

Drogi

Wygenerowano przez Doxygen 1.8.20



<b>1 Indeks klas</b>	<b>1</b>
1.1 Lista klas	1
<b>2 Indeks plików</b>	<b>3</b>
2.1 Lista plików	3
<b>3 Dokumentacja klas</b>	<b>5</b>
3.1 Dokumentacja struktury wezel	5
3.1.1 Opis szczegółowy	5
3.1.2 Dokumentacja atrybutów składowych	5
3.1.2.1 cena_od_miasta_poczatkowego	5
3.1.2.2 najtansze_polaczenie_z_wezla	6
3.1.2.3 nazwa	6
3.1.2.4 odwiedzono	6
3.1.2.5 polaczenia	6
<b>4 Dokumentacja plików</b>	<b>7</b>
4.1 Dokumentacja pliku funkcje.cpp	7
4.1.1 Dokumentacja funkcji	7
4.1.1.1 algorytm()	7
4.1.1.2 czytanie_z_wiersza_polecen()	8
4.1.1.3 przygotuj_dane()	8
4.1.1.4 relaksacja_krawedzi()	9
4.1.1.5 zczytaj_z_pliku()	9
4.1.1.6 znajdz_najblizsze_miasto()	10
4.1.1.7 zwroc_do_pliku()	10
4.1.1.8 zwroc_miasto()	10
4.2 Dokumentacja pliku funkcje.h	11
4.2.1 Dokumentacja funkcji	11
4.2.1.1 algorytm()	11
4.2.1.2 czytanie_z_wiersza_polecen()	12
4.2.1.3 przygotuj_dane()	12
4.2.1.4 relaksacja_krawedzi()	13
4.2.1.5 zczytaj_z_pliku()	13
4.2.1.6 znajdz_najblizsze_miasto()	14
4.2.1.7 zwroc_do_pliku()	14
4.2.1.8 zwroc_miasto()	14
4.3 Dokumentacja pliku infastruktura_drog.cpp	15
4.3.1 Dokumentacja funkcji	15
4.3.1.1 main()	15
4.4 Dokumentacja pliku struktury.h	15
<b>Indeks</b>	<b>17</b>



# Rozdział 1

## Indeks klas

### 1.1 Lista klas

Tutaj znajdują się klasy, struktury, unie i interfejsy wraz z ich krótkimi opisami:

<a href="#">wezel</a> . . . . .	5
---------------------------------	---



## Rozdział 2

# Indeks plików

### 2.1 Lista plików

Tutaj znajduje się lista wszystkich plików z ich krótkimi opisami:

<a href="#">funkcje.cpp</a>	7
<a href="#">funkcje.h</a>	11
<a href="#">infrastruktura_drog.cpp</a>	15
<a href="#">struktury.h</a>	15





## Rozdział 3

# Dokumentacja klas

### 3.1 Dokumentacja struktury wezel

```
#include <struktury.h>
```

#### Atrybuty publiczne

- `std::string nazwa`
- `std::map< wezel *, double > polaczenia`
- `double cena_od_miasta_poczatkowego`
- `bool odwiedzono = false`
- `std::pair< wezel *, double > najtansze_polaczenie_z_wezla`

#### 3.1.1 Opis szczegółowy

Struktora, która ilustruje miasto węzłowe (pojedyncze miasto) i wszystkie możliwe połączenia z nim oraz ceny.

Struktura przechowuje nazwę miasta węzłowego i wszystkie możliwe połączenia w postaci mapy (listy par) miast, z którymi miasto węzłowe jest połączone oraz ceny tych połączeń). Struktura ta później jest wykorzystywana przy algorytmie.

#### 3.1.2 Dokumentacja atrybutów składowych

##### 3.1.2.1 cena\_od\_miasta\_poczatkowego

```
double wezel::cena_od_miasta_poczatkowego
```

Zmienna, która jest wykorzystywana przy algorytmie, jest to cena od miasta do węzła

### 3.1.2.2 najtansze\_polaczenie\_z\_wezla

```
std::pair<wezel*, double> wezel::najtansze_polaczenie_z_wezla
```

Zmienna, która przechowuje parę, która przechowuje najtańsze połączenie

### 3.1.2.3 nazwa

```
std::string wezel::nazwa
```

Nazwa miasta węzłowego

### 3.1.2.4 odwiedzono

```
bool wezel::odwiedzono = false
```

Zmienna, która służy do sprawdzenia czy dane miasto zostało już odwiedzone

### 3.1.2.5 polaczenia

```
std::map<wezel*, double> wezel::polaczenia
```

Mapa, która przechowuje listę par, w tym przypadku para to nazwa miasta, z którym miasto węzłowe jest połączone oraz cena połączenia tych miast

Dokumentacja dla tej struktury została wygenerowana z pliku:

- [struktury.h](#)

## Rozdział 4

# Dokumentacja plików

### 4.1 Dokumentacja pliku funkcje.cpp

```
#include "funkcje.h"
#include "struktury.h"
#include <limits.h>
```

#### Funkcje

- bool `czytanie_z_wiersza_polecen` (int arg, char \*argv[], string &nazwa\_pliku\_wejsciowego, string &nazwa\_pliku\_wyjsciowego)
- `wezel * zwroc_miasto` (string &nazwa\_szukanego\_miasta, vector< `wezel` \* > &miasta\_wezlowe)
- vector< `wezel` \* > `zczytaj_z_pliku` (string &nazwa)
- void `przygotuj_dane` (vector< `wezel` \* > &lista\_miast)
- void `relaksacja_krawedzi` (`wezel` \*miasto, `wezel` \*miasto\_poczatkowe)
- `wezel * znajdz_najblizsze_miasto` (`wezel` \*obecny\_wezel, vector< `wezel` \* > &lista\_miast)
- void `algorytm` (vector< `wezel` \* > &lista\_miast, `wezel` \*miasto\_poczatkowe)
- void `zwroc_do_pliku` (vector< `wezel` \* > &lista\_miast, string &nazwa\_pliku\_wyjsciowego)

#### 4.1.1 Dokumentacja funkcji

##### 4.1.1.1 algorytm()

```
void algorytm (
    vector< wezel * > & lista_miast,
    wezel * miasto_poczatkowe )
```

Funkcja wykorzystująca algorytm Dijkstry.

Wykorzystuje funkcję `przygotuj_dane()`, `relaksacja_krawedzi()` oraz `znajdz_najblizsze_miasto()`. Algorytm działa, dla każdego miasta i jak wykryje mniejszą cenę to zapisuje z jakiego połączenia ona pochodzi.

## Parametry

<i>lista_miast</i>	zmienna typu <code>vector&lt;wezel*&gt;</code> , która zawiera listę miast
<i>miasto_pocatkowe</i>	zmienna typu wskaźnik na wezel

## Zwraca

zwraca miasto, z którym połączenie jest najtańsze.

## 4.1.1.2 czytanie\_z\_wiersza\_polecen()

```
bool czytanie_z_wiersza_polecen (
    int arg,
    char * argv[],
    string & nazwa_pliku_wejsciowego,
    string & nazwa_pliku_wyjsciowego )
```

Funkcja, która zajmuje się zczytywaniem parametrów z wiersza poleceń.

Funkcja sprawdza poprawność podanych parametrów i zwraca prawdę (jeśli są poprawne) lub fałsz (jeśli są błędne)

## Parametry

<i>arg</i>	ilość argumentów przekazanych
<i>argv[]</i>	wskaźnik na tablicę parametrów
<i>nazwa_pliku_wejsciowego</i>	do funkcji przekazujemy nazwę pliku tekstowego (wejściowego), zmienna typu string
<i>nazwa_pliku_wyjsciowego</i>	do funkcji przekazujemy nazwę pliku tekstowego (wyjściowego), zmienna typu string

## Zwraca

jeśli miasto już znajduje się na liście to zwraca nullptr, jeśli miasto jeszcze nie znajduje się na liście to zwraca jego nazwę

## 4.1.1.3 przygotuj\_dane()

```
void przygotuj_dane (
    vector< wezel * > & lista_miast )
```

Funkcja, która przygotowuje dane, aby później można było wykorzystać je w algorytmie.

Funkcja ceny (*cena\_od\_misata\_początkowego*) ustawia na nieskończoność i zmienia status odwiedzenia na false. Ustawia im wartość maksymalną dla double.

## Parametry

<i>lista_miast</i>	zmienna typu <code>vector&lt;wezel*&gt;</code> , która zawiera listę miast
--------------------	--

## 4.1.1.4 relaksacja\_krawedzi()

```
void relaksacja_krawedzi (
    wezel * miasto,
    wezel * miasto_poczatkowe )
```

Funkcja służąca do szukania czy istnieje jakieś tańsze połączenie do węzła. Tańsze połączenia są zapamiętywane przez węzeł.

Funkcja sprawdza ceny połączeń z węzła i jeżeli znalazł krótszą trasę to wpisuje tą trasę jako połączenie.

## Parametry

<i>miasto</i>	zmienna typu wskaźnik na wezeł, przechowuje nawę miasta
<i>miasto_poczatkowe</i>	zmienna typu wskaźnik na wezeł, przechowuje nawę miasta początkowego

## 4.1.1.5 zczytaj\_z\_pliku()

```
vector<wezel *> zczytaj_z_pliku (
    string & nazwa )
```

Funkcja służąca do odczytania danych z pliku oraz umieszczenia ich w strukturze. Wykorzystuje funkcję `zwroc_↔` miasto.

Funkcja zczytuje dane z pliku za pomocą stringstream po linijce i sprawdza poprawność danych w danej linijce, jeśli dane są poprawne to sprawdza czy dane miasta istnieją już w strukturze, jeśli nie to dla danego miasta tworzy nowy węzeł

## Parametry

<i>nazwa</i>	to zmienna typu string przechowująca nazwę pliku wejściowego wraz z jego rozszerzeniem
--------------	--

## Zwraca

zwraca zmienną typu `vector<wezel*>` (wektor węzłów), która przechowuje listę miast węzłowych i ich połączenia z innymi miastami.

#### 4.1.1.6 `znajdz_najblizsze_miasto()`

```
wezel* znajdz_najblizsze_miasto (
    wezel * obecny_wezel,
    vector< wezel * > & lista_miast )
```

Funkcja, która sprawdza czy miasto zostało odwiedzone a następnie sprawdza najtańsze połączenia z nim.

Funkcja służąca do szukania najbliższego (najtaniejszego) miasta połączonego z węzłem lub najtańszego połączenia.

##### Parametry

<i>obecny_wezel</i>	zmienna typu wskaźnik na wezel, miasto które obecnie sprawdzamy
<i>lista_miast</i>	zmienna typu <code>vector&lt;wezel*&gt;</code> , która zawiera listę miast

##### Zwraca

zwraca miasto, z którym połączenie jest najtańsze.

#### 4.1.1.7 `zwroc_do_pliku()`

```
void zwroc_do_pliku (
    vector< wezel * > & lista_miast,
    string & nazwa_pliku_wyjsciowego )
```

Funkcja, która zwraca dane do pliku wyjściowego.

##### Parametry

<i>lista_miast</i>	zmienna typu <code>vector&lt;wezel*&gt;</code> , która zawiera listę miast
<i>nazwa_pliku_wyjsciowego</i>	zmienna typu <code>string</code> , która przechowuje nazwę pliku wyjściowego

#### 4.1.1.8 `zwroc_miasto()`

```
wezel* zwroc_miasto (
    string & nazwa_szukanego_miasta,
    vector< wezel * > & miasta_wezlowe )
```

Funkcja służąca do sprawdzenia, czy czytane z pliku miasto istnieje już na liście miast.

Jeśli miasto istnieje już na liście to funkcja zwraca pusty wskaźnik, jeśli miasto jeszcze nie istnieje na liście miast to dopisuje je do listy.

## Parametry

<i>nazwa_szukanego_miasta</i>	to zmienna typu string, która przechwuje nazwę miasta, które czytaliśmy z pliku i przekazujemy do funkcji
<i>miasta_wezlowe</i>	to zmienna typu <code>vector&lt;wezel*&gt;</code> (wektor wskaźników na węzeł), która zawiera listę miast węzłowych, które zostały już czytane z pliku

## Zwraca

jeśli miasto już znajduje się na liście to zwraca jego nazwę, jeśli miasto jeszcze nie znajduje się na liście to zwraca nullptr

## 4.2 Dokumentacja pliku funkcje.h

```
#include <iostream>
#include <fstream>
#include <string.h>
#include <sstream>
#include <vector>
#include "funkcje.h"
#include "struktury.h"
```

### Funkcje

- bool `czytanie_z_wiersza_polecen` (int arg, char \*argv[], string &nazwa\_pliku\_wejscowego, string &nazwa\_pliku\_wyjscowego)
- `wezel * zwroc_miasto` (string &nazwa\_szukanego\_miasta, vector< `wezel` \* > &miasta\_wezlowe)
- vector< `wezel` \* > `zczytaj_z_pliku` (string &nazwa)
- void `przygotuj_dane` (vector< `wezel` \* > &lista\_miast)
- void `relaksacja_krawedzi` (`wezel` \*miasto, `wezel` \*miasto\_pocztkowe)
- `wezel * znajdz_najblizsze_miasto` (`wezel` \*obecny\_wezel, vector< `wezel` \* > &lista\_miast)
- void `algorytm` (vector< `wezel` \* > &lista\_miast, `wezel` \*miasto\_pocztkowe)
- void `zwroc_do_pliku` (vector< `wezel` \* > &lista\_miast, string &nazwa\_pliku\_wyjscowego)

### 4.2.1 Dokumentacja funkcji

#### 4.2.1.1 algorytm()

```
void algorytm (
    vector< wezel * > & lista_miast,
    wezel * miasto_pocztkowe )
```

Funkcja wykorzystująca algorytm Dijkstry.

Wykorzystuje funkcję `przygotuj_dane()`, `relaksacja_krawedzi()` oraz `znajdz_najblizsze_miasto()`. Algorytm działa, dla każdego miasta i jak wykryje mniejszą cenę to zapisuje z jakiego połączenia ona pochodzi.

## Parametry

<i>lista_miast</i>	zmienna typu <code>vector&lt;wezel*&gt;</code> , która zawiera listę miast
<i>miasto_pocatkowe</i>	zmienna typu wskaźnik na wezel

## Zwraca

zwraca miasto, z którym połączenie jest najtańsze.

## 4.2.1.2 czytanie\_z\_wiersza\_polecen()

```
bool czytanie_z_wiersza_polecen (
    int arg,
    char * argv[],
    string & nazwa_pliku_wejsciowego,
    string & nazwa_pliku_wyjsciowego )
```

Funkcja, która zajmuje się zczytywaniem parametrów z wiersza poleceń.

Funkcja sprawdza poprawność podanych parametrów i zwraca prawdę (jeśli są poprawne) lub fałsz (jeśli są błędne)

## Parametry

<i>arg</i>	ilość argumentów przekazanych
<i>argv[]</i>	wskaźnik na tablicę parametrów
<i>nazwa_pliku_wejsciowego</i>	do funkcji przekazujemy nazwę pliku tekstowego (wejściowego), zmienna typu string
<i>nazwa_pliku_wyjsciowego</i>	do funkcji przekazujemy nazwę pliku tekstowego (wyjściowego), zmienna typu string

## Zwraca

jeśli miasto już znajduje się na liście to zwraca nullptr, jeśli miasto jeszcze nie znajduje się na liście to zwraca jego nazwę

## 4.2.1.3 przygotuj\_dane()

```
void przygotuj_dane (
    vector< wezel * > & lista_miast )
```

Funkcja, która przygotowuje dane, aby później można było wykorzystać je w algorytmie.

Funkcja ceny (*cena\_od\_misata\_początkowego*) ustawia na nieskończoność i zmienia status odwiedzenia na false. Ustawia im wartość maksymalną dla double.



## Parametry

<i>lista_miast</i>	zmienna typu <code>vector&lt;wezel*&gt;</code> , która zawiera listę miast
--------------------	--

## 4.2.1.4 relaksacja\_krawedzi()

```
void relaksacja_krawedzi (
    wezel * miasto,
    wezel * miasto_poczatkowe )
```

Funkcja służąca do szukania czy istnieje jakieś tańsze połączenie do węzła. Tańsze połączenia są zapamiętywane przez węzeł.

Funkcja sprawdza ceny połączeń z węzła i jeżeli znalazł krótszą trasę to wpisuje tą trasę jako połączenie.

## Parametry

<i>miasto</i>	zmienna typu wskaźnik na wezeł, przechowuje nawę miasta
<i>miasto_poczatkowe</i>	zmienna typu wskaźnik na wezeł, przechowuje nawę miasta początkowego

## 4.2.1.5 zczytaj\_z\_pliku()

```
vector<wezel *> zczytaj_z_pliku (
    string & nazwa )
```

Funkcja służąca do odczytania danych z pliku oraz umieszczenia ich w strukturze. Wykorzystuje funkcję `zwroc_↔` miasto.

Funkcja zczytuje dane z pliku za pomocą stringstream po linijce i sprawdza poprawność danych w danej linijce, jeśli dane są poprawne to sprawdza czy dane miasta istnieją już w strukturze, jeśli nie to dla danego miasta tworzy nowy węzeł

## Parametry

<i>nazwa</i>	to zmienna typu string przechowująca nazwę pliku wejściowego wraz z jego rozszerzeniem
--------------	--

## Zwraca

zwraca zmienną typu `vector<wezel*>` (wektor węzłów), która przechowuje listę miast węzłowych i ich połączenia z innymi miastami.

#### 4.2.1.6 `znajdz_najblizsze_miasto()`

```
wezel* znajdz_najblizsze_miasto (
    wezel * obecny_wezel,
    vector< wezel * > & lista_miast )
```

Funkcja, która sprawdza czy miasto zostało odwiedzone a następnie sprawdza najtańsze połączenia z nim.

Funkcja służąca do szukania najbliższego (najtaniejszego) miasta połączonego z węzłem lub najtaniejszego połączenia.

##### Parametry

<i>obecny_wezel</i>	zmienna typu wskaźnik na wezel, miasto które obecnie sprawdzamy
<i>lista_miast</i>	zmienna typu <code>vector&lt;wezel*&gt;</code> , która zawiera listę miast

##### Zwraca

zwraca miasto, z którym połączenie jest najtańsze.

#### 4.2.1.7 `zwroc_do_pliku()`

```
void zwroc_do_pliku (
    vector< wezel * > & lista_miast,
    string & nazwa_pliku_wyjsciowego )
```

Funkcja, która zwraca dane do pliku wyjściowego.

##### Parametry

<i>lista_miast</i>	zmienna typu <code>vector&lt;wezel*&gt;</code> , która zawiera listę miast
<i>nazwa_pliku_wyjsciowego</i>	zmienna typu <code>string</code> , która przechowuje nazwę pliku wyjściowego

#### 4.2.1.8 `zwroc_miasto()`

```
wezel* zwroc_miasto (
    string & nazwa_szukanego_miasta,
    vector< wezel * > & miasta_wezlowe )
```

Funkcja służąca do sprawdzenia, czy czytane z pliku miasto istnieje już na liście miast.

Jeśli miasto istnieje już na liście to funkcja zwraca pusty wskaźnik, jeśli miasto jeszcze nie istnieje na liście miast to dopisuje je do listy.

#### Parametry

<i>nazwa_szukanego_miasta</i>	to zmienna typu string, która przechwuje nazwę miasta, które zczytaliśmy z pliku i przekazujemy do funkcji
<i>miasta_wezlowe</i>	to zmienna typu <code>vector&lt;wezel*&gt;</code> (wektor wskaźników na węzeł), która zawiera listę miast węzłowych, które zostały już zczytane z pliku

#### Zwraca

jeśli miasto już znajduje się na liście to zwraca jego nazwę, jeśli miasto jeszcze nie znajduje się na liście to zwraca nullptr

## 4.3 Dokumentacja pliku infastruktura\_drog.cpp

```
#include <iostream>
#include <vector>
#include "funkcje.h"
#include "struktury.h"
```

### Funkcje

- int `main` (int arg, char \*argv[])

#### 4.3.1 Dokumentacja funkcji

##### 4.3.1.1 main()

```
int main (
    int arg,
    char * argv[] )
```

## 4.4 Dokumentacja pliku struktury.h

```
#include <string>
#include <iostream>
#include <vector>
#include <map>
#include <limits.h>
```

### Komponenty

- struct `wezel`



# Indeks

- algorytm
  - [funkcje.cpp, 7](#)
  - [funkcje.h, 11](#)
- cena\_od\_miasta\_poczatkowego
  - [wezel, 5](#)
- czytanie\_z\_wiersza\_polecen
  - [funkcje.cpp, 8](#)
  - [funkcje.h, 12](#)
- funkcje.cpp, 7
  - [algorytm, 7](#)
  - [czytanie\\_z\\_wiersza\\_polecen, 8](#)
  - [przygotuj\\_dane, 8](#)
  - [relaksacja\\_krawedzi, 9](#)
  - [zczytaj\\_z\\_pliku, 9](#)
  - [znajdz\\_najblizsze\\_miasto, 9](#)
  - [zwroc\\_do\\_pliku, 10](#)
  - [zwroc\\_miasto, 10](#)
- funkcje.h, 11
  - [algorytm, 11](#)
  - [czytanie\\_z\\_wiersza\\_polecen, 12](#)
  - [przygotuj\\_dane, 12](#)
  - [relaksacja\\_krawedzi, 13](#)
  - [zczytaj\\_z\\_pliku, 13](#)
  - [znajdz\\_najblizsze\\_miasto, 13](#)
  - [zwroc\\_do\\_pliku, 14](#)
  - [zwroc\\_miasto, 14](#)
- infastruktura\_drog.cpp, 15
  - [main, 15](#)
- main
  - [infastruktura\\_drog.cpp, 15](#)
- najtansze\_polaczenie\_z\_wezla
  - [wezel, 5](#)
- nazwa
  - [wezel, 6](#)
- odwiedzono
  - [wezel, 6](#)
- polaczenia
  - [wezel, 6](#)
- przygotuj\_dane
  - [funkcje.cpp, 8](#)
  - [funkcje.h, 12](#)
- relaksacja\_krawedzi
  - [funkcje.cpp, 9](#)
- [funkcje.h, 13](#)
- struktury.h, 15
- wezel, 5
  - [cena\\_od\\_miasta\\_poczatkowego, 5](#)
  - [najtansze\\_polaczenie\\_z\\_wezla, 5](#)
  - [nazwa, 6](#)
  - [odwiedzono, 6](#)
  - [polaczenia, 6](#)
- zczytaj\_z\_pliku
  - [funkcje.cpp, 9](#)
  - [funkcje.h, 13](#)
- znajdz\_najblizsze\_miasto
  - [funkcje.cpp, 9](#)
  - [funkcje.h, 13](#)
- zwroc\_do\_pliku
  - [funkcje.cpp, 10](#)
  - [funkcje.h, 14](#)
- zwroc\_miasto
  - [funkcje.cpp, 10](#)
  - [funkcje.h, 14](#)