Genetic Algorithm Solving Travelling Salesman Problem

Generated by Doxygen 1.8.18

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Namespace Documentation

5.1 FileIO Namespace Reference

The functions of processing input and output files.

Functions

- std::string checkFile (std::string path)

  Validates the path and extension of the input file.
- void geneticFile (std::string path, int number, std::shared_ptr< Path > &best, std::shared_ptr< std::vector< City >> &cities)

  Output file for the genetic algorithm solution.
- void bruteFile (std::string path, std::shared_ptr< Path > &best, std::shared_ptr< std::vector< City >> &cities)

  Output file for the brute force solution.
- std::vector< std::vector< double > > readDist (std::string file)

  Reads the input from the "DIST" type input file.
- std::vector< sf::Vector2f > readCoord (std::string file)

  Reads the input from the "COORD" type input file.

5.1.1 Detailed Description

The functions of processing input and output files.

5.1.2 Function Documentation

5.1.2.1 bruteFile()

```
void FileIO::bruteFile (
            std::string path,
            std::shared_ptr< Path > & best,
            std::shared_ptr< std::vector< City >> & cities )
```

Output file for the brute force solution.

Generates the output file for the brute force solution.

Parameters

	path	Full pathname of the file.
in,out	*best*	The best path.
in,out	*cities*	Cities.

Exceptions

FileOpen	if the file cannot be created.
FilePath	if the path is invalid.

5.1.2.2 checkFile()

```
std::string FileIO::checkFile (
            std::string path )
```

Validates the path and extension of the input file.

Checks the path and extension of the input file. Returns string containing the type of the input.

Parameters

path	Full pathname of the file.

Exceptions

FilePathExt	if the path or extension is invalid.
FileOpen	if the file cannot be opened.

Returns

String containing the input type.

5.1.2.3 geneticFile()

```
void FileIO::geneticFile (
            std::string path,
            int number,
            std::shared_ptr< Path > & best,
            std::shared_ptr< std::vector< City >> & cities )
```

Output file for the genetic algorithm solution.

Generates the output file for the genetic algorithm solution.

Parameters

	path	Full pathname of the file.
	number	Generation number.
in,out	*best*	The best path.
in,out	*cities*	Cities.

Exceptions

FileOpen	if the file cannot be created.
FilePath	if the path is invalid.

5.1.2.4 readCoord()

```
std::vector< sf::Vector2f > FileIO::readCoord (
            std::string file )
```

Reads the input from the "COORD" type input file.

Reads the input from a file with "COORD" input type. Input type "COORD" means that the file contains lines with x and y coordinates of each city.

Parameters

file	Full pathname of the file.

Exceptions

std::invalid_argument	if the number of cities is less than 2 or it's not an integer.
FileInput	if the coordinates are invalid, with the number of the wrong line in the file.
CitiesAmount	if there is not enough data.
FileOpen	if the file cannot be opened.

Returns

Vector with coordinates of the cities.

5.1.2.5 readDist()

```
std::vector< std::vector< double > > FileIO::readDist (
            std::string file )
```

Reads the input from the "DIST" type input file.

Reads the input from a file with "DIST" input type. Input type "DIST" means that the file contains matrix with distances between the cities.

Parameters

file	Full pathname of the file.

Exceptions

std::invalid_argument	if the number of cities is less than 2 or it's not an integer.
FileInput	if the data in the matrix is invalid, with the number of the wrong line in the file.
CitiesAmount	if there is not enough data.
FileOpen	if the file cannot be opened.

Returns

Matrix with distances.

5.2 helpers Namespace Reference

Helper functions.

Functions

- template<class T >
  bool contains (const std::vector< T > &vec, const std::pair< int, int > border, const T &value)

  Checks whether the vector contains the given value in the given range.
- template<class T >
  bool contains (const std::vector< T > &vec, const int border, const T &value)

  Checks whether the vector contains the given value in the range from the beginning to the given border.
- template<class T >
  bool contains (const std::vector< T > &vec, const T &value)

  Checks whether the vector contains the given value.

5.2.1 Detailed Description

Helper functions.

5.2.2 Function Documentation

5.2.2.1 contains() [1/3]

```
template<class T >
bool helpers::contains (
            const std::vector< T > & vec,
            const int border,
            const T & value )
```

Checks whether the vector contains the given value in the range from the beginning to the given border.

Template Parameters

T	Generic type parameter.

Parameters

vec	The vector.
border	The border.
value	The value.

Returns

True if the object is in this collection, false if not.

5.2.2.2 contains() [2/3]

```
template<class T >
bool helpers::contains (
            const std::vector< T > & vec,
            const std::pair< int, int > border,
            const T & value )
```

Checks whether the vector contains the given value in the given range.

Template Parameters

T	Generic type parameter.

Parameters

vec	The vector.
border	The range.
value	The value.

Returns

True if the object is in this collection, false if not.

5.2.2.3 contains() [3/3]

```
template<class T >
bool helpers::contains (
            const std::vector< T > & vec,
            const T & value )
```

Checks whether the vector contains the given value.

Template Parameters

T	Generic type parameter.

Parameters

vec	The vector.
value	The value.

Returns

True if the object is in this collection, false if not.

Chapter 6

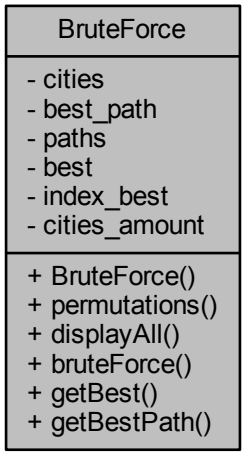
Class Documentation

6.1 BruteForce Class Reference

Brute Force class.

`#include <BruteForce.h>`

Collaboration diagram for BruteForce:



Public Member Functions

- BruteForce (std::shared_ptr< std::vector< City >> &c)

  Constructor.
- void permutations ()

  Generates all possible paths.
- void displayAll (tgui::Canvas::Ptr canvas1, tgui::Canvas::Ptr canvas2)

Displays the first and the best path on the screen.

- void bruteForce (tgui::Canvas::Ptr canvas1, tgui::Canvas::Ptr canvas2)

  Calls functions in order to generate the solution.
- double getBest () const

  Gets the best distance.
- std::shared_ptr< Path > & getBestPath ()

  Gets the best path.

Private Attributes

- std::shared_ptr< std::vector< City > > cities

  The cities.
- std::shared_ptr< Path > best_path

  Pointer to the shortest path.
- std::vector< Path > paths

  Vector with all possible paths.
- double best

  Shortest distance.
- int index_best

  Index of the best path.
- const int cities_amount

  Number of cities.

6.1.1 Detailed Description

Brute Force class.

Class responsible for solving TSP using brute force method.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 BruteForce()

```
BruteForce::BruteForce (
            std::shared_ptr< std::vector< City >> & c )
```

Constructor.

Parameters

c	Pointer to the vector with cities.

6.1.3 Member Function Documentation

6.1.3.1 bruteForce()

```
void BruteForce::bruteForce (
            tgui::Canvas::Ptr canvas1,
            tgui::Canvas::Ptr canvas2 )
```

Calls functions in order to generate the solution.

Parameters

canvas1	The first canvas.
canvas2	The second canvas.

6.1.3.2 displayAll()

```
void BruteForce::displayAll (
            tgui::Canvas::Ptr canvas1,
            tgui::Canvas::Ptr canvas2 )
```

Displays the first and the best path on the screen.

Parameters

canvas1	The first canvas.
canvas2	The second canvas.

6.1.3.3 getBest()

```
double BruteForce::getBest ( ) const
```

Gets the best distance.

Returns

Shortest distance.

6.1.3.4 getBestPath()

```
std::shared_ptr< Path > & BruteForce::getBestPath ( )
```

Gets the best path.

Returns

The best path.

6.1.3.5 permutations()

```
void BruteForce::permutations ( )
```

Generates all possible paths.

6.1.4 Member Data Documentation

6.1.4.1 best

```
double BruteForce::best  [private]
```

Shortest distance.

6.1.4.2 best_path

```
std::shared_ptr<Path> BruteForce::best_path  [private]
```

Pointer to the shortest path.

6.1.4.3 cities

```
std::shared_ptr<std::vector<City> > BruteForce::cities  [private]
```

The cities.

6.1.4.4 cities_amount

```
const int BruteForce::cities_amount  [private]
```

Number of cities.

6.1.4.5 index_best

```
int BruteForce::index_best  [private]
```

Index of the best path.

6.1.4.6 paths

```
std::vector<Path> BruteForce::paths  [private]
```

Vector with all possible paths.

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

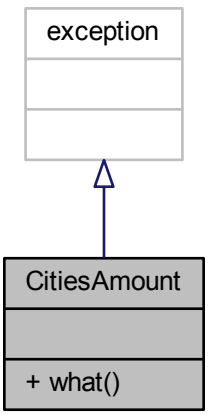
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/BruteForce.h
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/BruteForce.cpp

6.2 CitiesAmount Class Reference

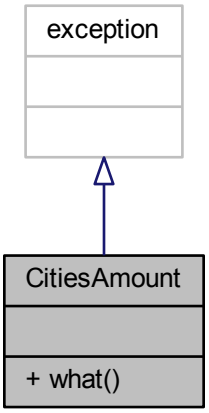
Responsible for throwing a message when there is not enough data for declared number of cities.

```
#include <exceptions.h>
```

Inheritance diagram for CitiesAmount:



Collaboration diagram for CitiesAmount:



Public Member Functions

- const char * what () const override throw ()
  Gets the message.

6.2.1 Detailed Description

Responsible for throwing a message when there is not enough data for declared number of cities.

6.2.2 Member Function Documentation

6.2.2.1 what()

```
const char* CitiesAmount::what ( ) const throw ( )   [inline], [override]
```

Gets the message.

Returns

Null if it fails, else a pointer to a const char.

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

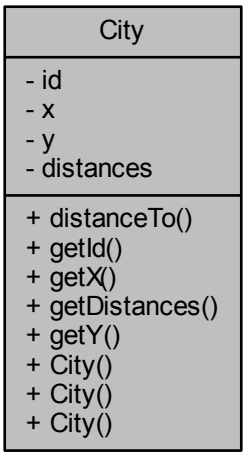
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/exceptions.h

6.3 City Class Reference

Represents the city.

`#include <City.h>`

Collaboration diagram for City:



Public Member Functions

- double distanceTo (City &a)

  Calculates the distance between this and the given city.
- int getId () const

  Gets the identifier.
- double getX () const

  Get x coordinate.
- std::vector< double > & getDistances ()

  Returns the reference to the vector with distances to each city.
- double getY () const

  Get y coordinate.
- City (double _x, double _y, int _id, int n)

  Constructor. Resizes the vector so that it can hold the distances to all cities.
- City (sf::Vector2f coords, int _id, int n)

  Constructor for the circle representation.
- City (std::vector< double > dist, sf::Vector2f coords, int _id, int n)

  Constructor.

Private Attributes

- int id

  The identifier.
- double x

  The x coordinate.
- double y

  The y coordinate.
- std::vector< double > distances

  The distances to other cities.

Friends

- bool operator== (const City &lhs, const City &rhs)

  Equality operator.
- bool operator< (const City &lhs, const City &rhs)

  Less-than comparison operator.

6.3.1 Detailed Description

Represents the city.

6.3.2 Constructor & Destructor Documentation

6.3.2.1 City() [1/3]

```
City::City (
            double _x,
            double _y,
            int _id,
            int n )  [inline]
```

Constructor. Resizes the vector so that it can hold the distances to all cities.

Parameters

↩ _↩ *x*	The x coordinate.
↩ _↩ *y*	The y coordinate.
↩ _↩ *id*	The identifier.
n	The number of cities.

6.3.2.2 City() [2/3]

```
City::City (
            sf::Vector2f coords,
            int _id,
            int n )  [inline]
```

Constructor for the circle representation.

Constructs city using the Vector with x and y coordinates. This constructor is used when the cities lie on the circle.

Parameters

coords	The coordinates.
_id	The identifier.
n	The number of cities.

6.3.2.3 City() [3/3]

```
City::City (
            std::vector< double > dist,
            sf::Vector2f coords,
            int _id,
            int n )
```

Constructor.

Parameters

dist	The distances between cities.
coords	The coordinates.
_id	The identifier.
n	The number of cities.

6.3.3 Member Function Documentation

6.3.3.1 distanceTo()

```
double City::distanceTo (
            City & a )
```

Calculates the distance between this and the given city.

Parameters

a	A City to process.

Returns

The distance between cities.

6.3.3.2 getDistances()

```
std::vector< double > & City::getDistances ( )
```

Returns the reference to the vector with distances to each city.

Returns

Vector with distances.

6.3.3.3 getId()

```
int City::getId ( ) const
```

Gets the identifier.

Returns

The identifier.

6.3.3.4 getX()

```
double City::getX ( ) const
```

Get x coordinate.

Returns

The x coordinate.

6.3.3.5 getY()

```
double City::getY ( ) const
```

Get y coordinate.

Returns

The y coordinate.

6.3.4 Friends And Related Function Documentation

6.3.4.1 operator<

```
bool operator< (
            const City & lhs,
            const City & rhs )  [friend]
```

Less-than comparison operator.

Parameters

lhs	The first city to compare.
rhs	The second city to compare.

Returns

True if the first parameter is less than the second.

6.3.4.2 operator==

```
bool operator== (
            const City & lhs,
            const City & rhs )  [friend]
```

Equality operator.

Parameters

lhs	The first city to compare.
rhs	The second city to compare.

Returns

True if the parameters are considered equivalent.

6.3.5 Member Data Documentation

6.3.5.1 distances

```
std::vector<double> City::distances  [private]
```

The distances to other cities.

6.3.5.2 id

```
int City::id  [private]
```

The identifier.

6.3.5.3 x

```
double City::x  [private]
```

The x coordinate.

6.3.5.4 y

```
double City::y  [private]
```

The y coordinate.

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

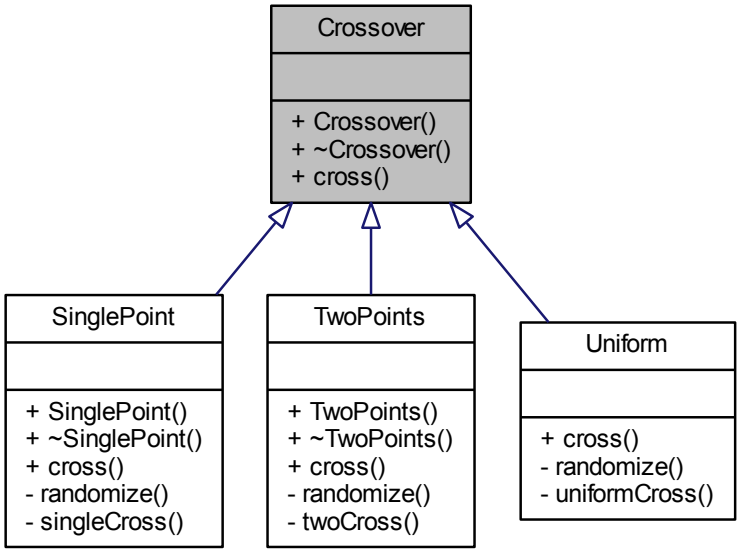
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/City.h
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/City.cpp

6.4 Crossover Class Reference

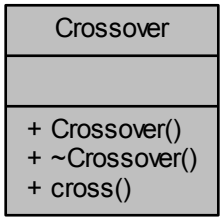
Abstract class for crossover strategy interface.

`#include <Crossover.h>`

Inheritance diagram for Crossover:



Collaboration diagram for Crossover:



Public Member Functions

- Crossover ()=default
- virtual ~Crossover ()=default
- virtual std::pair< Path, Path > cross (std::pair< Path, Path > &parents_pair)

  Crosses a given pair of parents and returns their children. The paths from parents are mixed, the result of that operation creates the path for the children.

6.4.1 Detailed Description

Abstract class for crossover strategy interface.

6.4.2 Constructor & Destructor Documentation

6.4.2.1 Crossover()

```
Crossover::Crossover ( )  [default]
```

6.4.2.2 ~Crossover()

```
virtual Crossover::~Crossover ( )  [virtual], [default]
```

6.4.3 Member Function Documentation

6.4.3.1 cross()

```
std::pair< Path, Path > Crossover::cross (
            std::pair< Path, Path > & parents_pair )  [virtual]
```

Crosses a given pair of parents and returns their children. The paths from parents are mixed, the result of that operation creates the path for the children.

Parameters

parents_pair	Pair of two individuals(parents).

Returns

std::pair containing new individuals (children) resulting from their crossing;

Reimplemented in SinglePoint, TwoPoints, and Uniform.

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

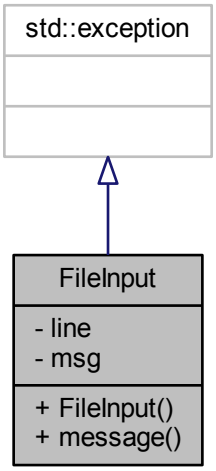
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Crossover.h
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Crossover.cpp

6.5 FileInput Class Reference

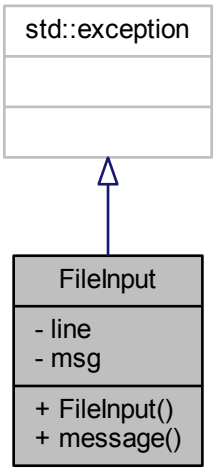
Responsible for throwing a message when the data in the file is invalid.

`#include <exceptions.h>`

Inheritance diagram for FileInput:



Collaboration diagram for FileInput:



Public Member Functions

- FileInput (int i)

  Constructor.
- const std::string & message ()

  Gets the message.

Private Attributes

- int line

  Number of the line.
- std::string msg

  The message.

6.5.1 Detailed Description

Responsible for throwing a message when the data in the file is invalid.

6.5.2 Constructor & Destructor Documentation

6.5.2.1 FileInput()

```
FileInput::FileInput (
            int i )  [inline]
```

Constructor.

Parameters

i	Number of the line containing incorrect input.

6.5.3 Member Function Documentation

6.5.3.1 message()

```
const std::string& FileInput::message ( )  [inline]
```

Gets the message.

Returns

A reference to a const std::string.

6.5.4 Member Data Documentation

6.5.4.1 line

```
int FileInput::line  [private]
```

Number of the line.

6.5.4.2 msg

```
std::string FileInput::msg  [private]
```

The message.

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

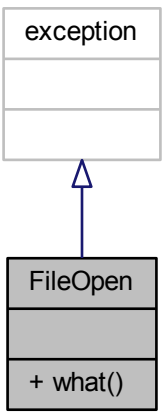
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/exceptions.h

6.6 FileOpen Class Reference

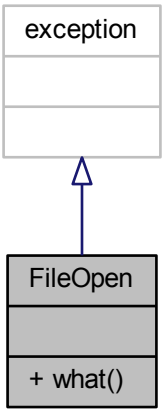
Responsible for throwing a message when the file cannot be opened or created.

```
#include <exceptions.h>
```

Inheritance diagram for FileOpen:



Collaboration diagram for FileOpen:



Public Member Functions

- const char * what () const override throw ()
  Gets the message.

6.6.1 Detailed Description

Responsible for throwing a message when the file cannot be opened or created.

6.6.2 Member Function Documentation

6.6.2.1 what()

```
const char* FileOpen::what ( ) const throw ( )   [inline], [override]
```

Gets the message.

Returns

Null if it fails, else a pointer to a const char.

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

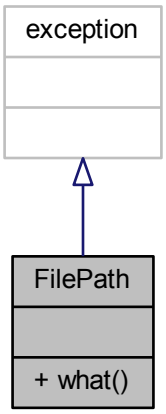
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/exceptions.h

6.7 FilePath Class Reference

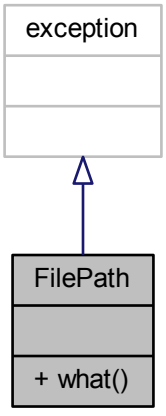
Responsible for throwing a message when the path of the file is invalid.

`#include <exceptions.h>`

Inheritance diagram for FilePath:



Collaboration diagram for FilePath:



Public Member Functions

- const char * what () const override throw ()

  Gets the message.

6.7.1 Detailed Description

Responsible for throwing a message when the path of the file is invalid.

6.7.2 Member Function Documentation

6.7.2.1 what()

```
const char* FilePath::what ( ) const throw ( )   [inline], [override]
```

Gets the message.

Returns

Null if it fails, else a pointer to a const char.

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

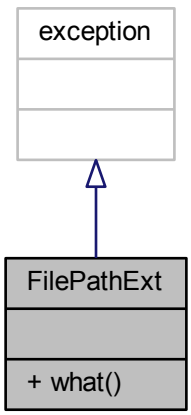
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/exceptions.h

6.8 FilePathExt Class Reference

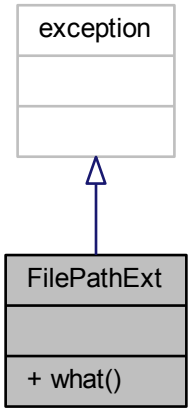
Responsible for throwing a message when the path or extension of the file is invalid.

```
#include <exceptions.h>
```

Inheritance diagram for FilePathExt:



Collaboration diagram for FilePathExt:



Public Member Functions

- const char * what () const override throw ()
  Gets the message.

6.8.1 Detailed Description

Responsible for throwing a message when the path or extension of the file is invalid.

6.8.2 Member Function Documentation

6.8.2.1 what()

```
const char* FilePathExt::what ( ) const throw ( )   [inline], [override]
```

Gets the message.

Returns

Null if it fails, else a pointer to a const char.

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

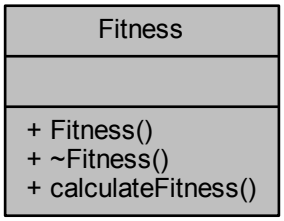
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/exceptions.h

6.9 Fitness Class Reference

Fitness class.

`#include <Fitness.h>`

Collaboration diagram for Fitness:



Public Member Functions

- Fitness ()=default

  Default constructor.
- ~Fitness ()=default

  Destructor.
- void calculateFitness (std::unique_ptr< Generation > &generation)

  Calculates the fitness of each individual in the generation.

6.9.1 Detailed Description

Fitness class.

6.9.2 Constructor & Destructor Documentation

6.9.2.1 Fitness()

`Fitness::Fitness ( ) [default]`

Default constructor.

6.9.2.2 ~Fitness()

```
Fitness::~Fitness ( )  [default]
```

Destructor.

6.9.3 Member Function Documentation

6.9.3.1 calculateFitness()

```
void Fitness::calculateFitness (
            std::unique_ptr< Generation > & generation )
```

Calculates the fitness of each individual in the generation.

Sorts the generation by fitness in descending order.

Parameters

generation	The generation.

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

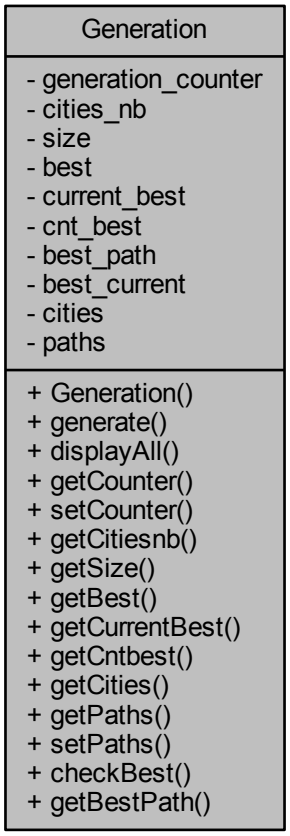
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Fitness.h
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Fitness.cpp

6.10 Generation Class Reference

Represents the generation.

```
#include <Generation.h>
```

Collaboration diagram for Generation:



Public Member Functions

- Generation (int cs, int s, std::shared_ptr< std::vector< City >> &c)

  Constructor.
- void generate ()

  Generates the initial population.
- void displayAll (tgui::Canvas::Ptr canvas1, tgui::Canvas::Ptr canvas2)

  Displays the shortest path from the current generation and the shortest ever.
- int getCounter () const

  Gets the number of the current generation.
- void setCounter (int counter)

  Sets a counter.
- const int getCitiesnb () const

  Gets the number of cities.
- const int getSize () const

  Gets the size of the generation.
- double getBest () const

  Gets the shortest distance ever.

- double getCurrentBest () const

  Gets the shortest distance from the current generation.
- int getCntbest () const

  Gets the generation number of the best ever solution.
- std::shared_ptr< std::vector< City > > getCities () const

  Gets pointer to the vector with cities.
- std::vector< Path > & getPaths ()

  Gets the vector with paths.
- void setPaths (std::vector< Path > paths)

  Sets the paths.
- void checkBest ()

  Checks which path from the generation is the best and if it is better than the best ever solution.
- std::shared_ptr< Path > & getBestPath ()

  Gets pointer to the best ever path.

Private Attributes

- int generation_counter

  Number of the current generation.
- const int cities_nb

  Number of cities.
- const int size

  Size of the generation.
- double best

  The shortest distance ever.
- double current_best

  The shortest distance from the current generation.
- int cnt_best

  Generation number of the best ever solution.
- std::shared_ptr< Path > best_path

  Pointer to the best ever path.
- std::shared_ptr< Path > best_current

  Pointer to the best path from the current generation.
- std::shared_ptr< std::vector< City > > cities

  Pointer to the vector with cities.
- std::vector< Path > paths

  Vector with all paths from the current generation.

6.10.1 Detailed Description

Represents the generation.

Contains all individuals from the generation.

6.10.2 Constructor & Destructor Documentation

6.10.2.1 Generation()

```
Generation::Generation (
            int cs,
            int s,
            std::shared_ptr< std::vector< City >> & c )
```

Constructor.

Parameters

cs	Number of cities.
s	Size of the generation.
c	Pointer to the vector with cities.

6.10.3 Member Function Documentation

6.10.3.1 checkBest()

```
void Generation::checkBest ( )
```

Checks which path from the generation is the best and if it is better than the best ever solution.

6.10.3.2 displayAll()

```
void Generation::displayAll (
            tgui::Canvas::Ptr canvas1,
            tgui::Canvas::Ptr canvas2 )
```

Displays the shortest path from the current generation and the shortest ever.

Parameters

canvas1	The first canvas for current best.
canvas2	The second canvas for best ever.

6.10.3.3 generate()

```
void Generation::generate ( )
```

Generates the initial population.

Paths are generated randomly.

6.10.3.4 getBest()

```
double Generation::getBest ( ) const
```

Gets the shortest distance ever.

Returns

The shortest distance.

6.10.3.5 getBestPath()

```
std::shared_ptr< Path > & Generation::getBestPath ( )
```

Gets pointer to the best ever path.

Returns

The best path.

6.10.3.6 getCities()

```
std::shared_ptr< std::vector< City > > Generation::getCities ( ) const
```

Gets pointer to the vector with cities.

Returns

The cities.

6.10.3.7 getCitiesnb()

```
const int Generation::getCitiesnb ( ) const
```

Gets the number of cities.

Returns

Number of cities.

6.10.3.8 getCntbest()

```
int Generation::getCntbest ( ) const
```

Gets the generation number of the best ever solution.

Returns

The cntbest.

6.10.3.9 getCounter()

```
int Generation::getCounter ( ) const
```

Gets the number of the current generation.

Returns

Number of the current generation.

6.10.3.10 getCurrentBest()

```
double Generation::getCurrentBest ( ) const
```

Gets the shortest distance from the current generation.

Returns

The current best.

6.10.3.11 getPaths()

```
std::vector< Path > & Generation::getPaths ( )
```

Gets the vector with paths.

Returns

Reference to the vector.

6.10.3.12 getSize()

```
const int Generation::getSize ( ) const
```

Gets the size of the generation.

Returns

The size.

6.10.3.13 setCounter()

```
void Generation::setCounter (
            int counter )
```

Sets a counter.

Parameters

counter	The counter.

6.10.3.14 setPaths()

```
void Generation::setPaths (
            std::vector< Path > paths )
```

Sets the paths.

Parameters

paths	Vector with paths.

6.10.4 Member Data Documentation

6.10.4.1 best

```
double Generation::best  [private]
```

The shortest distance ever.

6.10.4.2 best_current

```
std::shared_ptr<Path> Generation::best_current  [private]
```

Pointer to the best path from the current generation.

6.10.4.3 best_path

```
std::shared_ptr<Path> Generation::best_path  [private]
```

Pointer to the best ever path.

6.10.4.4 cities

```
std::shared_ptr<std::vector<City> > Generation::cities  [private]
```

Pointer to the vector with cities.

6.10.4.5 cities_nb

```
const int Generation::cities_nb  [private]
```

Number of cities.

6.10.4.6 cnt_best

```
int Generation::cnt_best  [private]
```

Generation number of the best ever solution.

6.10.4.7 current_best

```
double Generation::current_best  [private]
```

The shortest distance from the current generation.

6.10.4.8 generation_counter

```
int Generation::generation_counter  [private]
```

Number of the current generation.

6.10.4.9 paths

```
std::vector<Path> Generation::paths  [private]
```

Vector with all paths from the current generation.

6.10.4.10 size

```
const int Generation::size  [private]
```

Size of the generation.

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

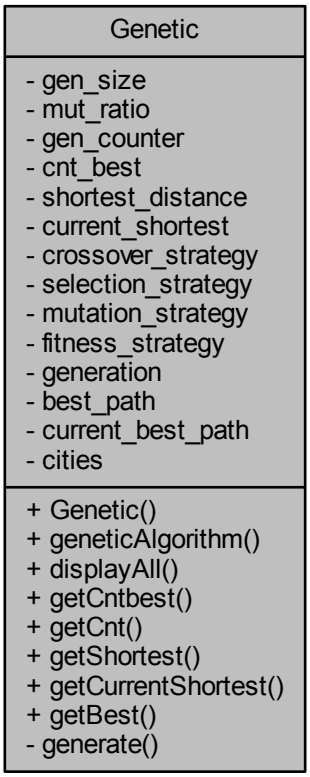
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Generation.h
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Generation.cpp

6.11 Genetic Class Reference

Declares the Genetic Algorithm class.

```
#include <Genetic.h>
```

Collaboration diagram for Genetic:



Public Member Functions

- Genetic (int c, int s, double r, std::shared_ptr< std::vector< City >> &cit, int sel_strat, int cross_strat)

  Constructor.
- void geneticAlgorithm ()

  Performs genetic algorithm on the generation.
- void displayAll (tgui::Canvas::Ptr canvas1, tgui::Canvas::Ptr canvas2)

  Calls function from the generation class displaying best current and best ever solutions.
- int getCntbest () const

  Gets the number of the best generation.
- int getCnt () const

  Gets the number of the current generation.
- double getShortest () const

  Gets the shortest distance.
- double getCurrentShortest () const

  Gets the shortest distance from the current generation.
- std::shared_ptr< Path > & getBest ()

  Gets pointer to the best path.

Private Member Functions

- void generate ()

  Calls function from the generation object which generates the initial population.

Private Attributes

- const int gen_size

  Size of the generation.
- const double mut_ratio

  Mutation ratio.
- int gen_counter

  Number of the current generation.
- int cnt_best

  Number of the best generation.
- double shortest_distance

  Shortest distance ever.
- double current_shortest

  Shortest distance from the current generation.
- std::unique_ptr< Crossover > crossover_strategy

  Pointer to the crossover strategy interface.
- std::unique_ptr< Selection > selection_strategy

  Pointer to the selection strategy interface.
- std::unique_ptr< Mutation > mutation_strategy

  Pointer to the mutation object.
- std::unique_ptr< Fitness > fitness_strategy

  Pointer to the fitness object.
- std::unique_ptr< Generation > generation

  Pointer to the generation object.
- std::shared_ptr< Path > best_path

  Pointer to the best ever path.
- std::shared_ptr< Path > current_best_path

  Pointer to the best path from the current generation.
- std::shared_ptr< std::vector< City > > cities

  Pointer to the vector with cities.

6.11.1 Detailed Description

Declares the Genetic Algorithm class.

6.11.2 Constructor & Destructor Documentation

6.11.2.1 Genetic()

```
Genetic::Genetic (
            int c,
            int s,
            double r,
            std::shared_ptr< std::vector< City >> & cit,
            int sel_strat,
            int cross_strat )
```

Constructor.

Creates unique pointers that manages objects used in the algorithm. Crossover and selection are created depending on the users choice. Switch statements create and assign to interfaces pointers that manages objects that implement concrete strategies.

Parameters

c	Number of cities.
s	Size of the generation.
r	Mutation ratio.
cit	Pointer to the vector with cities.
sel_strat	Selection strategy chosen by the user.
cross_strat	Crossover strategy chosen by the user.

6.11.3 Member Function Documentation

6.11.3.1 displayAll()

```
void Genetic::displayAll (
            tgui::Canvas::Ptr canvas1,
            tgui::Canvas::Ptr canvas2 )
```

Calls function from the generation class displaying best current and best ever solutions.

Parameters

canvas1	The first canvas for current best.
canvas2	The second canvas for best ever.

6.11.3.2 generate()

```
void Genetic::generate ( )  [private]
```

Calls function from the generation object which generates the initial population.

6.11.3.3 geneticAlgorithm()

```
void Genetic::geneticAlgorithm ( )
```

Performs genetic algorithm on the generation.

Calls appropriate functions from the other objects. If this is the first generation, it generates the chromosomes and calculate their fitnesses, otherwise calculate fitness, performs selection, crossover and mutation. Sets pointers to the best paths.

6.11.3.4 getBest()

```
std::shared_ptr< Path > & Genetic::getBest ( )
```

Gets pointer to the best path.

Returns

Pointer to the best ever path.

6.11.3.5 getCnt()

```
int Genetic::getCnt ( ) const
```

Gets the number of the current generation.

Returns

Current generation number.

6.11.3.6 getCntbest()

```
int Genetic::getCntbest ( ) const
```

Gets the number of the best generation.

Returns

The number of the best generation.

6.11.3.7 getCurrentShortest()

```
double Genetic::getCurrentShortest ( ) const
```

Gets the shortest distance from the current generation.

Returns

The shortest distance from the current generation.

6.11.3.8 getShortest()

```
double Genetic::getShortest ( ) const
```

Gets the shortest distance.

Returns

The shortest distance ever.

6.11.4 Member Data Documentation

6.11.4.1 best_path

```
std::shared_ptr<Path> Genetic::best_path  [private]
```

Pointer to the best ever path.

6.11.4.2 cities

```
std::shared_ptr<std::vector<City> > Genetic::cities  [private]
```

Pointer to the vector with cities.

6.11.4.3 cnt_best

```
int Genetic::cnt_best  [private]
```

Number of the best generation.

6.11.4.4 crossover_strategy

```
std::unique_ptr<Crossover> Genetic::crossover_strategy  [private]
```

Pointer to the crossover strategy interface.

6.11.4.5 current_best_path

```
std::shared_ptr<Path> Genetic::current_best_path  [private]
```

Pointer to the best path from the current generation.

6.11.4.6 current_shortest

```
double Genetic::current_shortest  [private]
```

Shortest distance from the current generation.

6.11.4.7 fitness_strategy

```
std::unique_ptr<Fitness> Genetic::fitness_strategy  [private]
```

Pointer to the fitness object.

6.11.4.8 gen_counter

```
int Genetic::gen_counter  [private]
```

Number of the current generation.

6.11.4.9 gen_size

```
const int Genetic::gen_size  [private]
```

Size of the generation.

6.11.4.10 generation

```
std::unique_ptr<Generation> Genetic::generation  [private]
```

Pointer to the generation object.

6.11.4.11 mut_ratio

```
const double Genetic::mut_ratio  [private]
```

Mutation ratio.

6.11.4.12 mutation_strategy

```
std::unique_ptr<Mutation> Genetic::mutation_strategy  [private]
```

Pointer to the mutation object.

6.11.4.13 selection_strategy

```
std::unique_ptr<Selection> Genetic::selection_strategy  [private]
```

Pointer to the selection strategy interface.

6.11.4.14 shortest_distance

```
double Genetic::shortest_distance  [private]
```

Shortest distance ever.

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

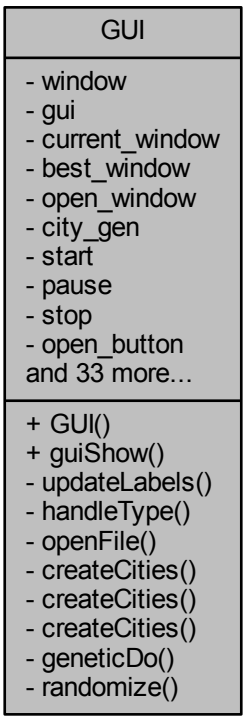
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Genetic.h
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Genetic.cpp

6.12 GUI Class Reference

Graphical User Interface.

`#include <GUI.h>`

Collaboration diagram for GUI:



Public Member Functions

- GUI ()

  GUI constructor.

- void guiShow ()

  Shows The gui.

Private Member Functions

- void updateLabels ()

  Updates the labels displaying values of the sliders.

- void handleType ()

  Handles the type of the solution.

- void openFile ()

  Opens and reads the input file.
- void createCities ()

  Creates cities with random x and y coordinates.
- void createCities (std::vector< std::vector< double >> input)

  Creates the cities from the file input using vectors containing distances between the cities.
- void createCities (std::vector< sf::Vector2f > input)

  Creates the cities from the file input using vector of x and y coordinates of each city.
- void geneticDo ()

  Performs genetic algorithm.
- double randomize ()

  Generates random number in a range.

Private Attributes

- sf::RenderWindow window

  The render window.
- tgui::Gui gui

  The graphical user interface.
- tgui::ChildWindow::Ptr current_window

  Current path window.
- tgui::ChildWindow::Ptr best_window

  Best path window.
- tgui::ChildWindow::Ptr open_window

  Input file window.
- tgui::Button::Ptr city_gen

  Generate Cities button.
- tgui::Button::Ptr start

  Start Button.
- tgui::Button::Ptr pause

  Pause Button.
- tgui::Button::Ptr stop

  Stop button.
- tgui::Button::Ptr open_button

  Open file button.
- tgui::Button::Ptr input_button

  Input file button.
- tgui::Button::Ptr output_button

  Output file button.
- tgui::Canvas::Ptr current_canvas

  The current canvas.
- tgui::Canvas::Ptr best_canvas

  The best canvas.
- tgui::Slider::Ptr gen_size

  Generation size slider.
- tgui::Slider::Ptr mutation

  Mutation slider.
- tgui::Slider::Ptr nb

  Number of cities slider.
- tgui::Slider::Ptr type

Solution type slider.

- tgui::Slider::Ptr selection

  Selection method slider.
- tgui::Slider::Ptr crossover

  Crossover method slider.
- tgui::Label::Ptr type_text

  The type text.
- tgui::Label::Ptr gen_text

  The generate text.
- tgui::Label::Ptr mut_text

  The mutation text.
- tgui::Label::Ptr nb_text

  The number text.
- tgui::Label::Ptr shortest

  The shortest text.
- tgui::Label::Ptr best

  The best.
- tgui::Label::Ptr current

  The current.
- tgui::Label::Ptr current_best

  The current best.
- tgui::Label::Ptr selection_text

  The selection text.
- tgui::Label::Ptr crossover_text

  The crossover text.
- tgui::Label::Ptr output_text

  The output text.
- tgui::EditBox::Ptr input_path

  Full pathname of the input file.
- tgui::EditBox::Ptr output_path

  Full pathname of the output file.
- tgui::TextBox::Ptr file_text

  The file text.
- tgui::MessageBox::Ptr output_error

  Messagebox for output file.
- tgui::MessageBox::Ptr input_error

  Messagebox for input file.
- std::unique_ptr< BruteForce > brute

  Pointer to the brute force class.
- std::unique_ptr< Genetic > genetic

  Pointer to the genetic algorithm class.
- std::shared_ptr< std::vector< City > > cities

  Pointer to the vector with cities.
- bool generated

  True if cities were generated.
- bool performed

  True if start button was clicked.
- bool first

  True if it will be the first generation of the algorithm.
- bool paused

  True if paused.

6.12.1 Detailed Description

Graphical User Interface.

Class responsible for GUI. GUI is created using TGUI library. Contains pointers to BruteForce and Genetic classes. Also creates cities that will be used in TSP.

6.12.2 Constructor & Destructor Documentation

6.12.2.1 GUI()

```
GUI::GUI ( )
```

GUI constructor.

Loads TGUI widgets from txt file. Handles button signals.

6.12.3 Member Function Documentation

6.12.3.1 createCities() [1/3]

```
void GUI::createCities ( )  [private]
```

Creates cities with random x and y coordinates.

6.12.3.2 createCities() [2/3]

```
void GUI::createCities (
            std::vector< sf::Vector2f > input )  [private]
```

Creates the cities from the file input using vector of x and y coordinates of each city.

Parameters

input	x and y coordinates of each city.

6.12.3.3 createCities() [3/3]

```
void GUI::createCities (
            std::vector< std::vector< double >> input )  [private]
```

Creates the cities from the file input using vectors containing distances between the cities.

Parameters

input	Distances between the cities.

6.12.3.4 geneticDo()

```
void GUI::geneticDo ( )  [private]
```

Performs genetic algorithm.

Calls functions form the genetic class and updates labels connected with the algorithm.

6.12.3.5 guiShow()

```
void GUI::guiShow ( )
```

Shows The gui.

Contains main GUI loop. Responsible for detecting events, passing them to the widgets and displaying the interface.

6.12.3.6 handleType()

```
void GUI::handleType ( )  [private]
```

Handles the type of the solution.

Changes visibility of the GUI elements depending on the type of the solution.

6.12.3.7 openFile()

```
void GUI::openFile ( )  [private]
```

Opens and reads the input file.

Uses functions from FileIO namespace.

Exceptions

InputType	Thrown if the input data type description in the file is incorrect.

6.12.3.8 randomize()

```
double GUI::randomize ( )  [private]
```

Generates random number in a range.

Used to generate coordinates of the cities.

Returns

Random number.

6.12.3.9 updateLabels()

```
void GUI::updateLabels ( )  [private]
```

Updates the labels displaying values of the sliders.

Changes the texts in labels to show values of the sliders(number of cities etc.).

6.12.4 Member Data Documentation

6.12.4.1 best

```
tgui::Label::Ptr GUI::best  [private]
```

The best.

6.12.4.2 best_canvas

```
tgui::Canvas::Ptr GUI::best_canvas  [private]
```

The best canvas.

6.12.4.3 best_window

```
tgui::ChildWindow::Ptr GUI::best_window  [private]
```

Best path window.

6.12.4.4 brute

```
std::unique_ptr<BruteForce> GUI::brute  [private]
```

Pointer to the brute force class.

6.12.4.5 cities

```
std::shared_ptr<std::vector<City> > GUI::cities  [private]
```

Pointer to the vector with cities.

6.12.4.6 city_gen

```
tgui::Button::Ptr GUI::city_gen  [private]
```

Generate Cities button.

6.12.4.7 crossover

```
tgui::Slider::Ptr GUI::crossover  [private]
```

Crossover method slider.

6.12.4.8 crossover_text

```
tgui::Label::Ptr GUI::crossover_text  [private]
```

The crossover text.

6.12.4.9 current

```
tgui::Label::Ptr GUI::current  [private]
```

The current.

6.12.4.10 current_best

```
tgui::Label::Ptr GUI::current_best  [private]
```

The current best.

6.12.4.11 current_canvas

```
tgui::Canvas::Ptr GUI::current_canvas  [private]
```

The current canvas.

6.12.4.12 current_window

```
tgui::ChildWindow::Ptr GUI::current_window  [private]
```

Current path window.

6.12.4.13 file_text

```
tgui::TextBox::Ptr GUI::file_text  [private]
```

The file text.

6.12.4.14 first

```
bool GUI::first  [private]
```

True if it will be the first generation of the algorithm.

6.12.4.15 gen_size

```
tgui::Slider::Ptr GUI::gen_size  [private]
```

Generation size slider.

6.12.4.16 gen_text

```
tgui::Label::Ptr GUI::gen_text  [private]
```

The generate text.

6.12.4.17 generated

```
bool GUI::generated  [private]
```

True if cities were generated.

6.12.4.18 genetic

```
std::unique_ptr<Genetic> GUI::genetic  [private]
```

Pointer to the genetic algorithm class.

6.12.4.19 gui

```
tgui::Gui GUI::gui  [private]
```

The graphical user interface.

6.12.4.20 input_button

```
tgui::Button::Ptr GUI::input_button  [private]
```

Input file button.

6.12.4.21 input_error

```
tgui::MessageBox::Ptr GUI::input_error  [private]
```

Messagebox for input file.

6.12.4.22 input_path

```
tgui::EditBox::Ptr GUI::input_path  [private]
```

Full pathname of the input file.

6.12.4.23 mut_text

```
tgui::Label::Ptr GUI::mut_text  [private]
```

The mutation text.

6.12.4.24 mutation

```
tgui::Slider::Ptr GUI::mutation  [private]
```

Mutation slider.

6.12.4.25 nb

```
tgui::Slider::Ptr GUI::nb  [private]
```

Number of cities slider.

6.12.4.26 nb_text

```
tgui::Label::Ptr GUI::nb_text  [private]
```

The number text.

6.12.4.27 open_button

```
tgui::Button::Ptr GUI::open_button  [private]
```

Open file button.

6.12.4.28 open_window

```
tgui::ChildWindow::Ptr GUI::open_window  [private]
```

Input file window.

6.12.4.29 output_button

```
tgui::Button::Ptr GUI::output_button  [private]
```

Output file button.

6.12.4.30 output_error

```
tgui::MessageBox::Ptr GUI::output_error  [private]
```

Messagebox for output file.

6.12.4.31 output_path

```
tgui::EditBox::Ptr GUI::output_path  [private]
```

Full pathname of the output file.

6.12.4.32 output_text

```
tgui::Label::Ptr GUI::output_text  [private]
```

The output text.

6.12.4.33 pause

```
tgui::Button::Ptr GUI::pause  [private]
```

Pause Button.

6.12.4.34 paused

```
bool GUI::paused  [private]
```

True if paused.

6.12.4.35 performed

```
bool GUI::performed  [private]
```

True if start button was clicked.

6.12.4.36 selection

```
tgui::Slider::Ptr GUI::selection  [private]
```

Selection method slider.

6.12.4.37 selection_text

```
tgui::Label::Ptr GUI::selection_text  [private]
```

The selection text.

6.12.4.38 shortest

```
tgui::Label::Ptr GUI::shortest  [private]
```

The shortest text.

6.12.4.39 start

```
tgui::Button::Ptr GUI::start  [private]
```

Start Button.

6.12.4.40 stop

```
tgui::Button::Ptr GUI::stop  [private]
```

Stop button.

6.12.4.41 type

```
tgui::Slider::Ptr GUI::type  [private]
```

Solution type slider.

Slider values: 0 - brute force. 1 - genetic algorithm.

6.12.4.42 type_text

```
tgui::Label::Ptr GUI::type_text  [private]
```

The type text.

6.12.4.43 window

```
sf::RenderWindow GUI::window  [private]
```

The render window.

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

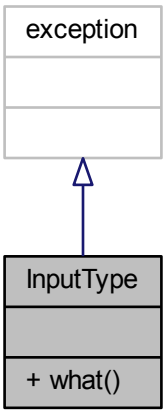
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/GUI.h
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/GUI.cpp

6.13 InputType Class Reference

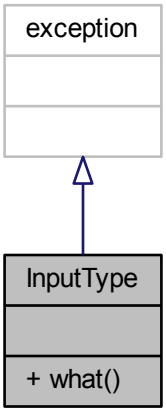
Responsible for throwing a message when the line informing about the input type of the file is incorrect.

`#include <exceptions.h>`

Inheritance diagram for InputType:



Collaboration diagram for InputType:



Public Member Functions

- const char * what () const override throw ()

  Gets the message.

6.13.1 Detailed Description

Responsible for throwing a message when the line informing about the input type of the file is incorrect.

6.13.2 Member Function Documentation

6.13.2.1 what()

```
const char* InputType::what ( ) const throw ( )   [inline], [override]
```

Gets the message.

Returns

Null if it fails, else a pointer to a const char.

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

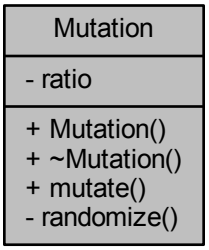
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/exceptions.h

6.14 Mutation Class Reference

Mutation class.

```
#include <Mutation.h>
```

Collaboration diagram for Mutation:



Public Member Functions

- Mutation (double r)

  Constructor.
- ~Mutation ()

  Destructor.
- void mutate (std::unique_ptr< Generation > &generation)

  Performs mutation on the generation.

Private Member Functions

- double randomize (int i, int j)

  Returns random number generated in a given range.

Private Attributes

- double ratio

  Mutation probability.

6.14.1 Detailed Description

Mutation class.

Responsible for mutation.

6.14.2 Constructor & Destructor Documentation

6.14.2.1 Mutation()

```
Mutation::Mutation (
            double r )
```

Constructor.

Parameters

r	Mutation ratio.

6.14.2.2 ~Mutation()

```
Mutation::~Mutation ( )
```

Destructor.

6.14.3 Member Function Documentation

6.14.3.1 mutate()

```
void Mutation::mutate (
            std::unique_ptr< Generation > & generation )
```

Performs mutation on the generation.

Picks the individuals from the generation and randomly swaps cities in its path. Individuals are picked with given probability.

Parameters

generation	The generation.

6.14.3.2 randomize()

```
double Mutation::randomize (
            int i,
            int j )  [private]
```

Returns random number generated in a given range.

Parameters

i	The beginning of the range.
j	The end of the range.

Returns

Random Number.

6.14.4 Member Data Documentation

6.14.4.1 ratio

```
double Mutation::ratio  [private]
```

Mutation probability.

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

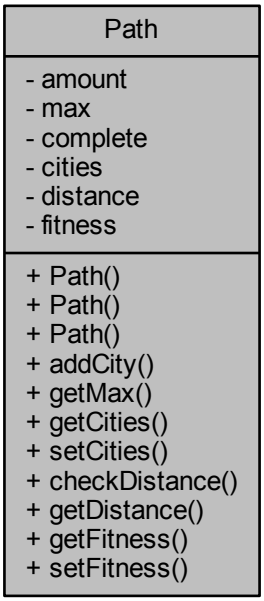
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Mutation.h
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Mutation.cpp

6.15 Path Class Reference

Represents the path.

`#include <Path.h>`

Collaboration diagram for Path:



Public Member Functions

- Path ()=default

  Default constructor.
- Path (int m)

  Constructor.
- Path (const Path &p)

  Copy constructor.
- void addCity (City city)

  Adds a city to the path. When the path is full (all cities have been added) calculates the total distance of the path.
- int getMax () const

  Gets the maximum number of Cities.
- std::vector< City > & getCities ()

  Gets the cities.

- void setCities (std::vector< City > cities)

  Sets the cities.
- void checkDistance ()

  Calculates the distance of the path.
- double getDistance () const

  Gets the distance.
- double getFitness () const

  Gets the fitness.
- void setFitness (double fitness)

  Sets the fitness.

Private Attributes

- int amount

  Current amount of the cities.
- int max

  Maximum number of the cities.
- bool complete

  True if path is full.
- std::vector< City > cities

  The cities.
- double distance

  The distance.
- double fitness

  The fitness.

Friends

- bool operator< (const Path &lhs, const Path &rhs)

  Less-than comparison operator.
- bool operator<= (const Path &lhs, const Path &rhs)

  Less-than-or-equal comparison operator.
- bool operator> (const Path &lhs, const Path &rhs)

  Greater-than comparison operator.
- bool operator>= (const Path &lhs, const Path &rhs)

  Greater-than-or-equal comparison operator.

6.15.1 Detailed Description

Represents the path.

6.15.2 Constructor & Destructor Documentation

6.15.2.1 Path() [1/3]

```
Path::Path ( )  [default]
```

Default constructor.

6.15.2.2 Path() [2/3]

```
Path::Path (
            int m )  [inline]
```

Constructor.

Parameters

m	Number of cities.

6.15.2.3 Path() [3/3]

```
Path::Path (
            const Path & p )
```

Copy constructor.

Parameters

p	A Path to process.

6.15.3 Member Function Documentation

6.15.3.1 addCity()

```
void Path::addCity (
            City city )
```

Adds a city to the path. When the path is full (all cities have been added) calculates the total distance of the path.

Parameters

city	The city to add.

6.15.3.2 checkDistance()

```
void Path::checkDistance ( )
```

Calculates the distance of the path.

6.15.3.3 getCities()

```
std::vector< City > & Path::getCities ( )
```

Gets the cities.

Returns

The reference to the cities vector.

6.15.3.4 getDistance()

```
double Path::getDistance ( ) const
```

Gets the distance.

Returns

The distance.

6.15.3.5 getFitness()

```
double Path::getFitness ( ) const
```

Gets the fitness.

Returns

The fitness.

6.15.3.6 getMax()

```
int Path::getMax ( ) const
```

Gets the maximum number of Cities.

Returns

Maximum number of Cities.

6.15.3.7 setCities()

```
void Path::setCities (
            std::vector< City > cities )
```

Sets the cities.

Parameters

cities	Vector with cities.

6.15.3.8 setFitness()

```
void Path::setFitness (
            double fitness )
```

Sets the fitness.

Parameters

fitness	The fitness.

6.15.4 Friends And Related Function Documentation

6.15.4.1 operator<

```
bool operator< (
            const Path & lhs,
            const Path & rhs )  [friend]
```

Less-than comparison operator.

Parameters

lhs	The first instance to compare.
rhs	The second instance to compare.

Returns

This is true if the second path is more fit than the first.

6.15.4.2 operator<=

```
bool operator<= (
            const Path & lhs,
            const Path & rhs )  [friend]
```

Less-than-or-equal comparison operator.

Parameters

lhs	The first instance to compare.
rhs	The second instance to compare.

Returns

True if the first path is worse than or equal to the second.

6.15.4.3 operator>

```
bool operator> (
            const Path & lhs,
            const Path & rhs )  [friend]
```

Greater-than comparison operator.

Parameters

lhs	The first instance to compare.
rhs	The second instance to compare.

Returns

True if the first path is better than the second.

6.15.4.4 operator>=

```
bool operator>= (
            const Path & lhs,
            const Path & rhs )  [friend]
```

Greater-than-or-equal comparison operator.

Parameters

lhs	The first instance to compare.
rhs	The second instance to compare.

Returns

True if the first path is better or equal to the second.

6.15.5 Member Data Documentation

6.15.5.1 amount

```
int Path::amount  [private]
```

Current amount of the cities.

6.15.5.2 cities

```
std::vector<City> Path::cities  [private]
```

The cities.

6.15.5.3 complete

```
bool Path::complete  [private]
```

True if path is full.

6.15.5.4 distance

```
double Path::distance  [private]
```

The distance.

6.15.5.5 fitness

```
double Path::fitness  [private]
```

The fitness.

6.15.5.6 max

```
int Path::max  [private]
```

Maximum number of the cities.

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

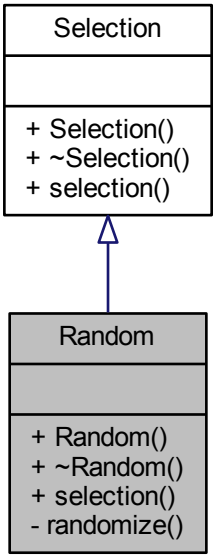
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Path.h
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Path.cpp

6.16 Random Class Reference

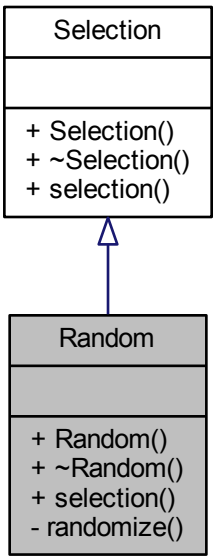
Concrete selection strategy class. Implements the Selection strategy.

```
#include <Random.h>
```

Inheritance diagram for Random:



Collaboration diagram for Random:



Public Member Functions

- Random ()=default
- ∼Random ()=default
- std::pair< Path, Path > selection (std::unique_ptr< Generation > &generation) override

  Selects two chromosomes from the generation. Method used to execute the selection strategy.

Private Member Functions

- int randomize (int size)

  Generates random number which is an index of the selected individual.

6.16.1 Detailed Description

Concrete selection strategy class. Implements the Selection strategy.

Selection method randomly picks 2 chromosomes from the generation.

6.16.2 Constructor & Destructor Documentation

6.16.2.1 Random()

```
Random::Random ( )  [default]
```

6.16.2.2 ~Random()

```
Random::~Random ( )  [default]
```

6.16.3 Member Function Documentation

6.16.3.1 randomize()

```
int Random::randomize (
            int size )  [private]
```

Generates random number which is an index of the selected individual.

Returns

Random number.

6.16.3.2 selection()

```
std::pair< Path, Path > Random::selection (
            std::unique_ptr< Generation > & generation )  [override], [virtual]
```

Selects two chromosomes from the generation. Method used to execute the selection strategy.

Parameters

generation	The generation.

Returns

std::pair containing selected chromosomes (parents);

Reimplemented from Selection.

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

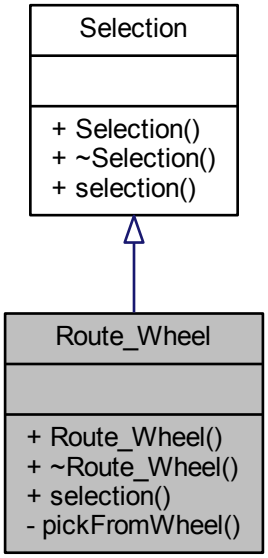
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Random.h
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Random.cpp

6.17 Route_Wheel Class Reference

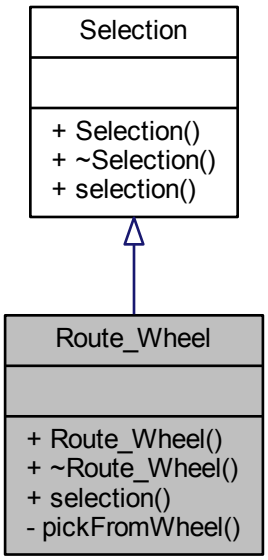
Concrete selection strategy class. Implements the Selection strategy.

```
#include <Route_Wheel.h>
```

Inheritance diagram for Route_Wheel:



Collaboration diagram for Route_Wheel:



Public Member Functions

- Route_Wheel ()=default
- ~Route_Wheel ()=default
- std::pair< Path, Path > selection (std::unique_ptr< Generation > &generation) override

  Selects two chromosomes from the generation. Method used to execute the selection strategy.

Private Member Functions

- int pickFromWheel (std::vector< double > wheel)

  Returns an index of the individual picked with a probability proportional to its fitness.

6.17.1 Detailed Description

Concrete selection strategy class. Implements the Selection strategy.

Selection method picks two chromosomes with a probability which is proportional to its fitness.

6.17.2 Constructor & Destructor Documentation

6.17.2.1 Route_Wheel()

```
Route_Wheel::Route_Wheel ( )  [default]
```

6.17.2.2 ~Route_Wheel()

```
Route_Wheel::~Route_Wheel ( )  [default]
```

6.17.3 Member Function Documentation

6.17.3.1 pickFromWheel()

```
int Route_Wheel::pickFromWheel (
            std::vector< double > wheel )  [private]
```

Returns an index of the individual picked with a probability proportional to its fitness.

Returns

Index.

6.17.3.2 selection()

```
std::pair< Path, Path > Route_Wheel::selection (
            std::unique_ptr< Generation > & generation )  [override], [virtual]
```

Selects two chromosomes from the generation. Method used to execute the selection strategy.

Parameters

generation	The generation.

Returns

std::pair containing selected chromosomes (parents);

Reimplemented from Selection.

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

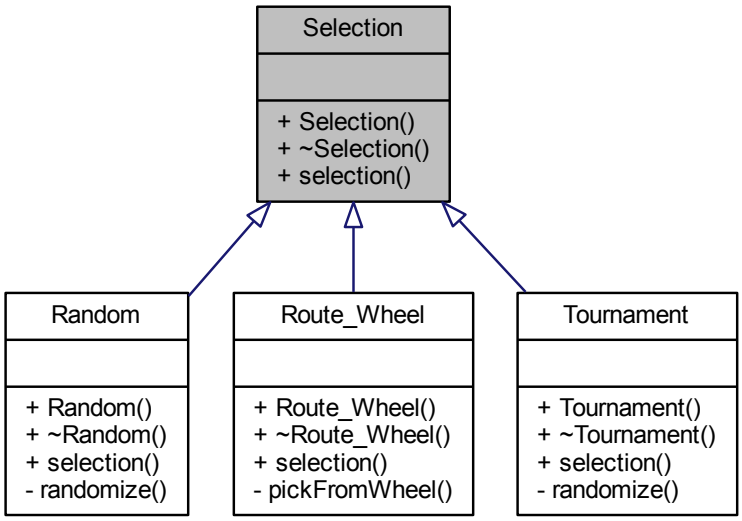
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Route_Wheel.h
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Route_Wheel.cpp

6.18 Selection Class Reference

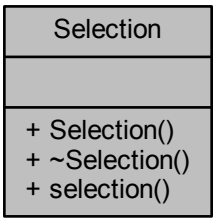
Abstract class for selection strategy interface.

`#include <Selection.h>`

Inheritance diagram for Selection:



Collaboration diagram for Selection:



Public Member Functions

- Selection ()=default
- virtual ~Selection ()=default
- virtual std::pair< Path, Path > selection (std::unique_ptr< Generation > &generation)

  Selects two chromosomes from the generation. Method used to execute the selection strategy.

6.18.1 Detailed Description

Abstract class for selection strategy interface.

6.18.2 Constructor & Destructor Documentation

6.18.2.1 Selection()

```
Selection::Selection ( )  [default]
```

6.18.2.2 ~Selection()

```
virtual Selection::~Selection ( )  [virtual], [default]
```

6.18.3 Member Function Documentation

6.18.3.1 selection()

```
std::pair< Path, Path > Selection::selection (
            std::unique_ptr< Generation > & generation )  [virtual]
```

Selects two chromosomes from the generation. Method used to execute the selection strategy.

Parameters

generation	The generation.

Returns

std::pair containing selected chromosomes (parents);

Reimplemented in Random, Route_Wheel, and Tournament.

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

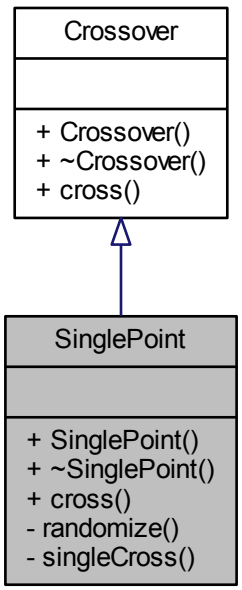
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Selection.h
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Selection.cpp

6.19 SinglePoint Class Reference

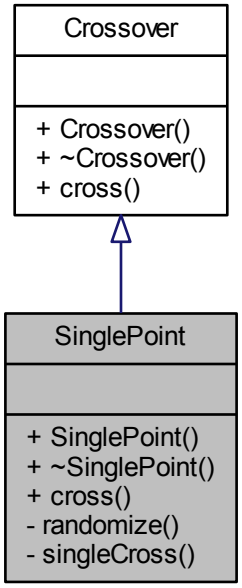
Concrete crossover strategy class. Implements the Crossover strategy.

```
#include <Single_Point.h>
```

Inheritance diagram for SinglePoint:



Collaboration diagram for SinglePoint:



Public Member Functions

- SinglePoint ()=default
- ~SinglePoint ()=default
- std::pair< Path, Path > cross (std::pair< Path, Path > &parents_pair) override

  Crosses a given pair of parents and returns their children. The paths from parents are mixed, the result of that operation creates the path for the children.

Private Member Functions

- int randomize (int size)

  Generates random number which is the crossover point.
- void singleCross (Path &child, int border, Path p1, Path p2)

  Crosses the parents and assigns the result to the child.

6.19.1 Detailed Description

Concrete crossover strategy class. Implements the Crossover strategy.

Random crossover point is selected and the tails (cities after that point) of its two parents paths are swapped to get new paths (children). Cross method creates children, pick the crossover point, performs the crossover on the parents (using helper singleCross function) and returns the result of this operation.

6.19.2 Constructor & Destructor Documentation

6.19.2.1 SinglePoint()

```
SinglePoint::SinglePoint ( )  [default]
```

6.19.2.2 ~SinglePoint()

```
SinglePoint::~SinglePoint ( )  [default]
```

6.19.3 Member Function Documentation

6.19.3.1 cross()

```
std::pair< Path, Path > SinglePoint::cross (
            std::pair< Path, Path > & parents_pair )  [override], [virtual]
```

Crosses a given pair of parents and returns their children. The paths from parents are mixed, the result of that operation creates the path for the children.

Parameters

parents_pair	Pair of two individuals(parents).

Returns

std::pair containing new individuals (children) resulting from their crossing;

Reimplemented from Crossover.

6.19.3.2 randomize()

```
int SinglePoint::randomize (
            int size )  [private]
```

Generates random number which is the crossover point.

Parameters

size	Number of cities in a path.

Returns

Random number.

6.19.3.3 singleCross()

```
void SinglePoint::singleCross (
            Path & child,
            int border,
            Path p1,
            Path p2 )  [private]
```

Crosses the parents and assigns the result to the child.

Parameters

in,out	*child*	Child with empty path.
	border	Crossover point.
	p1	The first parent.
	p2	The second parent.

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

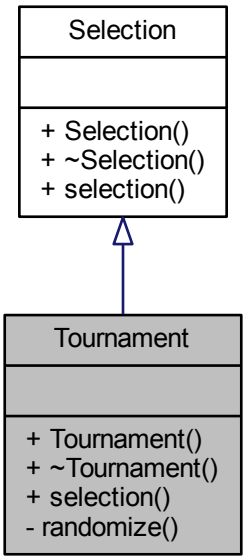
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Single_Point.h
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Single_Point.cpp

6.20 Tournament Class Reference

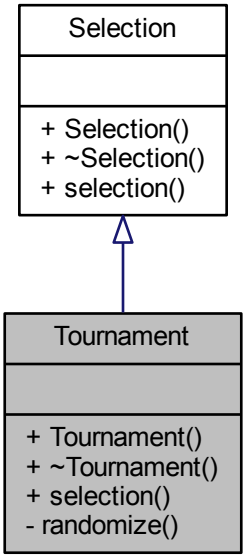
Concrete selection strategy class. Implements the Selection strategy.

```
#include <Tournament.h>
```

Inheritance diagram for Tournament:



Collaboration diagram for Tournament:



Public Member Functions

- Tournament ()=default
- ~Tournament ()=default
- std::pair< Path, Path > selection (std::unique_ptr< Generation > &generation) override

  Selects two chromosomes from the generation. Method used to execute the selection strategy.

Private Member Functions

- int randomize (int size)

  Generates random number which is an index of the individual from the pool.

6.20.1 Detailed Description

Concrete selection strategy class. Implements the Selection strategy.

Selection method picks the two fittest individuals from a randomly selected pool of four.

6.20.2 Constructor & Destructor Documentation

6.20.2.1 Tournament()

```
Tournament::Tournament ( )  [default]
```

6.20.2.2 ~Tournament()

```
Tournament::~Tournament ( )  [default]
```

6.20.3 Member Function Documentation

6.20.3.1 randomize()

```
int Tournament::randomize (
            int size )  [private]
```

Generates random number which is an index of the individual from the pool.

Returns

Random number.

6.20.3.2 selection()

```
std::pair< Path, Path > Tournament::selection (
            std::unique_ptr< Generation > & generation )  [override], [virtual]
```

Selects two chromosomes from the generation. Method used to execute the selection strategy.

Parameters

generation	The generation.

Returns

std::pair containing selected chromosomes (parents);

Reimplemented from Selection.

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

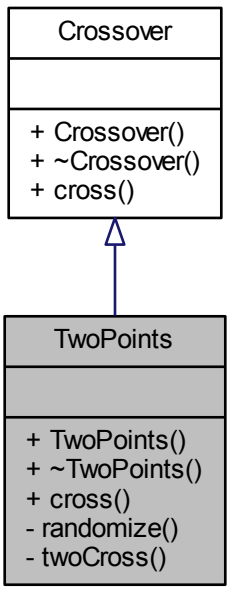
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Tournament.h
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Tournament.cpp

6.21 TwoPoints Class Reference

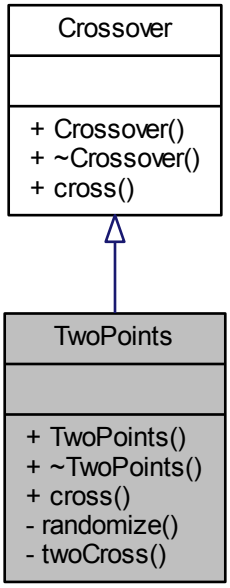
Concrete crossover strategy class. Implements the Crossover strategy.

`#include <Two_Points.h>`

Inheritance diagram for TwoPoints:



Collaboration diagram for TwoPoints:



Public Member Functions

- TwoPoints ()=default
- ~TwoPoints ()=default
- std::pair< Path, Path > cross (std::pair< Path, Path > &parents_pair) override

  Crosses a given pair of parents and returns their children. The paths from parents are mixed, the result of that operation creates the path for the children.

Private Member Functions

- std::pair< int, int > randomize (int size)

  Generates a pair of random numbers which are the crossover points.
- void twoCross (Path &child, std::pair< int, int > border, Path p1, Path p2)

  Crosses the parents and assigns the result to the child.

6.21.1 Detailed Description

Concrete crossover strategy class. Implements the Crossover strategy.

Two random crossover points are selected and the cities between this points of its two parents paths are swapped to get new paths (children). Cross method creates children, pick the crossover points, performs the crossover on the parents (using helper twoCross function) and returns the result of this operation.

6.21.2 Constructor & Destructor Documentation

6.21.2.1 TwoPoints()

```
TwoPoints::TwoPoints ( )  [default]
```

6.21.2.2 ~TwoPoints()

```
TwoPoints::~TwoPoints ( )  [default]
```

6.21.3 Member Function Documentation

6.21.3.1 cross()

```
std::pair< Path, Path > TwoPoints::cross (
            std::pair< Path, Path > & parents_pair )  [override], [virtual]
```

Crosses a given pair of parents and returns their children. The paths from parents are mixed, the result of that operation creates the path for the children.

Parameters

parents_pair	Pair of two individuals(parents).

Returns

std::pair containing new individuals (children) resulting from their crossing;

Reimplemented from Crossover.

6.21.3.2 randomize()

```
std::pair< int, int > TwoPoints::randomize (
            int size )  [private]
```

Generates a pair of random numbers which are the crossover points.

Parameters

size	Number of cities in a path.

Returns

std::pair crossover points;

6.21.3.3 twoCross()

```
void TwoPoints::twoCross (
            Path & child,
            std::pair< int, int > border,
            Path p1,
            Path p2 )  [private]
```

Crosses the parents and assigns the result to the child.

Parameters

in,out	*child*	Child with empty path.
	border	Crossover points.
	p1	The first parent.
	p2	The second parent.

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

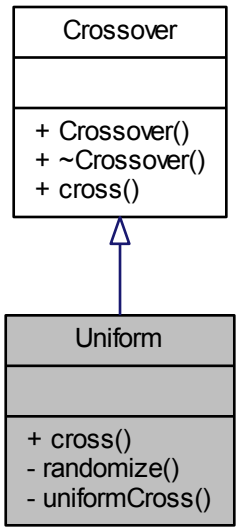
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Two_Points.h
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Two_Points.cpp

6.22 Uniform Class Reference

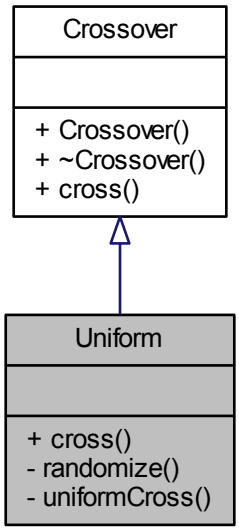
Concrete crossover strategy class. Implements the Crossover strategy.

```
#include <Uniform.h>
```

Inheritance diagram for Uniform:



Collaboration diagram for Uniform:



Public Member Functions

- std::pair< Path, Path > cross (std::pair< Path, Path > &parents_pair) override

Crosses a given pair of parents and returns their children. The paths from parents are mixed, the result of that operation creates the path for the children.

Private Member Functions

- int randomize ()

  Randomly returns 0 or 1.
- void uniformCross (Path &child, Path &p1, Path &p2)

  Crosses the parents and assigns the result to the child.

6.22.1 Detailed Description

Concrete crossover strategy class. Implements the Crossover strategy.

In this crossover method each gene (city) is selected randomly from one of the corresponding genes of the parent chromosomes (paths). Cross method creates children, performs the crossover on the parents (using helper uniformCross function) and returns the result of this operation.

6.22.2 Member Function Documentation

6.22.2.1 cross()

```
std::pair< Path, Path > Uniform::cross (
            std::pair< Path, Path > & parents_pair )  [override], [virtual]
```

Crosses a given pair of parents and returns their children. The paths from parents are mixed, the result of that operation creates the path for the children.

Parameters

parents_pair	Pair of two individuals(parents).

Returns

std::pair containing new individuals (children) resulting from their crossing;

Reimplemented from Crossover.

6.22.2.2 randomize()

```
int Uniform::randomize ( )  [private]
```

Randomly returns 0 or 1.

0 means city will be picked from the first parent, 1 from the second.

Returns

0 or 1.

6.22.2.3 uniformCross()

```
void Uniform::uniformCross (
            Path & child,
            Path & p1,
            Path & p2 )  [private]
```

Crosses the parents and assigns the result to the child.

Parameters

in,out	*child*	Child with empty path.
	p1	The first parent.
	p2	The second parent.

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

- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Uniform.h
- C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Uniform.cpp

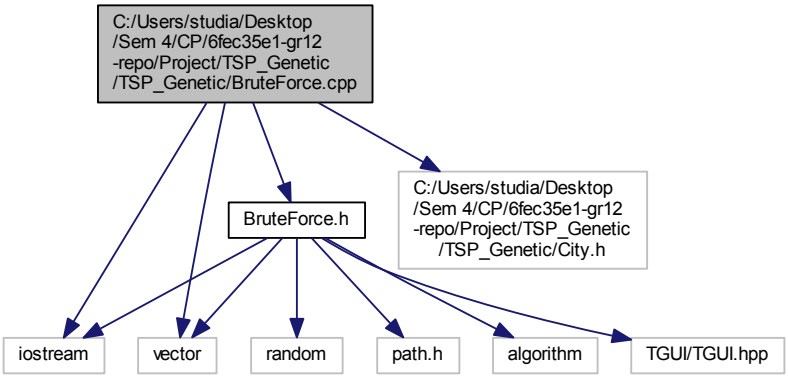
Chapter 7

File Documentation

7.1 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/BruteForce.cpp File Reference

```
#include "BruteForce.h"
```

Include dependency graph for BruteForce.cpp:



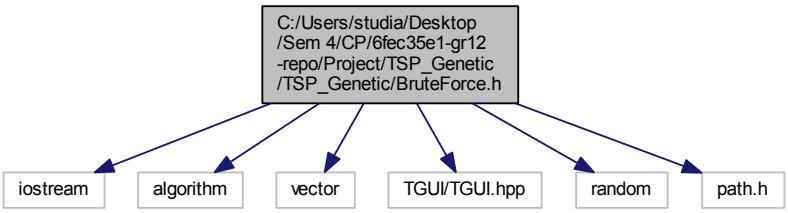
7.2 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/BruteForce.h File Reference

Declares the brute force class.

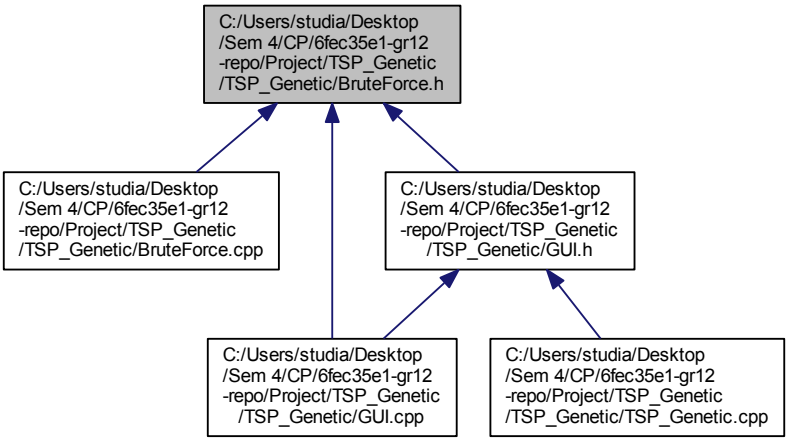
```
#include <iostream>
#include <algorithm>
#include <vector>
```

```
#include <TGUI/TGUI.hpp>
#include <random>
#include "path.h"
```

Include dependency graph for BruteForce.h:



This graph shows which files directly or indirectly include this file:



Classes

- class BruteForce

  Brute Force class.

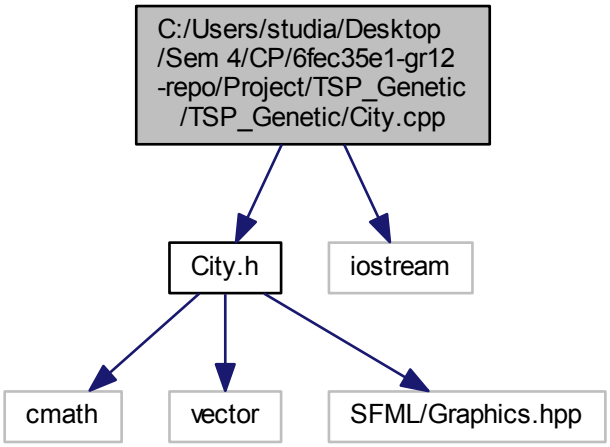
7.2.1 Detailed Description

Declares the brute force class.

7.3 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/City.cpp File Reference

```
#include "City.h"
#include <iostream>
```

Include dependency graph for City.cpp:

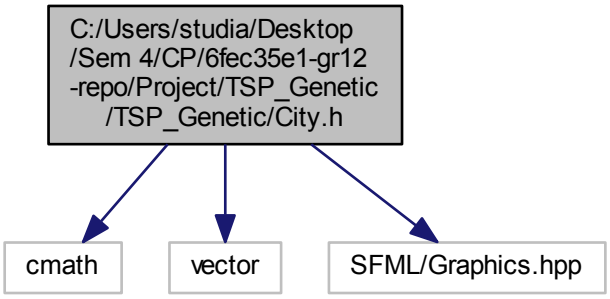


7.4 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/City.h File Reference

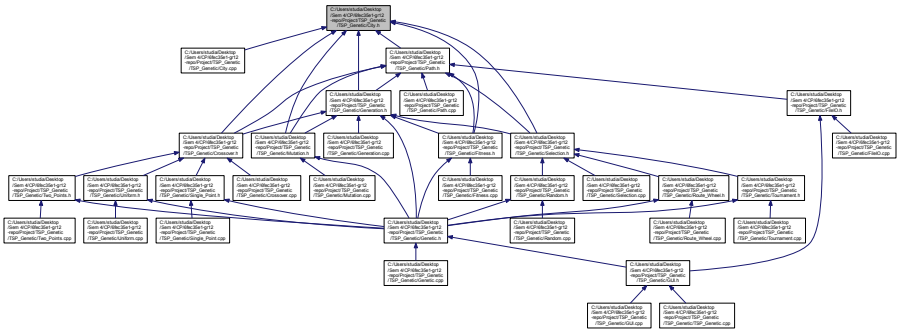
Declares the city class.

```
#include <cmath>
#include <vector>
#include <SFML/Graphics.hpp>
```

Include dependency graph for City.h:



This graph shows which files directly or indirectly include this file:



Classes

- class City

  Represents the city.

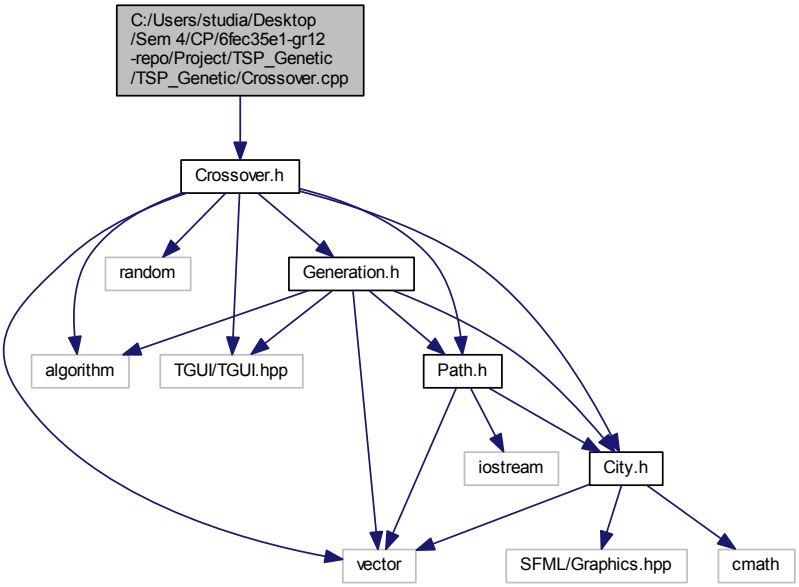
7.4.1 Detailed Description

Declares the city class.

7.5 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Crossover.cpp File Reference

```
#include "Crossover.h"
```

Include dependency graph for Crossover.cpp:

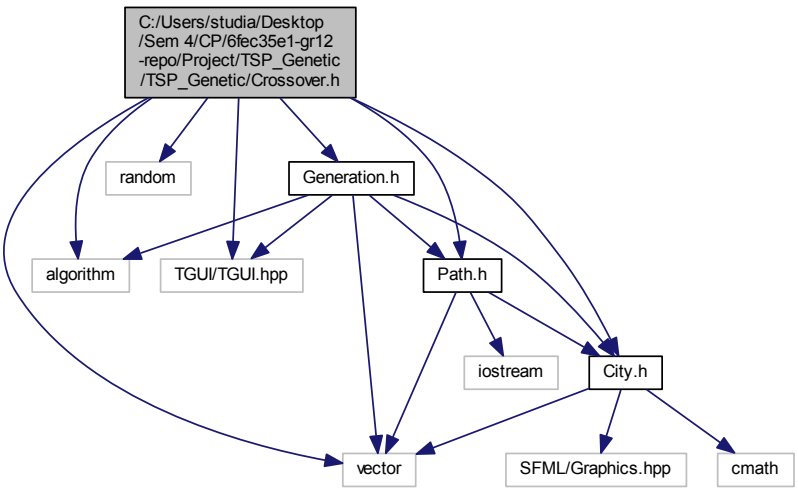


7.6 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Crossover.h File Reference

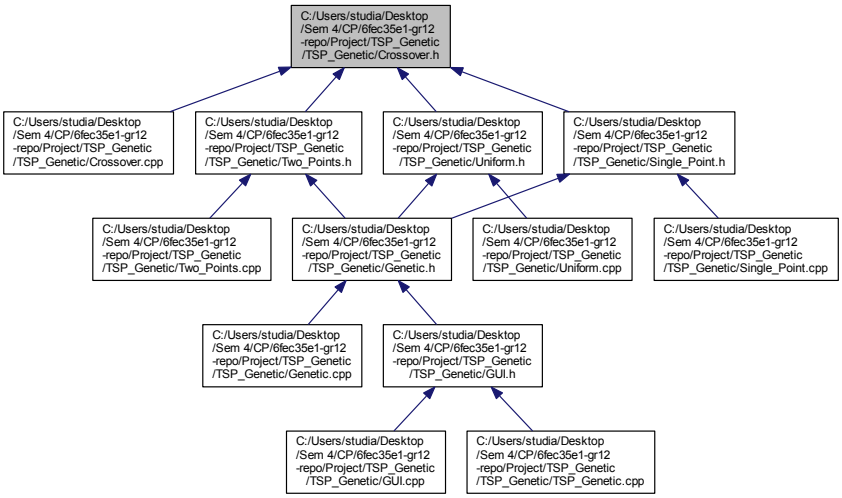
Declares the crossover strategy interface class.

```
#include <vector>
#include <algorithm>
#include <random>
#include <TGUI/TGUI.hpp>
#include "Generation.h"
#include "City.h"
#include "Path.h"
```

Include dependency graph for Crossover.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Crossover

  Abstract class for crossover strategy interface.

7.6.1 Detailed Description

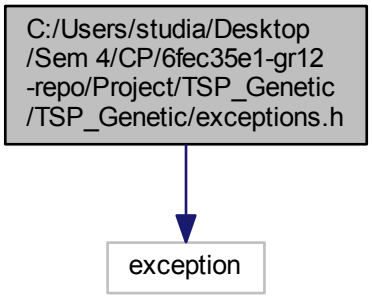
Declares the crossover strategy interface class.

7.7 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/exceptions.h File Reference

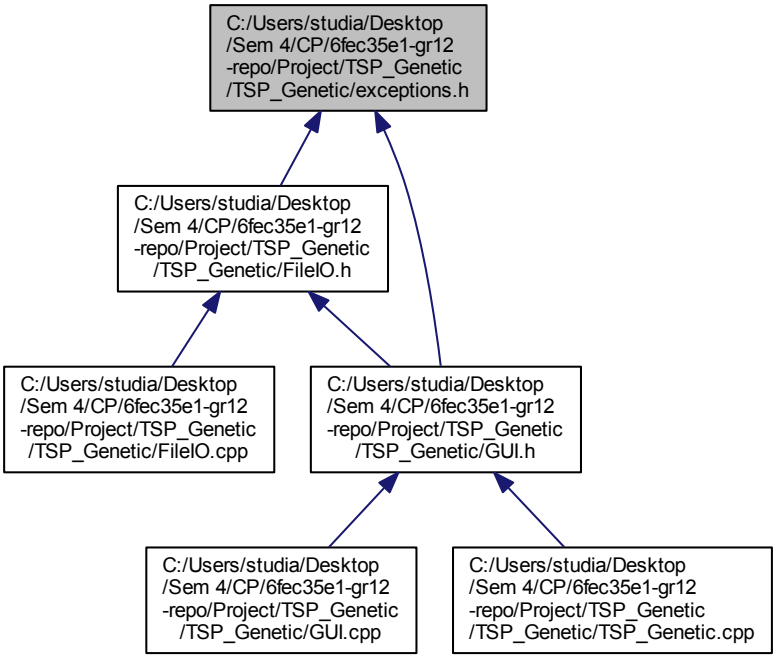
Contains custom exception classes.

```
#include <exception>
```

Include dependency graph for exceptions.h:



This graph shows which files directly or indirectly include this file:



Classes

- class FilePathExt

  Responsible for throwing a message when the path or extension of the file is invalid.

- class FilePath

  Responsible for throwing a message when the path of the file is invalid.

- class FileInput

  Responsible for throwing a message when the data in the file is invalid.

- class FileOpen

  Responsible for throwing a message when the file cannot be opened or created.

- class InputType

  Responsible for throwing a message when the line informing about the input type of the file is incorrect.

- class CitiesAmount

  Responsible for throwing a message when there is not enough data for declared number of cities.

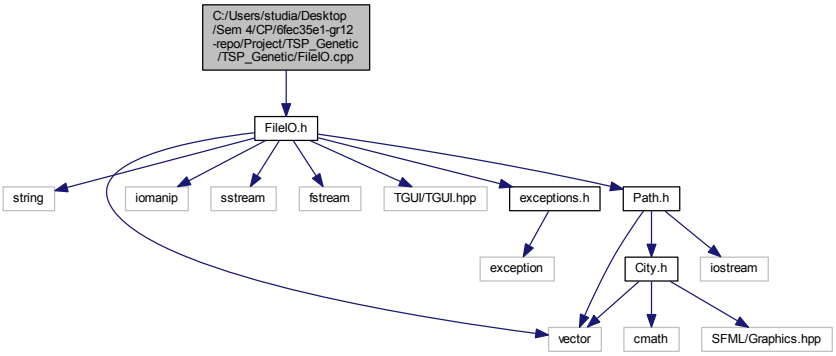
7.7.1 Detailed Description

Contains custom exception classes.

7.8 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/FileIO.cpp File Reference

```
#include "FileIO.h"
```

Include dependency graph for FileIO.cpp:



Namespaces

- FileIO

  The functions of processing input and output files.

Functions

- std::string FileIO::checkFile (std::string path)

  Validates the path and extension of the input file.

- void FileIO::geneticFile (std::string path, int number, std::shared_ptr< Path > &best, std::shared_ptr< std::vector< City >> &cities)

  Output file for the genetic algorithm solution.

- void FileIO::bruteFile (std::string path, std::shared_ptr< Path > &best, std::shared_ptr< std::vector< City >> &cities)

  Output file for the brute force solution.

- std::vector< std::vector< double > > FileIO::readDist (std::string file)

  Reads the input from the "DIST" type input file.

- std::vector< sf::Vector2f > FileIO::readCoord (std::string file)

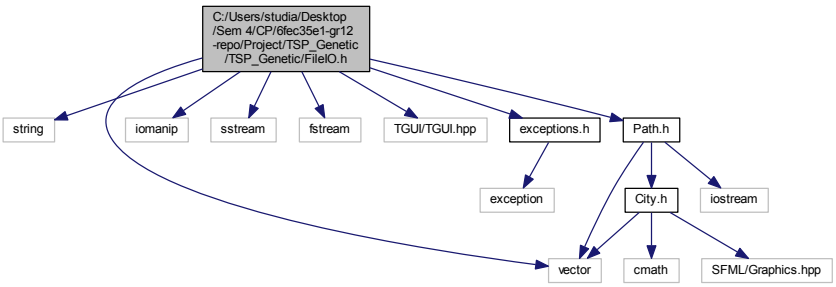
  Reads the input from the "COORD" type input file.

7.9 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/FileIO.h File Reference

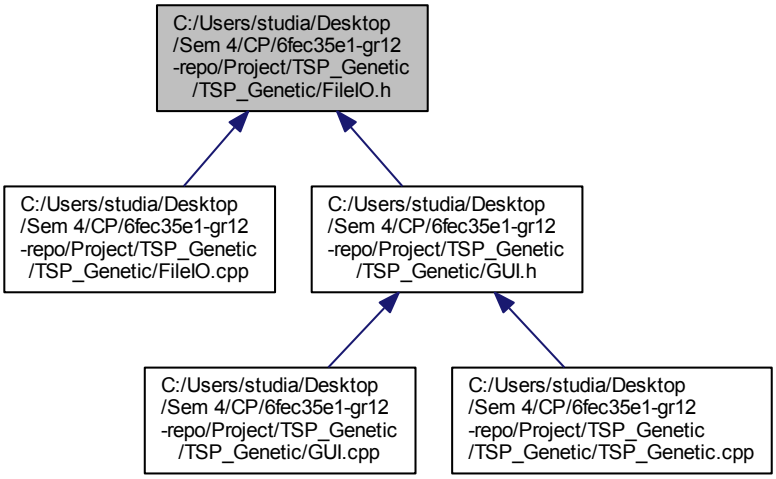
Declares the file i/o class.

```
#include <string>
#include <vector>
#include <iomanip>
#include <sstream>
#include <fstream>
#include "TGUI/TGUI.hpp"
#include "exceptions.h"
#include "Path.h"
```

Include dependency graph for FileIO.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- FileIO

  The functions of processing input and output files.

Functions

- std::string FileIO::checkFile (std::string path)

  Validates the path and extension of the input file.
- void FileIO::geneticFile (std::string path, int number, std::shared_ptr< Path > &best, std::shared_ptr< std::vector< City >> &cities)

  Output file for the genetic algorithm solution.
- void FileIO::bruteFile (std::string path, std::shared_ptr< Path > &best, std::shared_ptr< std::vector< City >> &cities)

  Output file for the brute force solution.
- std::vector< std::vector< double > > FileIO::readDist (std::string file)

  Reads the input from the "DIST" type input file.
- std::vector< sf::Vector2f > FileIO::readCoord (std::string file)

  Reads the input from the "COORD" type input file.

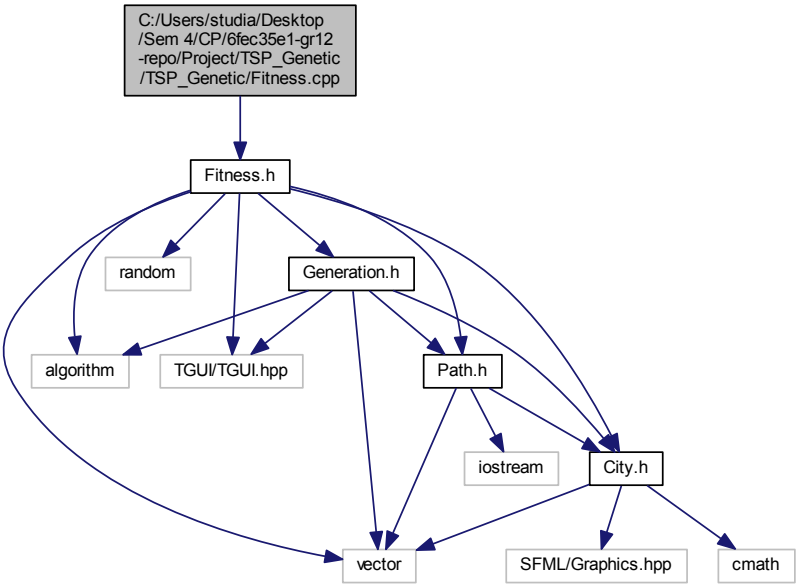
7.9.1 Detailed Description

Declares the file i/o class.

7.10 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Fitness.cpp File Reference

```
#include "Fitness.h"
```

Include dependency graph for Fitness.cpp:

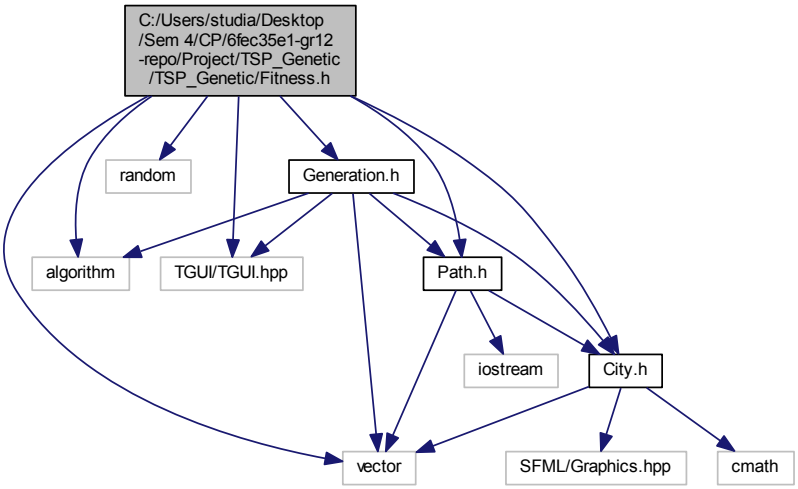


7.11 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Fitness.h File Reference

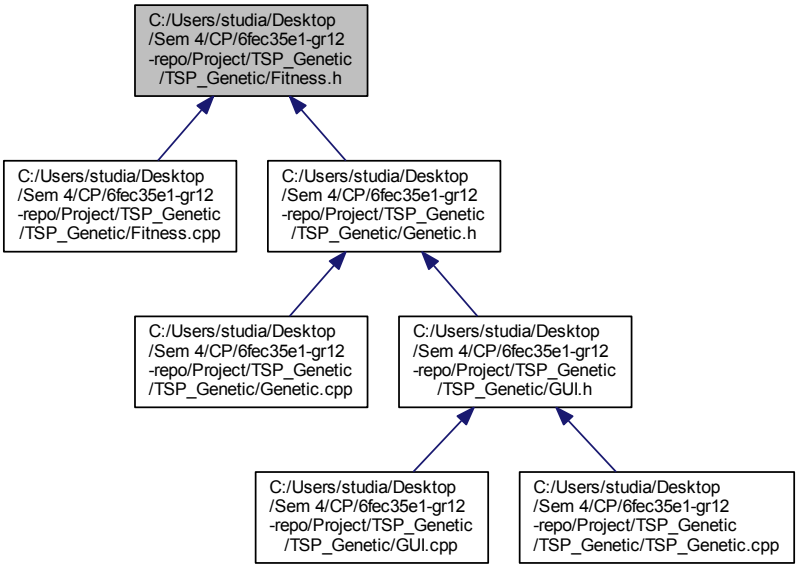
Declares the fitness class.

```
#include <vector>
#include <algorithm>
#include <random>
#include <TGUI/TGUI.hpp>
#include "Generation.h"
#include "City.h"
#include "Path.h"
```

Include dependency graph for Fitness.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Fitness

  Fitness class.

7.11.1 Detailed Description

Declares the fitness class.

7.12 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/form.txt File Reference

Variables

- Renderer
- backgroundcolordisabled = #E6E6E6
- backgroundcolordown = #EBEBEB
- backgroundcolorhover = white
- bordercolor = #3C3C3C
- bordercolordisabled = #7D7D7D
- bordercolordown = black
- bordercolorfocused = #1E1EB4
- bordercolorhover = black
- borders = (1, 1, 1, 1)
- textcolor = #3C3C3C
- textcolordisabled = #7D7D7D
- textcolordown = black
- textcolorhover = black
- texture = None
- texturedisabled = None
- texturedown = None
- texturefocused = None
- texturehover = None
- borderbelowtitlebar = 1
- distancetoside = 3
- paddingbetweenbuttons = 1
- showtextontitlebuttons = true
- texturebackground = None
- texturetitlebar = None
- titlebarcolor = white
- titlecolor = #3C3C3C
- closebutton
- caretcolor = black
- caretwidth = 1
- defaulttextcolor = #A0A0A0
- padding = (0, 0, 0, 0)
- selectedtextbackgroundcolor = #006EFF
- selectedtextcolor = white
- texturethumb = None
- texturethumbhover = None
- texturetrack = None
- texturetrackhover = None
- thumbcolor = #F5F5F5
- thumbcolorhover = white
- trackcolor = #F5F5F5

- trackcolorhover = white
- ChildWindow child1
- Size = (400, 300)
- Title = CURRENT
- TitleAlignment = Center
- TitleButtons = None
- renderer = &2
- ChildWindow child2
- PositionLocked = true
- Slider GenSize
- InvertedDirection = false
- Maximum = 2500
- Minimum = 20
- Position = (300, 490)
- Step = 20
- Value = 20
- Visible = false
- Button Start
- Text = Start
- TextSize = 13
- Button Stop
- Button Pause
- Label Gen_text
- ScrollbarPolicy = Never
- Slider Mutation
- Label Mut_text
- Slider NB
- Label NB_text
- Label Type_text
- Label shortest
- Label best
- Label current
- Slider Type
- Button citygen
- Slider selection
- Slider crossover
- Label current_best
- Label selection_text
- Label crossover_text
- Button input_button
- ChildWindow open_window
- UserData = ""
- TextBox file_text
- MaximumCharacters = 0
- ReadOnly = true
- EditBox path_box
- Button open_button
- EditBox output_path
- Button output_button
- Label output_txt

7.12.1 Variable Documentation

7.12.1.1 backgroundcolordisabled

```
backgroundcolordisabled = #E6E6E6
```

7.12.1.2 backgroundcolordown

```
backgroundcolordown = #EBEBEB
```

7.12.1.3 backgroundcolorhover

```
backgroundcolorhover = white
```

7.12.1.4 best

```
Label best
```

Initial value:

```
{
    Position = (570, 380)
```

7.12.1.5 borderbelowtitlebar

```
borderbelowtitlebar = 1
```

7.12.1.6 bordercolor

```
bordercolor = #3C3C3C
```

7.12.1.7 bordercolordisabled

```
bordercolordisabled = #7D7D7D
```

7.12.1.8 bordercolordown

```
bordercolordown = black
```

7.12.1.9 bordercolorfocused

```
bordercolorfocused = #1E1EB4
```

7.12.1.10 bordercolorhover

```
bordercolorhover = black
```

7.12.1.11 borders

```
borders = (1, 1, 1, 1)
```

7.12.1.12 caretcolor

```
caretcolor = black
```

7.12.1.13 caretwidth

```
caretwidth = 1
```

7.12.1.14 child1

```
ChildWindow child1
```

Initial value:

```
{
    PositionLocked = true
```

7.12.1.15 child2

```
ChildWindow child2
```

Initial value:

```
{
    Position = (400, 0)
```

7.12.1.16 citygen

```
Button citygen
```

Initial value:

```
{
    Position = (10, 360)
```

7.12.1.17 closebutton

```
closebutton
```

Initial value:

```
{
        backgroundcolor = #F5F5F5
```

7.12.1.18 crossover

```
Slider crossover
```

Initial value:

```
{
    ChangeValueOnScroll = true
```

7.12.1.19 crossover_text

```
Label crossover_text
```

Initial value:

```
{
    AutoSize = true
```

7.12.1.20 current

```
Label current
```

Initial value:

```
{
    AutoSize = true
```

7.12.1.21 current_best

```
Label current_best
```

Initial value:

```
{
    AutoSize = true
```

7.12.1.22 defaulttextcolor

```
defaulttextcolor = #A0A0A0
```

7.12.1.23 distancetoside

```
distancetoside = 3
```

7.12.1.24 file_text

```
TextBox file_text
```

Initial value:

```
{
        HorizontalScrollbarPolicy = Never
```

7.12.1.25 Gen_text

```
Label Gen_text
```

Initial value:

```
{
    AutoSize = true
```

7.12.1.26 GenSize

```
Slider GenSize
```

Initial value:

```
{
    ChangeValueOnScroll = true
```

7.12.1.27 input_button

```
Button input_button
```

Initial value:

```
{
    Position = (130, 360)
```

7.12.1.28 InvertedDirection

```
InvertedDirection = false
```

7.12.1.29 Maximum

```
Maximum = 2500
```

7.12.1.30 MaximumCharacters

```
MaximumCharacters = 0
```

7.12.1.31 Minimum

```
Minimum = 20
```

7.12.1.32 Mut_text

```
Label Mut_text
```

Initial value:

```
{
    AutoSize = true
```

7.12.1.33 Mutation

```
Slider Mutation
```

Initial value:

```
{
    ChangeValueOnScroll = true
```

7.12.1.34 NB

```
Slider NB
```

Initial value:

```
{
    ChangeValueOnScroll = true
```

7.12.1.35 NB_text

```
Label NB_text
```

Initial value:

```
{
    AutoSize = true
```

7.12.1.36 open_button

```
Button open_button
```

Initial value:

```
{
        Position = (160, 250)
```

7.12.1.37 open_window

```
ChildWindow open_window
```

Initial value:

```
{
    Enabled = false
```

7.12.1.38 output_button

```
Button output_button
```

Initial value:

```
{
    Position = (600, 530)
```

7.12.1.39 output_path

```
EditBox output_path
```

Initial value:

```
{
    Position = (560, 490)
```

7.12.1.40 output_txt

```
Label output_txt
```

Initial value:

```
{
    Position = (610, 460)
```

7.12.1.41 padding

```
padding = (0, 0, 0, 0)
```

7.12.1.42 paddingbetweenbuttons

```
paddingbetweenbuttons = 1
```

7.12.1.43 path_box

```
EditBox path_box
```

Initial value:

```
{
        DefaultText = "write path here"
```

7.12.1.44 Pause

```
Button Pause
```

Initial value:

```
{
    Position = (370, 360)
```

7.12.1.45 Position

```
Position = (300, 490)
```

7.12.1.46 PositionLocked

```
PositionLocked = true
```

7.12.1.47 ReadOnly

```
ReadOnly = true
```

7.12.1.48 Renderer

```
Renderer
```

Initial value:

```
{
    backgroundcolor = #F5F5F5
```

7.12.1.49 renderer

```
renderer = &2
```

7.12.1.50 ScrollbarPolicy

```
ScrollbarPolicy = Never
```

7.12.1.51 selectedtextbackgroundcolor

```
selectedtextbackgroundcolor = #006EFF
```

7.12.1.52 selectedtextcolor

```
selectedtextcolor = white
```

7.12.1.53 selection

```
Slider selection
```

Initial value:

```
{
    ChangeValueOnScroll = true
```

7.12.1.54 selection_text

```
Label selection_text
```

Initial value:

```
{
    AutoSize = true
```

7.12.1.55 shortest

```
Label shortest
```

Initial value:

```
{
    AutoSize = true
```

7.12.1.56 showtextontitlebuttons

```
showtextontitlebuttons = true
```

7.12.1.57 Size

```
Size = (400, 300)
```

7.12.1.58 Start

```
Button Start
```

Initial value:

```
{
    Position = (300, 360)
```

7.12.1.59 Step

```
Step = 20
```

7.12.1.60 Stop

```
Button Stop
```

Initial value:

```
{
    Position = (450, 360)
```

7.12.1.61 Text

```
Text = Start
```

7.12.1.62 textcolor

```
textcolor = #3C3C3C
```

7.12.1.63 textcolordisabled

```
textcolordisabled = #7D7D7D
```

7.12.1.64 textcolordown

```
textcolordown = black
```

7.12.1.65 textcolorhover

```
textcolorhover = black
```

7.12.1.66 TextSize

```
TextSize = 13
```

7.12.1.67 texture

```
texture = None
```

7.12.1.68 texturebackground

```
texturebackground = None
```

7.12.1.69 texturedisabled

```
texturedisabled = None
```

7.12.1.70 texturedown

```
texturedown = None
```

7.12.1.71 texturefocused

```
texturefocused = None
```

7.12.1.72 texturehover

```
texturehover = None
```

7.12.1.73 texturethumb

```
texturethumb = None
```

7.12.1.74 texturethumbhover

```
texturethumbhover = None
```

7.12.1.75 texturetitlebar

```
texturetitlebar = None
```

7.12.1.76 texturetrack

```
texturetrack = None
```

7.12.1.77 texturetrackhover

```
texturetrackhover = None
```

7.12.1.78 thumbcolor

```
thumbcolor = #F5F5F5
```

7.12.1.79 thumbcolorhover

```
thumbcolorhover = white
```

7.12.1.80 Title

```
Title = CURRENT
```

7.12.1.81 TitleAlignment

```
TitleAlignment = Center
```

7.12.1.82 titlebarcolor

```
titlebarcolor = white
```

7.12.1.83 TitleButtons

```
TitleButtons = None
```

7.12.1.84 titlecolor

```
titlecolor = #3C3C3C
```

7.12.1.85 trackcolor

```
trackcolor = #F5F5F5
```

7.12.1.86 trackcolorhover

```
trackcolorhover = white
```

7.12.1.87 Type

Slider Type

Initial value:

```
{
    ChangeValueOnScroll = true
```

7.12.1.88 Type_text

Label Type_text

Initial value:

```
{
    Position = (10, 390)
```

7.12.1.89 UserData

UserData = ""

7.12.1.90 Value

Value = 20

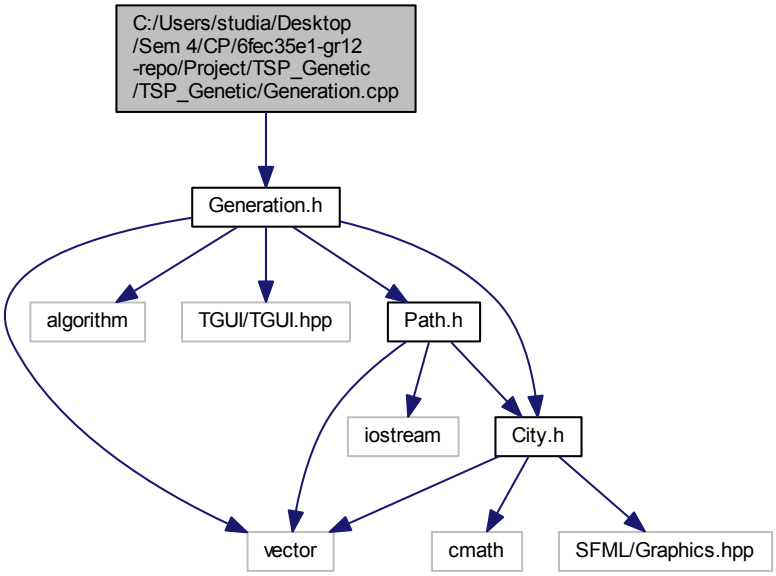
7.12.1.91 Visible

Visible = false

7.13 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Generation.cpp File Reference

```
#include "Generation.h"
```

Include dependency graph for Generation.cpp:

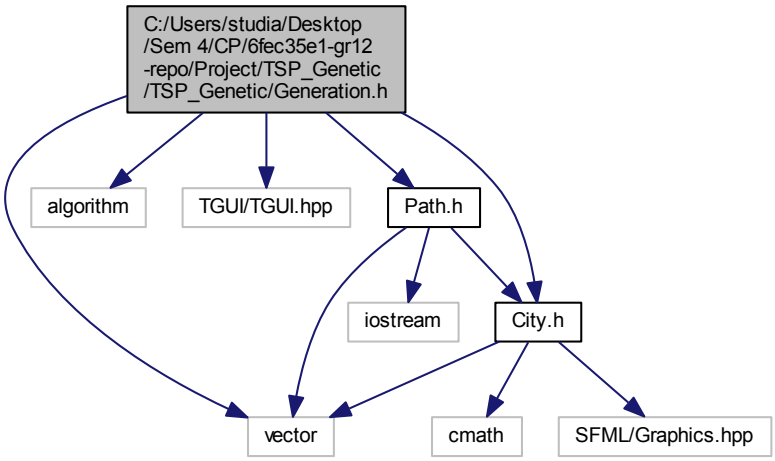


7.14 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Generation.h File Reference

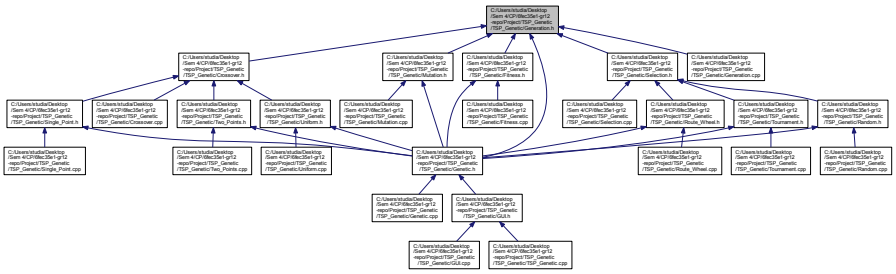
Declares the generation class.

```
#include <vector>
#include <algorithm>
#include <TGUI/TGUI.hpp>
#include "City.h"
#include "Path.h"
```

Include dependency graph for Generation.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Generation

  Represents the generation.

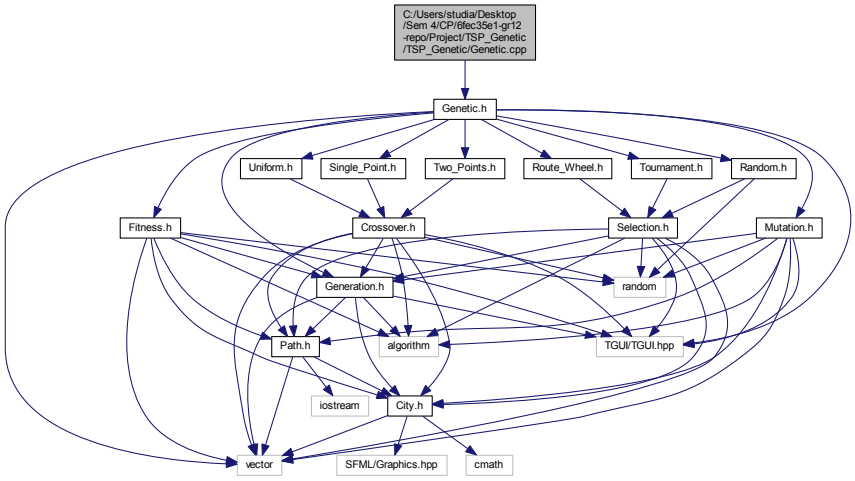
7.14.1 Detailed Description

Declares the generation class.

7.15 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Genetic.cpp File Reference

```
#include "Genetic.h"
```

Include dependency graph for Genetic.cpp:

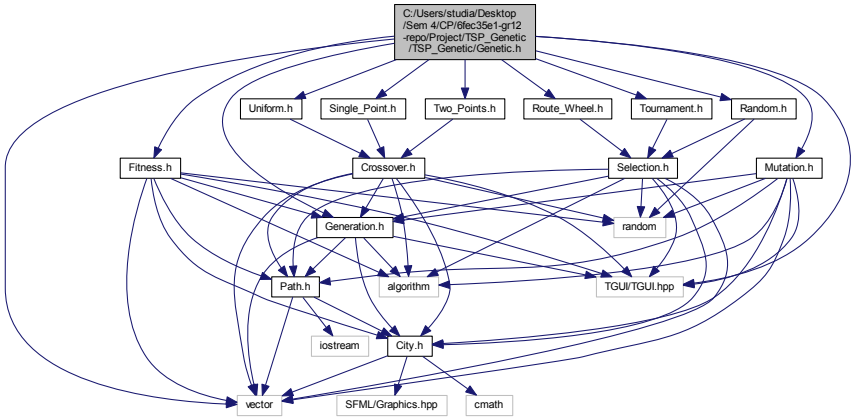


7.16 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Genetic.h File Reference

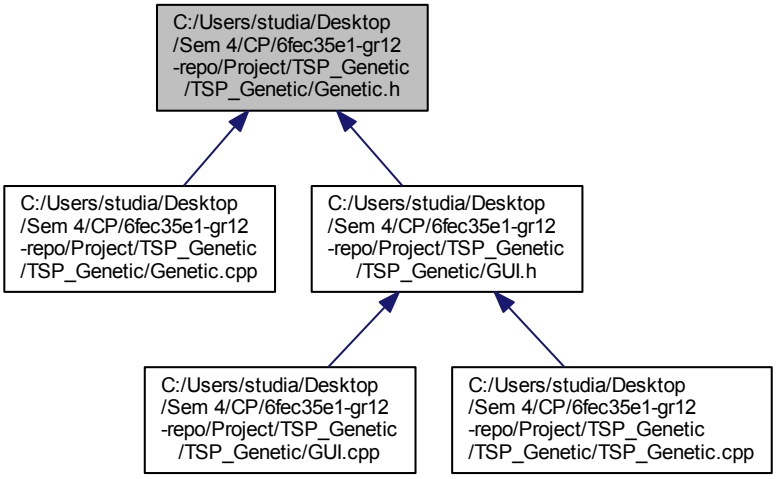
Genetic Algorithm class.

```
#include <vector>
#include <TGUI/TGUI.hpp>
#include "Generation.h"
#include "Mutation.h"
#include "Fitness.h"
#include "Single_Point.h"
#include "Two_Points.h"
#include "Uniform.h"
#include "Route_Wheel.h"
#include "Tournament.h"
#include "Random.h"
```

Include dependency graph for Genetic.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Genetic

  Declares the Genetic Algorithm class.

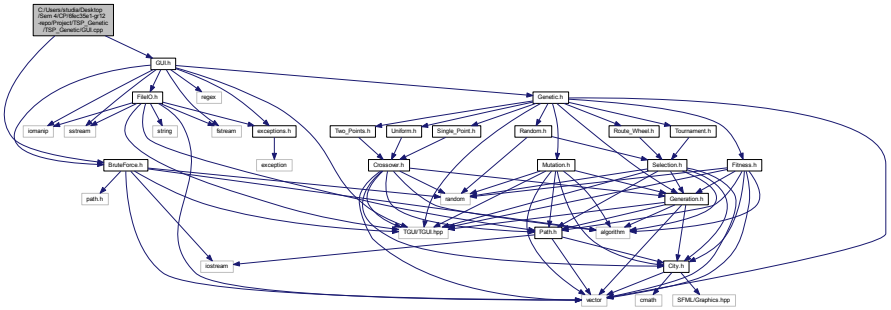
7.16.1 Detailed Description

Genetic Algorithm class.

7.17 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/GUI.cpp File Reference

```
#include "GUI.h"
#include "BruteForce.h"
```

Include dependency graph for GUI.cpp:

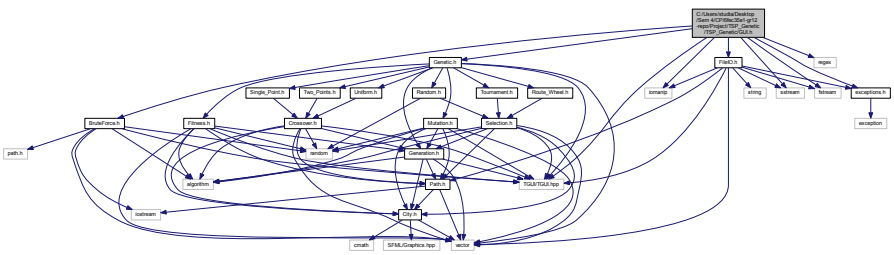


7.18 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/GUI.h File Reference

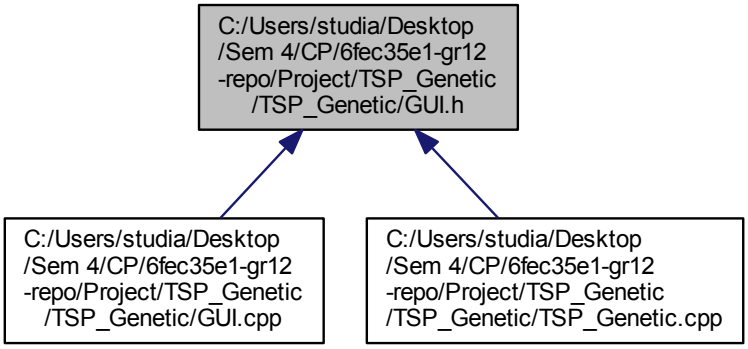
Declares the graphical user interface class.

```
#include <TGUI/TGUI.hpp>
#include <regex>
#include <iomanip>
#include <sstream>
#include <fstream>
#include "BruteForce.h"
#include "Genetic.h"
#include "exceptions.h"
#include "FileIO.h"
```

Include dependency graph for GUI.h:



This graph shows which files directly or indirectly include this file:



Classes

- class GUI

  Graphical User Interface.

7.18.1 Detailed Description

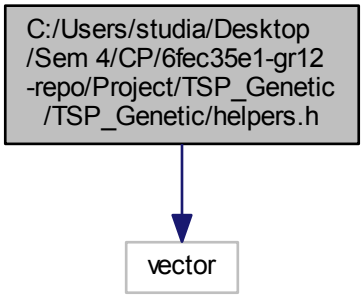
Declares the graphical user interface class.

7.19 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/helpers.h File Reference

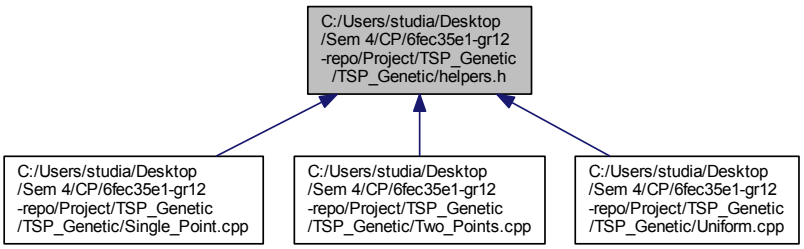
Namespace with helper functions.

```
#include <vector>
```

Include dependency graph for helpers.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- helpers

  Helper functions.

Functions

- template<class T >
  bool helpers::contains (const std::vector< T > &vec, const std::pair< int, int > border, const T &value)

  Checks whether the vector contains the given value in the given range.

- template<class T >
  bool helpers::contains (const std::vector< T > &vec, const int border, const T &value)

  Checks whether the vector contains the given value in the range from the beginning to the given border.

- template<class T >
  bool helpers::contains (const std::vector< T > &vec, const T &value)

  Checks whether the vector contains the given value.

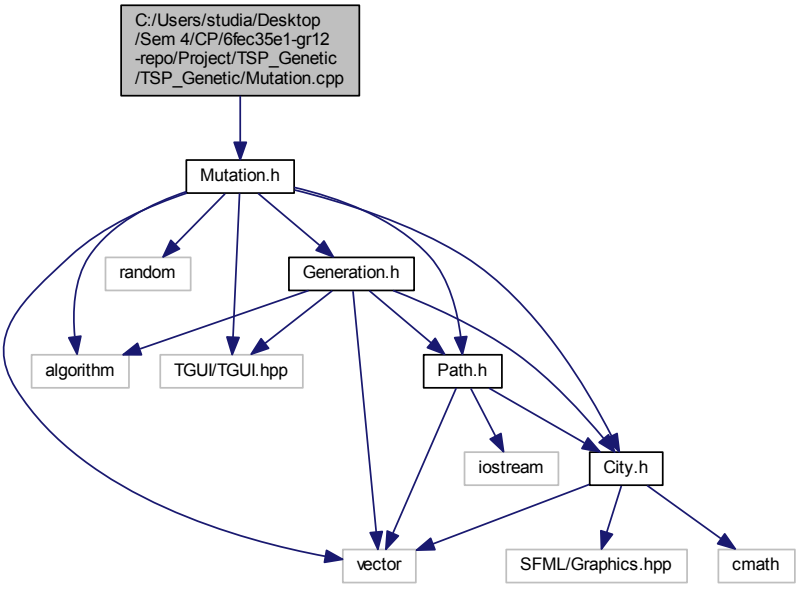
7.19.1 Detailed Description

Namespace with helper functions.

7.20 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Mutation.cpp File Reference

```
#include "Mutation.h"
```

Include dependency graph for Mutation.cpp:

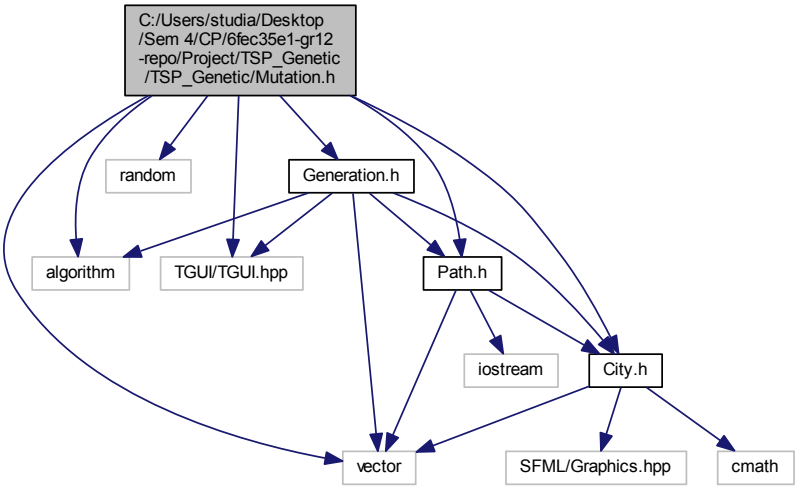


7.21 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Mutation.h File Reference

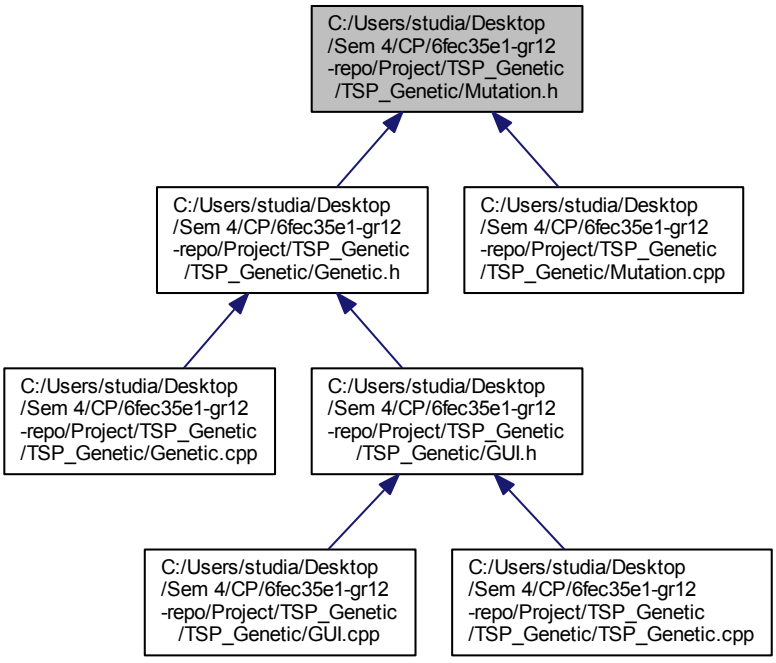
Declares the mutation class.

```
#include <vector>
#include <algorithm>
#include <random>
#include <TGUI/TGUI.hpp>
#include "Generation.h"
#include "City.h"
#include "Path.h"
```

Include dependency graph for Mutation.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Mutation

  Mutation class.

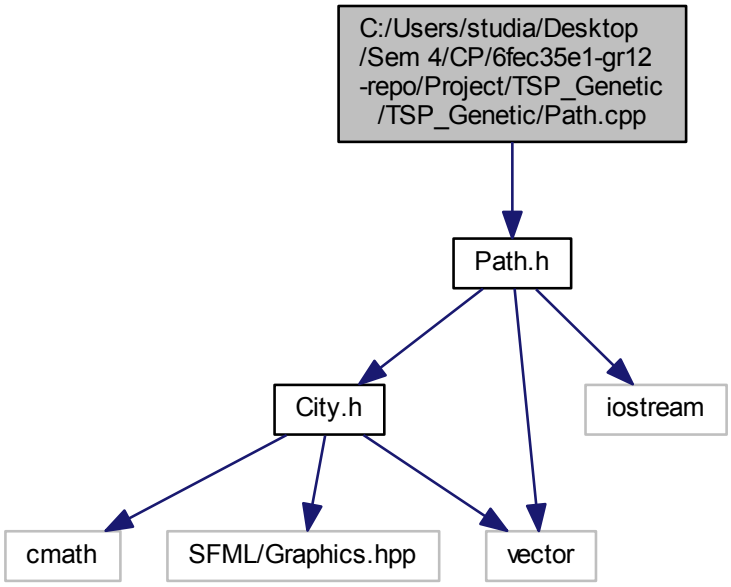
7.21.1 Detailed Description

Declares the mutation class.

7.22 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Path.cpp File Reference

```
#include "Path.h"
```

Include dependency graph for Path.cpp:



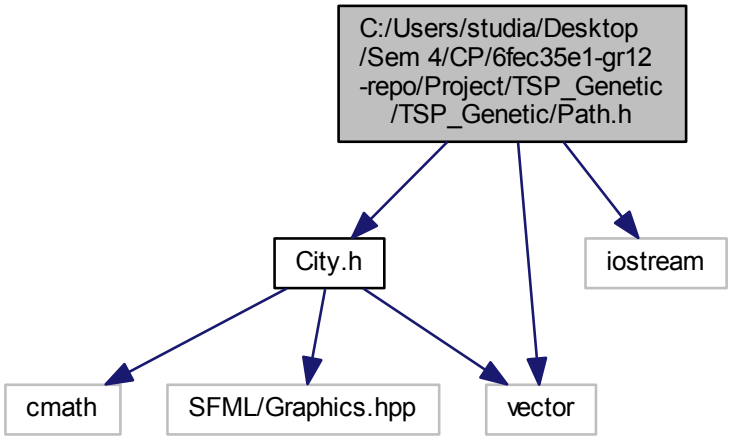
7.23 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Path.h File Reference

Declares the path class.

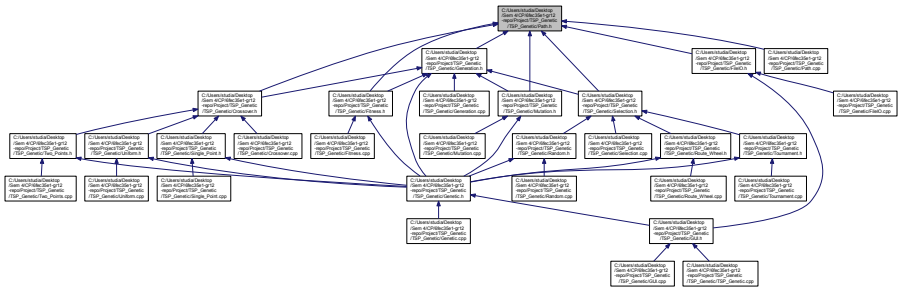
```
#include "City.h"
#include <vector>
```

```
#include <iostream>
```

Include dependency graph for Path.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Path

  Represents the path.

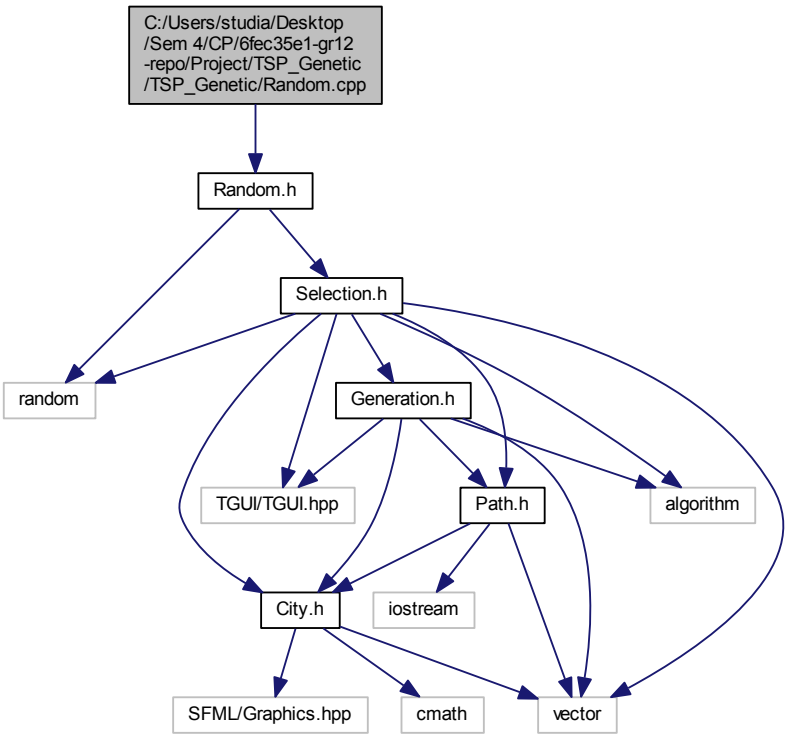
7.23.1 Detailed Description

Declares the path class.

7.24 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Random.cpp File Reference

```
#include "Random.h"
```

Include dependency graph for Random.cpp:

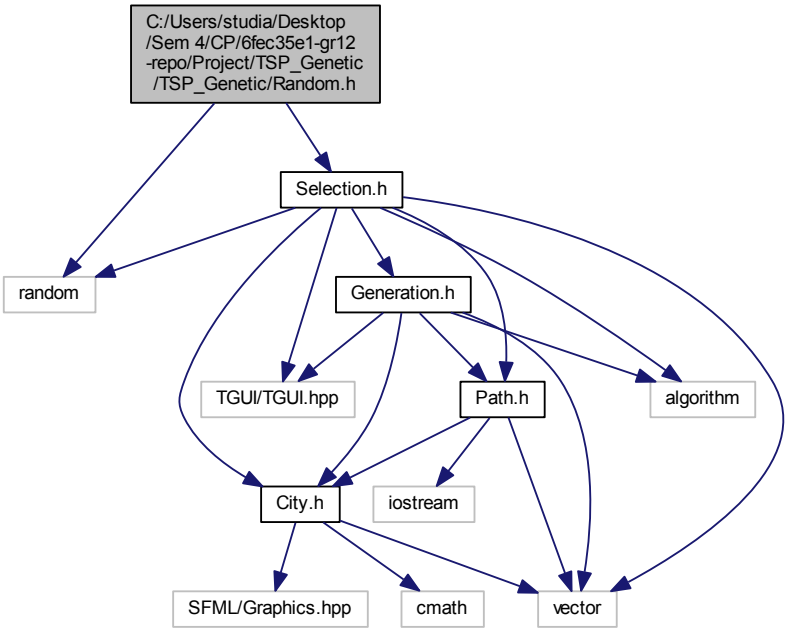


7.25 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Random.h File Reference

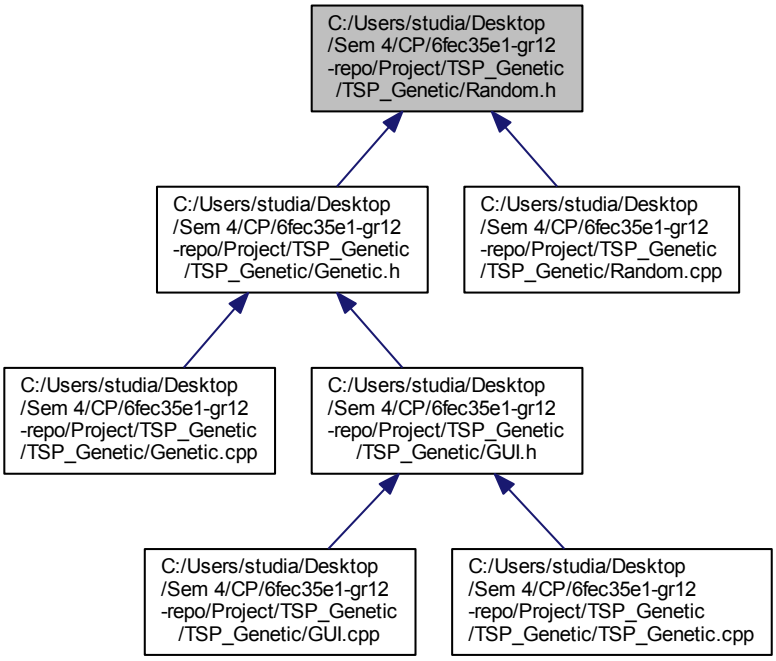
Declares the random class.

```
#include <random>
#include "Selection.h"
```

Include dependency graph for Random.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Random

  Concrete selection strategy class. Implements the Selection strategy.

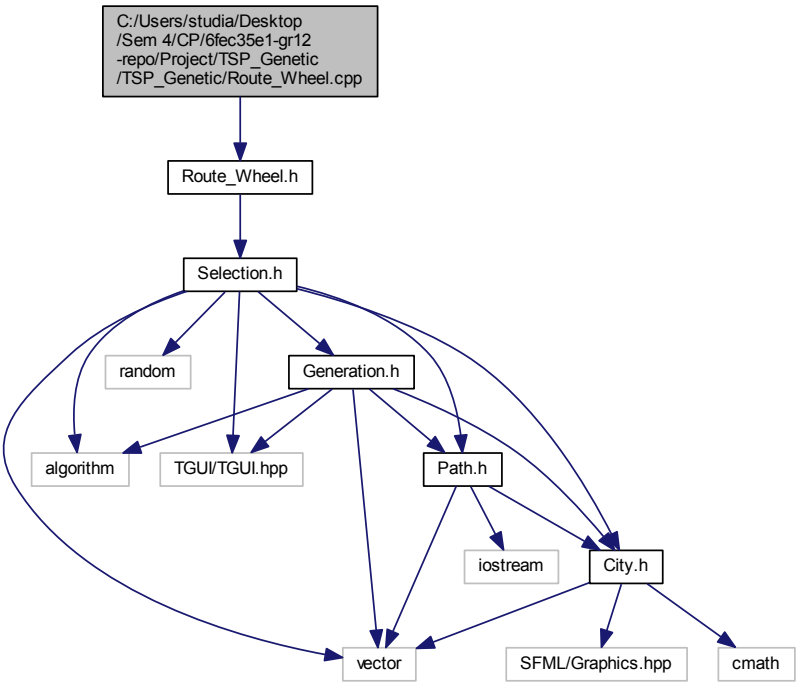
7.25.1 Detailed Description

Declares the random class.

7.26 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Route_Wheel.cpp File Reference

```
#include "Route_Wheel.h"
```

Include dependency graph for Route_Wheel.cpp:

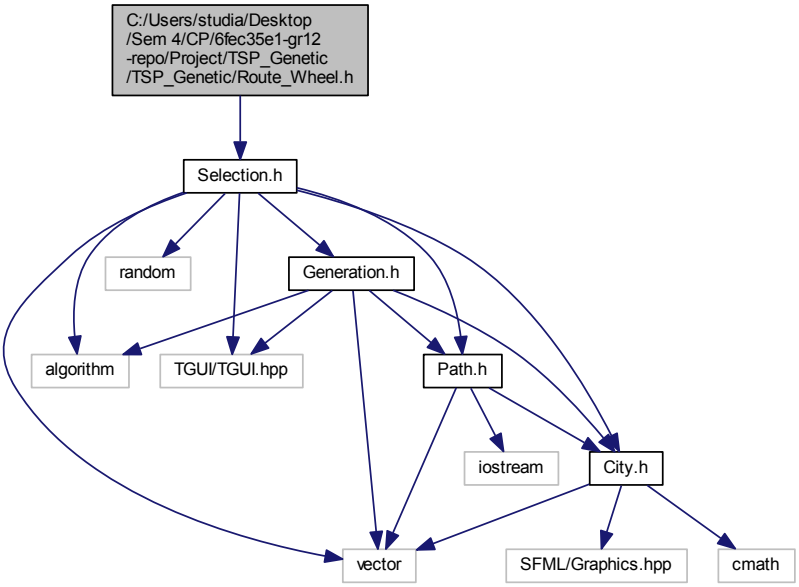


7.27 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Route_Wheel.h File Reference

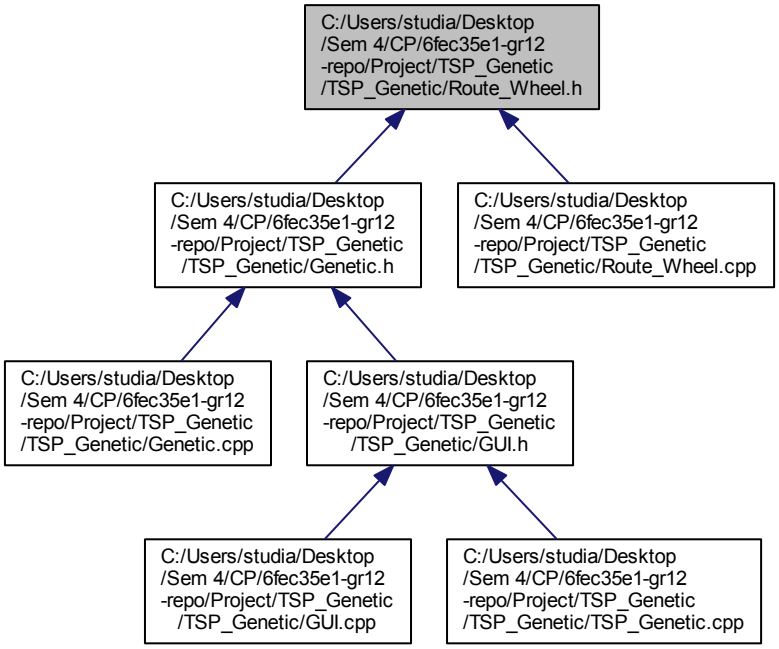
Declares the route wheel class.

```
#include "Selection.h"
```

Include dependency graph for Route_Wheel.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Route_Wheel

  Concrete selection strategy class. Implements the Selection strategy.

7.27.1 Detailed Description

Declares the route wheel class.

7.28 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/sample.txt File Reference

Variables

- COORD

7.28.1 Variable Documentation

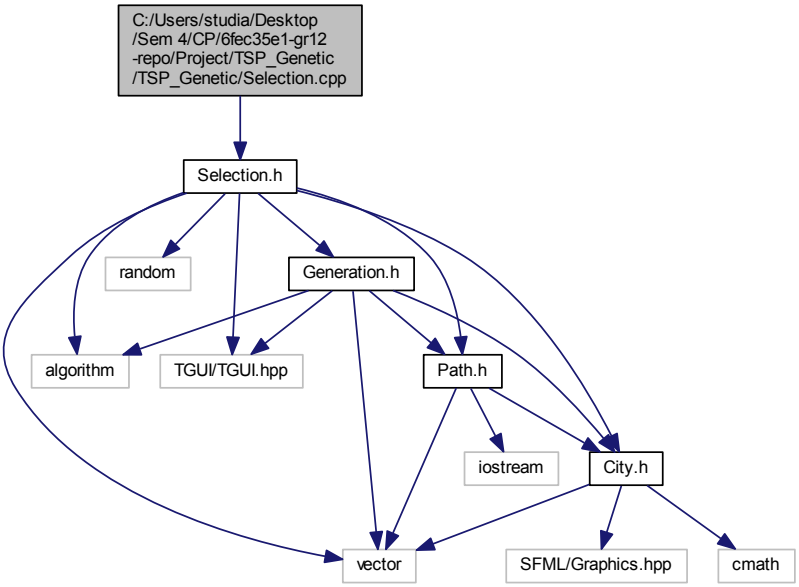
7.28.1.1 COORD

COORD

7.29 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Selection.cpp File Reference

```
#include "Selection.h"
```

Include dependency graph for Selection.cpp:

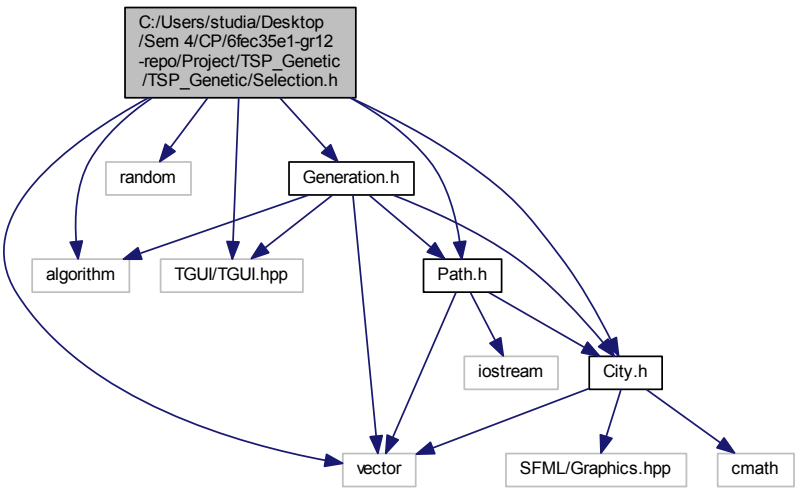


7.30 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Selection.h File Reference

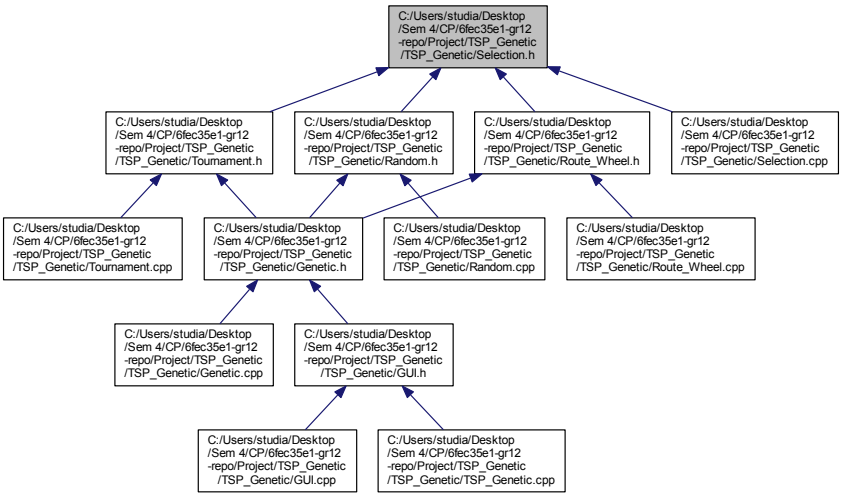
Declares the selection strategy interface class.

```
#include <vector>
#include <algorithm>
#include <random>
#include <TGUI/TGUI.hpp>
#include "Generation.h"
#include "City.h"
#include "Path.h"
```

Include dependency graph for Selection.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Selection

  Abstract class for selection strategy interface.

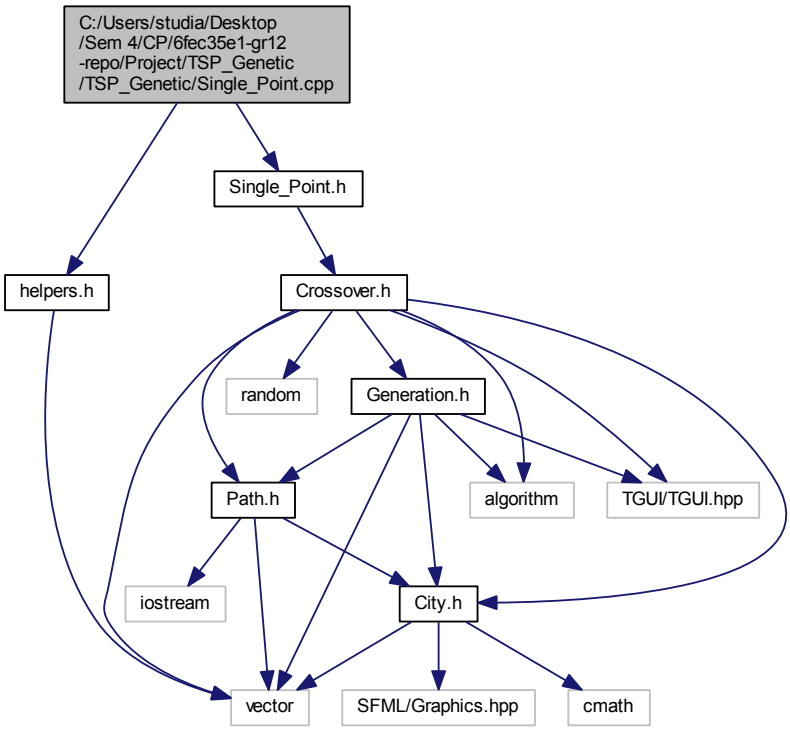
7.30.1 Detailed Description

Declares the selection strategy interface class.

7.31 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Single_Point.cpp File Reference

```
#include "Single_Point.h"
#include "helpers.h"
```

Include dependency graph for Single_Point.cpp:

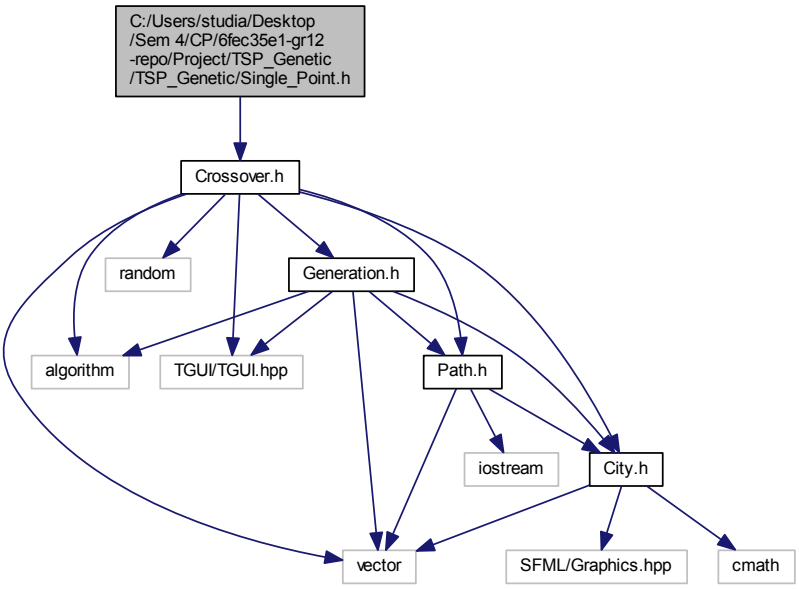


7.32 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Single_Point.h File Reference

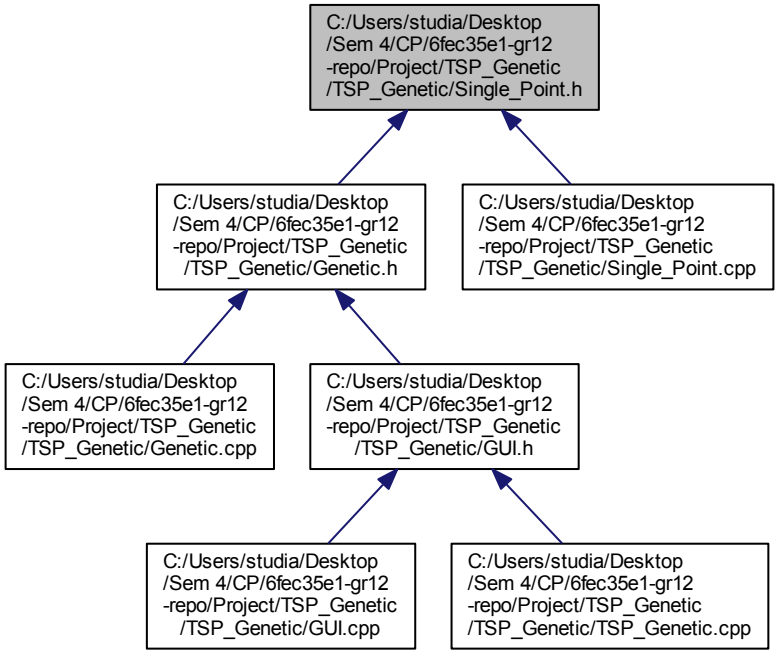
Declares the single point Crossover class.

```
#include "Crossover.h"
```

Include dependency graph for Single_Point.h:



This graph shows which files directly or indirectly include this file:



Classes

- class SinglePoint

  Concrete crossover strategy class. Implements the Crossover strategy.

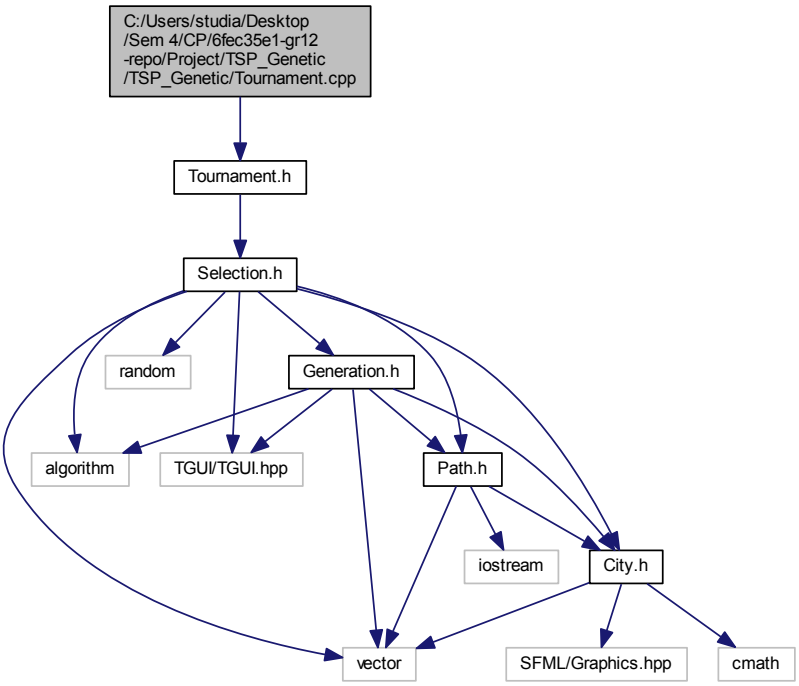
7.32.1 Detailed Description

Declares the single point Crossover class.

7.33 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Tournament.cpp File Reference

```
#include "Tournament.h"
```

Include dependency graph for Tournament.cpp:

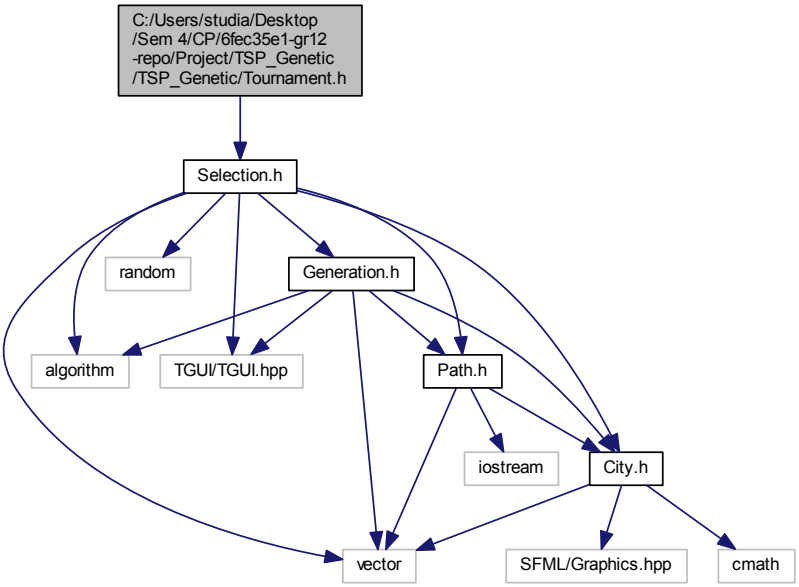


7.34 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Tournament.h File Reference

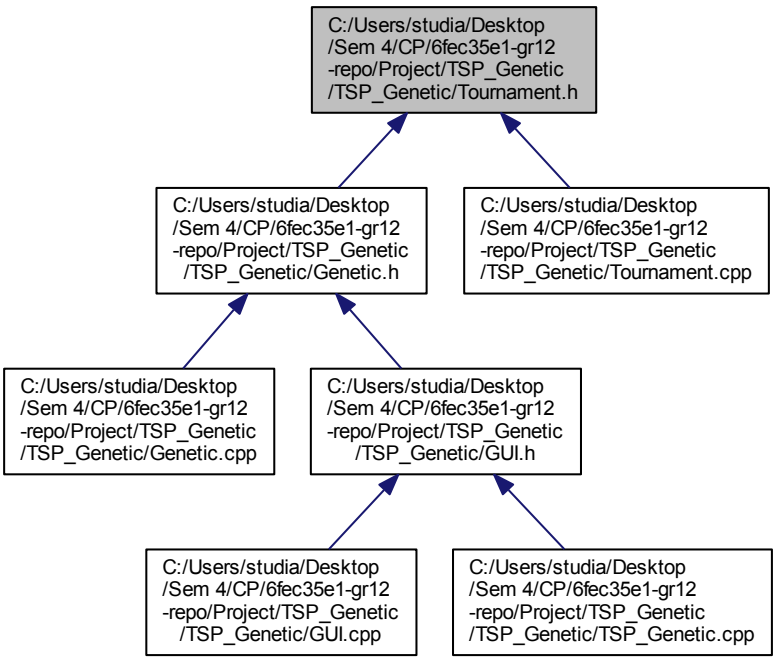
Declares the tournament class.

```
#include "Selection.h"
```

Include dependency graph for Tournament.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Tournament

  Concrete selection strategy class. Implements the Selection strategy.

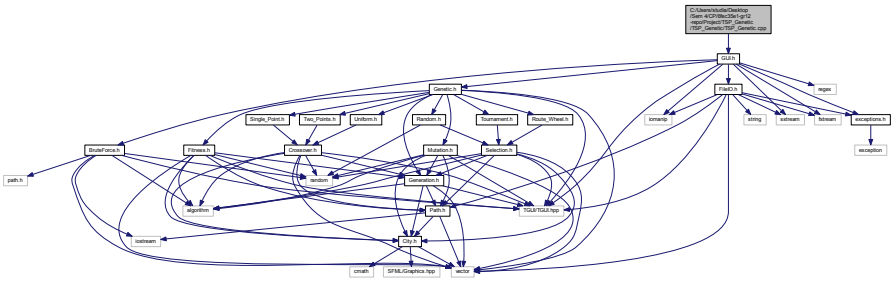
7.34.1 Detailed Description

Declares the tournament class.

7.35 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/TSP_Genetic.cpp File Reference

```
#include "GUI.h"
```

Include dependency graph for TSP_Genetic.cpp:



Functions

- int main ()

7.35.1 Function Documentation

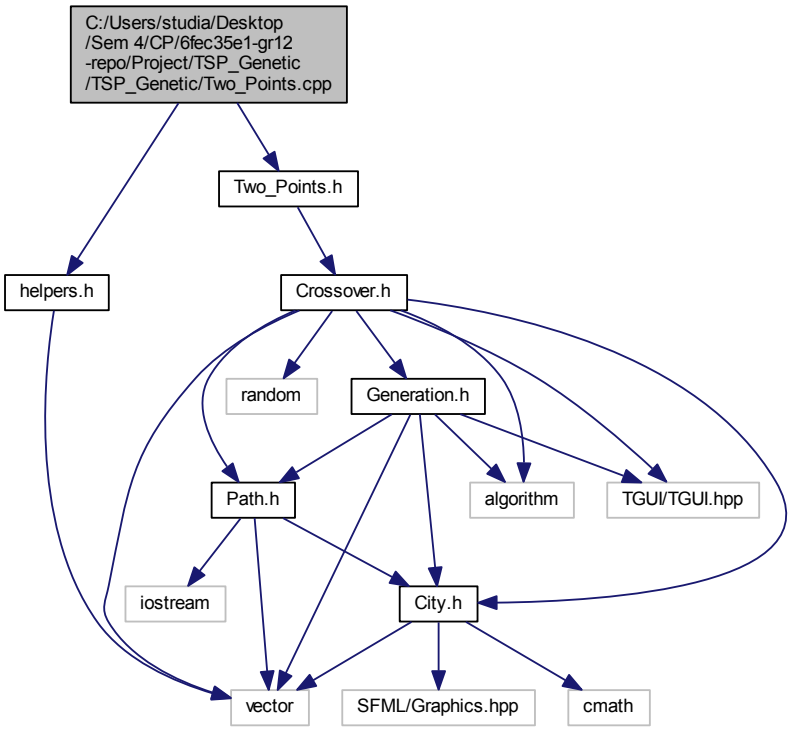
7.35.1.1 main()

```
int main ( )
```

7.36 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Two_Points.cpp File Reference

```
#include "Two_Points.h"
#include "helpers.h"
```

Include dependency graph for Two_Points.cpp:

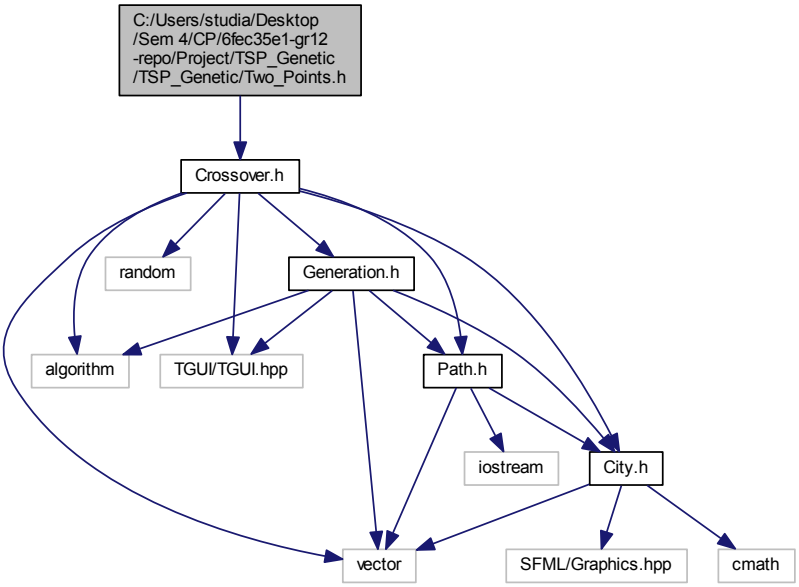


7.37 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Two_Points.h File Reference

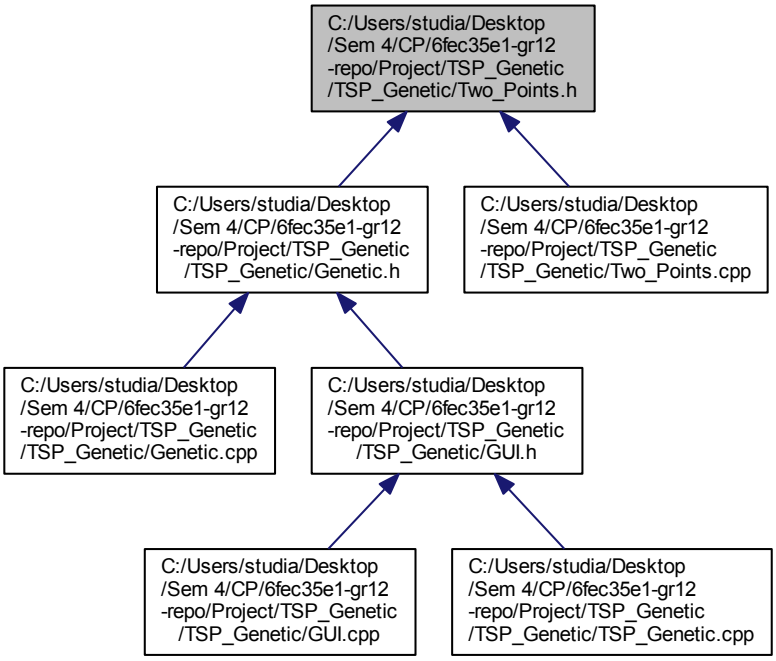
Declares the two points Crossover class.

```
#include "Crossover.h"
```

Include dependency graph for Two_Points.h:



This graph shows which files directly or indirectly include this file:



Classes

- class TwoPoints

  Concrete crossover strategy class. Implements the Crossover strategy.

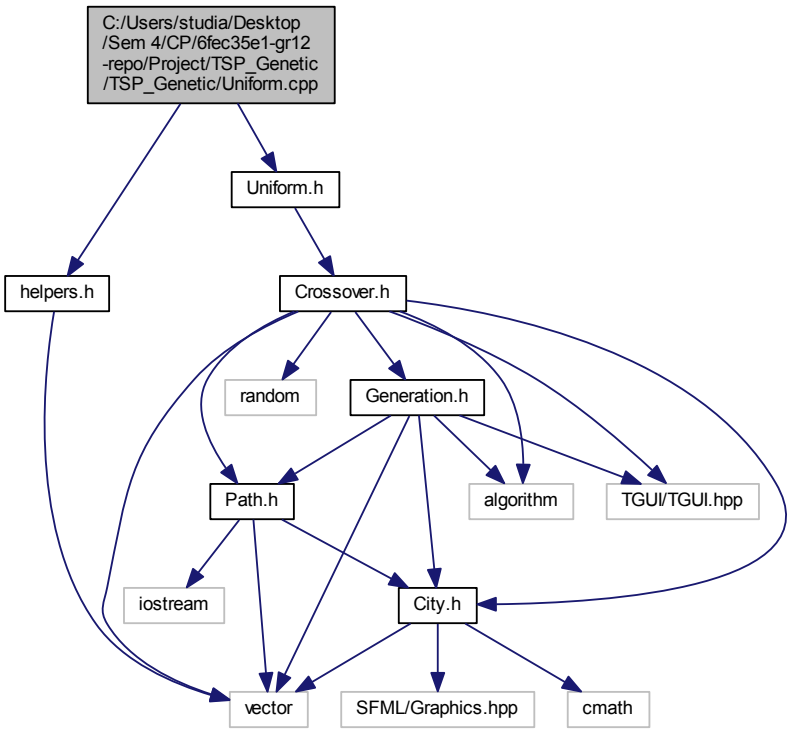
7.37.1 Detailed Description

Declares the two points Crossover class.

7.38 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Uniform.cpp File Reference

```
#include "Uniform.h"
#include "helpers.h"
```

Include dependency graph for Uniform.cpp:

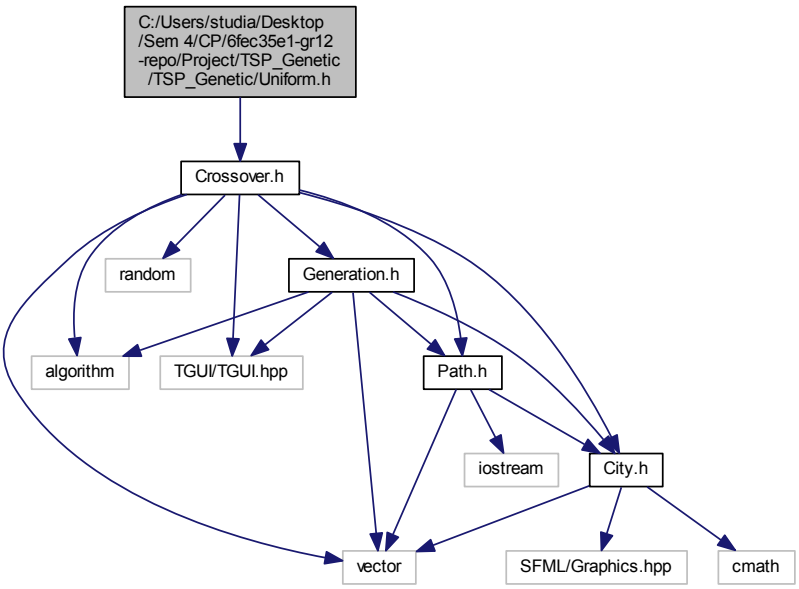


7.39 C:/Users/studia/Desktop/Sem 4/CP/6fec35e1-gr12-repo/Project/TSP_Genetic/TSP_Genetic/Uniform.h File Reference

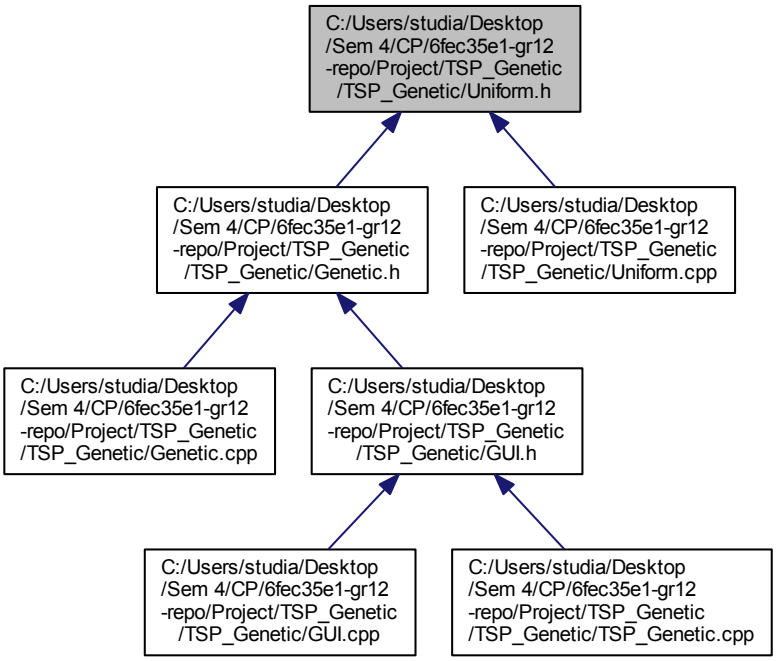
Declares the uniform Crossover class.

```
#include "Crossover.h"
```

Include dependency graph for Uniform.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Uniform

  Concrete crossover strategy class. Implements the Crossover strategy.

7.39.1 Detailed Description

Declares the uniform Crossover class.