

Huffman

Generated by Doxygen 1.9.3

1 Class Index	1
1.1 Class List	1
2 File Index	3
2.1 File List	3
3 Class Documentation	5
3.1 Node Struct Reference	5
3.1.1 Detailed Description	5
3.1.2 Constructor & Destructor Documentation	5
3.1.2.1 Node() [1/2]	5
3.1.2.2 Node() [2/2]	6
3.1.3 Member Data Documentation	6
3.1.3.1 character	6
3.1.3.2 code	6
3.1.3.3 frequency	6
3.1.3.4 left	6
3.1.3.5 right	7
3.2 NodeComparator Struct Reference	7
3.2.1 Detailed Description	7
3.2.2 Member Function Documentation	7
3.2.2.1 operator()	7
4 File Documentation	9
4.1 functions.cpp File Reference	9
4.1.1 Function Documentation	9
4.1.1.1 calculateCodes()	9
4.1.1.2 compress()	10
4.1.1.3 decode()	10
4.1.1.4 decompress()	10
4.1.1.5 encode()	11
4.1.1.6 loadFrequencyDictionary()	11
4.1.1.7 makeFrequencyDictionary()	12
4.1.1.8 makeHeap()	12
4.1.1.9 saveFrequencyDictionary()	12
4.2 functions.h File Reference	12
4.2.1 Function Documentation	13
4.2.1.1 calculateCodes()	13
4.2.1.2 compress()	13
4.2.1.3 decode()	14
4.2.1.4 decompress()	14
4.2.1.5 encode()	15
4.2.1.6 loadFrequencyDictionary()	15

4.2.1.7 makeFrequencyDictionary()	15
4.2.1.8 makeHeap()	16
4.2.1.9 saveFrequencyDictionary()	16
4.3 functions.h	16
4.4 main.cpp File Reference	17
4.4.1 Function Documentation	17
4.4.1.1 main()	17
4.4.1.2 printHelp()	17
4.5 node.cpp File Reference	17
4.6 node.h File Reference	18
4.7 node.h	18
4.8 utils.cpp File Reference	18
4.8.1 Function Documentation	18
4.8.1.1 getFileContent()	18
4.9 utils.h File Reference	19
4.9.1 Function Documentation	19
4.9.1.1 getFileContent()	19
4.10 utils.h	19
Index	21

Chapter 1

Class Index

1.1 Class List

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

Node	5
NodeComparator	7

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

functions.cpp	9
functions.h	12
main.cpp	17
node.cpp	17
node.h	18
utils.cpp	18
utils.h	19

Chapter 3

Class Documentation

3.1 Node Struct Reference

```
#include <node.h>
```

Public Member Functions

- `Node` (int `frequency`)
- `Node` (int `frequency`, char `character`)

Public Attributes

- int `frequency`
- char `character`
- std::string `code`
- `Node * left` = NULL
- `Node * right` = NULL

3.1.1 Detailed Description

Struktura ta to wezel przechowujacy informacje o czestotliwosci okreslonego znaku oraz jego kodu, stanowi element drzewa Huffmana.

3.1.2 Constructor & Destructor Documentation

3.1.2.1 `Node()` [1/2]

```
Node::Node (
    int frequency )
```

Domyslny konstruktor wezla pustego.

Parameters

<i>frequency</i>	czestotliwosc wystepowania znakow
------------------	-----------------------------------

3.1.2.2 Node() [2/2]

```
Node::Node (
    int frequency,
    char character )
```

Domyslny konstruktor wezla ze znakiem.

Parameters

<i>frequency</i>	czestotliwosc wystepowania znaku
<i>character</i>	znak dla jakiego jest przypisana czestotliwosc

3.1.3 Member Data Documentation**3.1.3.1 character**

```
char Node::character
```

3.1.3.2 code

```
std::string Node::code
```

3.1.3.3 frequency

```
int Node::frequency
```

3.1.3.4 left

```
Node* Node::left = NULL
```

3.1.3.5 right

```
Node* Node::right = NULL
```

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

- [node.h](#)
- [node.cpp](#)

3.2 NodeComparator Struct Reference

```
#include <node.h>
```

Public Member Functions

- `bool operator()` (const [Node](#) *leftNode, const [Node](#) *rightNode)

3.2.1 Detailed Description

Struktura ta sluzi jedynie do porownywania ze soba dwoch wezlow struktury '[Node](#)', jej implementacja jest wymagana przez `std::priority_queue`.

3.2.2 Member Function Documentation

3.2.2.1 operator()()

```
bool NodeComparator::operator() (
    const Node * leftNode,
    const Node * rightNode )
```

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

- [node.h](#)
- [node.cpp](#)

Chapter 4

File Documentation

4.1 functions.cpp File Reference

```
#include <iostream>
#include <fstream>
#include "functions.h"
#include "utils.h"
```

Functions

- void [makeFrequencyDictionary](#) (const std::string &text, std::map< char, int > &frequency)
- void [loadFrequencyDictionary](#) (const std::string &dict_filepath, std::map< char, int > &frequency)
- void [saveFrequencyDictionary](#) (const std::string &dict_filepath, const std::map< char, int > &frequency)
- void [makeHeap](#) (std::priority_queue< [Node](#) *, std::vector< [Node](#) * >, [NodeComparator](#) > &minBinaryHeap, const std::map< char, int > &frequency)
- void [calculateCodes](#) ([Node](#) *root, std::map< char, std::string > &dictionary, std::string code)
- void [encode](#) (const std::string &input_filepath, const std::string &output_filepath, std::map< char, std::string > &dictionary)
- void [compress](#) (const std::string &input_filepath, const std::string &output_filepath, const std::string &dict_↵
filepath)
- void [decompress](#) (const std::string &input_filepath, const std::string &output_filepath, const std::string &dict_↵
filepath)
- void [decode](#) (const std::string &input_filepath, const std::string &output_filepath, std::map< char, std::string > &dictionary, std::map< char, int > &frequency)

4.1.1 Function Documentation

4.1.1.1 calculateCodes()

```
void calculateCodes (
    Node * root,
    std::map< char, std::string > & dictionary,
    std::string code = "" )
```

Oblicza kody dla kazdego unikatowego znaku przemieszczajac sie po drzewie Huffmana okreslonym w zmiennej 'minBinaryHeap'.

Parameters

<i>root</i>	wskaznik na koniec kolejki priorytetowej
<i>dictionary</i>	referencja do slownika
<i>code</i>	argument opcjonalny do wewnetrznego zastosowania

4.1.1.2 compress()

```
void compress (
    const std::string & input_filepath,
    const std::string & output_filepath,
    const std::string & dict_filepath )
```

Interfejs sluzacy do kompresji pliku wejscowego do wyjscowego i zapisania pliku slownikowego za pomoca algorytmu Huffmana.

Parameters

<i>input_filepath</i>	sciezka do pliku wejscowego
<i>output_filepath</i>	sciezka do pliku wyjscowego
<i>dict_filepath</i>	sciezka do pliku slownikowego

4.1.1.3 decode()

```
void decode (
    const std::string & input_filepath,
    const std::string & output_filepath,
    std::map< char, std::string > & dictionary,
    std::map< char, int > & frequency )
```

Odkodowuje plik ze sciezki okreslonej w zmiennej klasowej 'input_filepath' do sciezki wyjscowej 'output_filepath' funkcja ta jedynie opakowuje funkcje prywatne do prostej uzywalnej formy.

Parameters

<i>input_filepath</i>	sciezka do pliku wejscowego
<i>output_filepath</i>	sciezka do pliku wyjscowego
<i>dictionary</i>	referencja do slownika
<i>frequency</i>	referencja do frekwencji wystepowania znakow

4.1.1.4 decompress()

```
void decompress (
```

```
const std::string & input_filepath,  
const std::string & output_filepath,  
const std::string & dict_filepath )
```

Interfejs sluzacy do dekompresji pliku wejscowego do wyjscowego przy uzyciu okreslonego slownika.

Parameters

<i>input_filepath</i>	sciezka do pliku wejscowego
<i>output_filepath</i>	sciezka do pliku wyjscowego
<i>dict_filepath</i>	sciezka do pliku slownikowego

4.1.1.5 encode()

```
void encode (  
    const std::string & input_filepath,  
    const std::string & output_filepath,  
    std::map< char, std::string > & dictionary )
```

Koduje plik ze sciezki okreslonej w zmiennej klasowej 'input_filepath' do sciezki wyjscowej 'output_filepath', funkcja ta jedynie opakowuje funkcje do prostej uzywalnej formy.

Parameters

<i>input_filepath</i>	sciezka do pliku wejscowego
<i>output_filepath</i>	sciezka do pliku wyjscowego
<i>dictionary</i>	referencja do slownika

4.1.1.6 loadFrequencyDictionary()

```
void loadFrequencyDictionary (  
    const std::string & dict_filepath,  
    std::map< char, int > & frequency )
```

Laduje slownik unikatowych znakow z pliku, sciezka do pliku jest okreslona w zmiennej 'dict_filepath' klasy.

Parameters

<i>dict_filepath</i>	sciezka do pliku slownikowego
<i>frequency</i>	referencja do frekwencji wystepowania znakow

4.1.1.7 makeFrequencyDictionary()

```
void makeFrequencyDictionary (
    const std::string & text,
    std::map< char, int > & frequency )
```

Tworzy słownik unikatowych znaków na podstawie przesłanego tekstu oraz zlicza ich wystąpienie.

Parameters

<i>text</i>	referencja do tekstu do wyszukania unikatowych znaków
<i>frequency</i>	referencja do frekwencji występowania znaków

4.1.1.8 makeHeap()

```
void makeHeap (
    std::priority_queue< Node *, std::vector< Node * >, NodeComparator > & minHeap,
    BinaryHeap,
    const std::map< char, int > & frequency )
```

Tworzy drzewo Huffmana w postaci kolejki priorytetowej <priority_queue> oraz zapełnia je danymi ze słownika.

Parameters

<i>minBinaryHeap</i>	referencja do kolejki priorytetowej
<i>frequency</i>	referencja do frekwencji występowania znaków

4.1.1.9 saveFrequencyDictionary()

```
void saveFrequencyDictionary (
    const std::string & dict_filepath,
    const std::map< char, int > & frequency )
```

Zapisuje słownik unikatowych znaków do pliku, ścieżka do pliku jest określona w zmiennej 'dict_filepath' klasy.

Parameters

<i>dict_filepath</i>	ścieżka do pliku słownikowego
<i>frequency</i>	referencja do frekwencji występowania znaków

4.2 functions.h File Reference

```
#include <string>
```



```
#include <map>
#include <queue>
#include <vector>
#include "node.h"
```

Functions

- void [makeFrequencyDictionary](#) (const std::string &text, std::map< char, int > &frequency)
- void [loadFrequencyDictionary](#) (const std::string &dict_filepath, std::map< char, int > &frequency)
- void [saveFrequencyDictionary](#) (const std::string &dict_filepath, const std::map< char, int > &frequency)
- void [makeHeap](#) (std::priority_queue< [Node](#) *, std::vector< [Node](#) * >, [NodeComparator](#) > &minBinaryHeap, const std::map< char, int > &frequency)
- void [calculateCodes](#) ([Node](#) *root, std::map< char, std::string > &dictionary, std::string code="")
- void [encode](#) (const std::string &input_filepath, const std::string &output_filepath, std::map< char, std::string > &dictionary)
- void [decode](#) (const std::string &input_filepath, const std::string &output_filepath, std::map< char, std::string > &dictionary, std::map< char, int > &frequency)
- void [compress](#) (const std::string &input_filepath, const std::string &output_filepath, const std::string &dict_↵
filepath)
- void [decompress](#) (const std::string &input_filepath, const std::string &output_filepath, const std::string &dict_↵
filepath)

4.2.1 Function Documentation

4.2.1.1 calculateCodes()

```
void calculateCodes (
    Node * root,
    std::map< char, std::string > & dictionary,
    std::string code = "" )
```

Oblicza kody dla kazdego unikatowego znaku przemieszczajac sie po drzewie Huffmana okreslonym w zmiennej 'minBinaryHeap'.

Parameters

<i>root</i>	wskaznik na koniec kolejki priorytetowej
<i>dictionary</i>	referencja do slownika
<i>code</i>	argument opcjonalny do wewnetrznego zastosowania

4.2.1.2 compress()

```
void compress (
    const std::string & input_filepath,
```

```
const std::string & output_filepath,  
const std::string & dict_filepath )
```

Interfejs sluzacy do kompresji pliku wejscowego do wyjscowego i zapisania pliku slownikowego za pomoca algorytmu Huffmana.

Parameters

<i>input_filepath</i>	sciezka do pliku wejscowego
<i>output_filepath</i>	sciezka do pliku wyjscowego
<i>dict_filepath</i>	sciezka do pliku slownikowego

4.2.1.3 decode()

```
void decode (  
    const std::string & input_filepath,  
    const std::string & output_filepath,  
    std::map< char, std::string > & dictionary,  
    std::map< char, int > & frequency )
```

Odkodowuje plik ze sciezki okreslonej w zmiennej klasowej 'input_filepath' do sciezki wyjscowej 'output_filepath' funkcja ta jedynie opakowuje funkcje prywatne do prostej uzywalnej formy.

Parameters

<i>input_filepath</i>	sciezka do pliku wejscowego
<i>output_filepath</i>	sciezka do pliku wyjscowego
<i>dictionary</i>	referencja do slownika
<i>frequency</i>	referencja do frekwencji wystepowania znakow

4.2.1.4 decompress()

```
void decompress (  
    const std::string & input_filepath,  
    const std::string & output_filepath,  
    const std::string & dict_filepath )
```

Interfejs sluzacy do dekompresji pliku wejscowego do wyjscowego przy uzyciu okreslonego slownika.

Parameters

<i>input_filepath</i>	sciezka do pliku wejscowego
<i>output_filepath</i>	sciezka do pliku wyjscowego
<i>dict_filepath</i>	sciezka do pliku slownikowego

4.2.1.5 encode()

```
void encode (
    const std::string & input_filepath,
    const std::string & output_filepath,
    std::map< char, std::string > & dictionary )
```

Koduje plik ze ścieżki określonej w zmiennej klasowej 'input_filepath' do ścieżki wyjściowej 'output_filepath', funkcja ta jedynie opakowuje funkcje do prostej używalnej formy.

Parameters

<i>input_filepath</i>	ścieżka do pliku wejściowego
<i>output_filepath</i>	ścieżka do pliku wyjściowego
<i>dictionary</i>	referencja do słownika

4.2.1.6 loadFrequencyDictionary()

```
void loadFrequencyDictionary (
    const std::string & dict_filepath,
    std::map< char, int > & frequency )
```

Laduje słownik unikatowych znaków z pliku, ścieżka do pliku jest określona w zmiennej 'dict_filepath' klasy.

Parameters

<i>dict_filepath</i>	ścieżka do pliku słownikowego
<i>frequency</i>	referencja do frekwencji występowania znaków

4.2.1.7 makeFrequencyDictionary()

```
void makeFrequencyDictionary (
    const std::string & text,
    std::map< char, int > & frequency )
```

Tworzy słownik unikatowych znaków na podstawie przesłanego tekstu oraz zlicza ich wystąpienie.

Parameters

<i>text</i>	referencja do tekstu do wyszukania unikatowych znaków
<i>frequency</i>	referencja do frekwencji występowania znaków

4.2.1.8 makeHeap()

```
void makeHeap (
    std::priority_queue< Node *, std::vector< Node * >, NodeComparator > & minBinaryHeap,
    const std::map< char, int > & frequency )
```

Tworzy drzewo Huffmana w postaci kolejki priorytetowej <priority_queue> oraz zapelnia je danymi ze slownika.

Parameters

<i>minBinaryHeap</i>	referencja do kolejki priorytetowej
<i>frequency</i>	referencja do frekwencji wystepowania znakow

4.2.1.9 saveFrequencyDictionary()

```
void saveFrequencyDictionary (
    const std::string & dict_filepath,
    const std::map< char, int > & frequency )
```

Zapisuje slownik unikatowych znakow do pliku, sciezka do pliku jest okreslona w zmiennej 'dict_filepath' klasy.

Parameters

<i>dict_filepath</i>	sciezka do pliku slownikowego
<i>frequency</i>	referencja do frekwencji wystepowania znakow

4.3 functions.h

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

```
1 #ifndef FUNCTIONS_H
2 #define FUNCTIONS_H
3
4 #include <string>
5 #include <map>
6 #include <queue>
7 #include <vector>
8
9 #include "node.h"
10
11 void makeFrequencyDictionary(const std::string &text, std::map<char, int> &frequency);
12
13 void loadFrequencyDictionary(const std::string &dict_filepath, std::map<char, int> &frequency);
14
15 void saveFrequencyDictionary(const std::string &dict_filepath, const std::map<char, int> &frequency);
16
17 void makeHeap(std::priority_queue<Node *, std::vector<Node *>, NodeComparator> &minBinaryHeap, const
    std::map<char, int> &frequency);
18
19 void calculateCodes(Node *root, std::map<char, std::string> &dictionary, std::string code = "");
20
```

```
69 void encode(const std::string &input_filepath, const std::string &output_filepath, std::map<char,  
    std::string> &dictionary);  
70  
82 void decode(const std::string &input_filepath, const std::string &output_filepath, std::map<char,  
    std::string> &dictionary, std::map<char, int> &frequency);  
83  
93 void compress(const std::string &input_filepath, const std::string &output_filepath, const std::string  
    &dict_filepath);  
94  
104 void decompress(const std::string &input_filepath, const std::string &output_filepath, const std::string  
    &dict_filepath);  
105  
106 #endif
```

4.4 main.cpp File Reference

```
#include <iostream>  
#include <unistd.h>  
#include "utils.h"  
#include "functions.h"
```

Functions

- void `printHelp` ()
- int `main` (int argc, char *argv[])

4.4.1 Function Documentation

4.4.1.1 main()

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

4.4.1.2 printHelp()

```
void printHelp ( )
```

4.5 node.cpp File Reference

```
#include "node.h"
```

4.6 node.h File Reference

```
#include <iostream>
```

Classes

- struct [Node](#)
- struct [NodeComparator](#)

4.7 node.h

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

```
1 #ifndef NODE_H
2 #define NODE_H
3
4 #include <iostream>
5
11 struct Node
12 {
13 public:
14     int frequency;
15     char character;
16     std::string code;
17
18     Node *left = NULL;
19     Node *right = NULL;
20
26     Node(int frequency);
27
34     Node(int frequency, char character);
35 };
36
42 struct NodeComparator
43 {
44 public:
45     bool operator() (const Node *leftNode, const Node *rightNode);
46 };
47
48 #endif
```

4.8 utils.cpp File Reference

```
#include "utils.h"
```

Functions

- std::string [getFileContent](#) (const std::string filepath)

4.8.1 Function Documentation

4.8.1.1 getFileContent()

```
std::string getFileContent (
    const std::string filepath )
```

Funkcja wczytuje zawartosc pliku okreslonego w sciezce.

Parameters

<i>filepath</i>	ściezka do pliku
-----------------	------------------

Returns

funkcja zwraca std::string zawartosci

4.9 utils.h File Reference

```
#include <iostream>
#include <fstream>
```

Functions

- std::string [getFileContent](#) (const std::string filepath)

4.9.1 Function Documentation

4.9.1.1 getFileContent()

```
std::string getFileContent (
    const std::string filepath )
```

Funkcja wczytuje zawartosc pliku okreslonego w sciezce.

Parameters

<i>filepath</i>	ściezka do pliku
-----------------	------------------

Returns

funkcja zwraca std::string zawartosci

4.10 utils.h

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

```
1 #ifndef UTILS_H
2 #define UTILS_H
3
4 #include <iostream>
5 #include <fstream>
6
13 std::string getFileContent (const std::string filepath);
14
15 #endif
```


Index

- calculateCodes
 - functions.cpp, [9](#)
 - functions.h, [13](#)
- character
 - Node, [6](#)
- code
 - Node, [6](#)
- compress
 - functions.cpp, [10](#)
 - functions.h, [13](#)
- decode
 - functions.cpp, [10](#)
 - functions.h, [14](#)
- decompress
 - functions.cpp, [10](#)
 - functions.h, [14](#)
- encode
 - functions.cpp, [11](#)
 - functions.h, [15](#)
- frequency
 - Node, [6](#)
- functions.cpp, [9](#)
 - calculateCodes, [9](#)
 - compress, [10](#)
 - decode, [10](#)
 - decompress, [10](#)
 - encode, [11](#)
 - loadFrequencyDictionary, [11](#)
 - makeFrequencyDictionary, [11](#)
 - makeHeap, [12](#)
 - saveFrequencyDictionary, [12](#)
- functions.h, [12](#)
 - calculateCodes, [13](#)
 - compress, [13](#)
 - decode, [14](#)
 - decompress, [14](#)
 - encode, [15](#)
 - loadFrequencyDictionary, [15](#)
 - makeFrequencyDictionary, [15](#)
 - makeHeap, [16](#)
 - saveFrequencyDictionary, [16](#)
- getFileContent
 - utils.cpp, [18](#)
 - utils.h, [19](#)
- left
 - Node, [6](#)
- loadFrequencyDictionary
 - functions.cpp, [11](#)
 - functions.h, [15](#)
- main
 - main.cpp, [17](#)
- main.cpp, [17](#)
 - main, [17](#)
 - printHelp, [17](#)
- makeFrequencyDictionary
 - functions.cpp, [11](#)
 - functions.h, [15](#)
- makeHeap
 - functions.cpp, [12](#)
 - functions.h, [16](#)
- Node, [5](#)
 - character, [6](#)
 - code, [6](#)
 - frequency, [6](#)
 - left, [6](#)
 - Node, [5](#), [6](#)
 - right, [6](#)
- node.cpp, [17](#)
- node.h, [18](#)
- NodeComparator, [7](#)
 - operator(), [7](#)
- operator()
 - NodeComparator, [7](#)
- printHelp
 - main.cpp, [17](#)
- right
 - Node, [6](#)
- saveFrequencyDictionary
 - functions.cpp, [12](#)
 - functions.h, [16](#)
- utils.cpp, [18](#)
 - getFileContent, [18](#)
- utils.h, [19](#)
 - getFileContent, [19](#)