

Calls for function that prints graph Here is the call graph for this function:



Here is the caller graph for this function:



7.8.3.11 shortestPath()

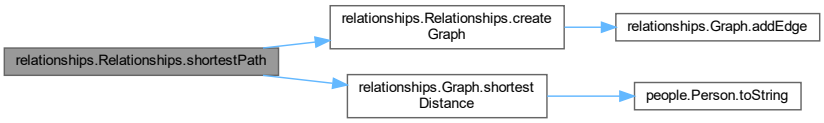
```
void relationships.Relationships.shortestPath (
            String id1,
            String id2 )
```

Calls for creating the graph and finding shortest path in graph between two people under 6 graph nodes

Parameters

id1	String id of first person
id2	String id of second person

Here is the call graph for this function:



Here is the caller graph for this function:

7.8.3.12 toString()

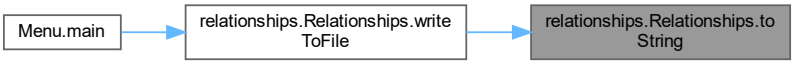
```
String relationships.Relationships.toString ( )
```

This method overrides default method and returns a string representation of this object.

Returns

string

Here is the caller graph for this function:



7.8.3.13 writeToFile()

```
void relationships.Relationships.writeToFile (
            String filename ) throws IOException
```

This method writes string to given file name.

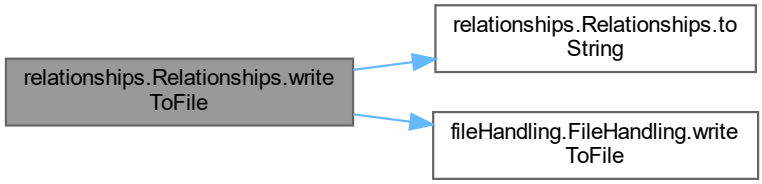
Parameters

filename	string

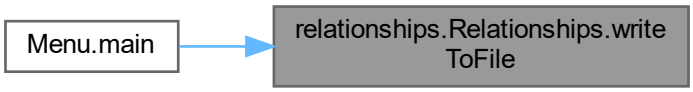
Exceptions

IOException	This throws could occur when writing to a file

Here is the call graph for this function:



Here is the caller graph for this function:



7.8.4 Member Data Documentation

7.8.4.1 FIRST_LINE_RELATIONSHIPS

```
final String relationships.Relationships.FIRST_LINE_RELATIONSHIPS = "friend1,friend2\n"  [static]
```

7.8.4.2 graph

```
Graph relationships.Relationships.graph
```

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

- D:/Projekty/DSA_ehu_JavaProject/src/relationships/Relationships.java

7.9 UnitTests Class Reference

Classes

- class **FileHandlingTest**
- class **GraphTest**
- class **MenuTests**
- class **MilestoneTest**
- class **PeopleTest**
- class **PersonTest**
- class **RelationshipsTest**

Public Member Functions

- void done ()

7.9.1 Member Function Documentation

7.9.1.1 done()

```
void UnitTests.done ( )
```

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

- D:/Projekty/DSA_ehu_JavaProject/src/UnitTests.java

Chapter 8

File Documentation

8.1 D:/Projekty/DSA_ehu_JavaProject/src/fileHandling/FileHandling.java File Reference

Classes

- class fileHandling.FileHandling

Packages

- package fileHandling

8.2 D:/Projekty/DSA_ehu_JavaProject/src/Menu.java File Reference

Classes

- class Menu

8.3 D:/Projekty/DSA_ehu_JavaProject/src/people/People.java File Reference

Classes

- class people.People

Packages

- package people

8.4 D:/Projekty/DSA_ehu_JavaProject/src/people/Person.java File Reference

Classes

- class people.Person

Packages

- package people

8.5 D:/Projekty/DSA_ehu_JavaProject/src/people/Quicksort.java File Reference

Classes

- class people.Quicksort

Packages

- package people

8.6 D:/Projekty/DSA_ehu_JavaProject/src/relationships/Graph.java File Reference

Classes

- class relationships.Graph

Packages

- package relationships

8.7 D:/Projekty/DSA_ehu_JavaProject/src/relationships/↩ Relationship.java File Reference

Classes

- class relationships.Relationship

Packages

- package relationships

8.8 D:/Projekty/DSA_ehu_JavaProject/src/relationships/↩ Relationships.java File Reference

Classes

- class relationships.Relationships

Packages

- package relationships

8.9 D:/Projekty/DSA_ehu_JavaProject/src/UnitTests.java File Reference

Classes

- class UnitTests
- class **UnitTests.MenuTests**
- class **UnitTests.FileHandlingTest**
- class **UnitTests.PeopleTest**
- class **UnitTests.PersonTest**
- class **UnitTests.RelationshipsTest**
- class **UnitTests.RelationshipsTest.RelationTest**
- class **UnitTests.MilestoneTest**
- class **UnitTests.GraphTest**

Index