

DCT

Generated by Doxygen 1.9.5



---

<b>1 Class Index</b>	<b>1</b>
1.1 Class List . . . . .	1
<b>2 File Index</b>	<b>3</b>
2.1 File List . . . . .	3
<b>3 Class Documentation</b>	<b>5</b>
3.1 gate Struct Reference . . . . .	5
<b>4 File Documentation</b>	<b>7</b>
4.1 DCT/gates.cpp File Reference . . . . .	7
4.1.1 Function Documentation . . . . .	7
4.1.1.1 evaluate_gate() . . . . .	7
4.1.1.2 load_data() . . . . .	8
4.1.1.3 startevaluation() . . . . .	8
4.2 DCT/gates.h File Reference . . . . .	9
4.2.1 Function Documentation . . . . .	9
4.2.1.1 evaluate_gate() . . . . .	9
4.2.1.2 load_data() . . . . .	10
4.2.1.3 startevaluation() . . . . .	10
4.3 gates.h . . . . .	11
4.4 DCT/main.cpp File Reference . . . . .	11
<b>Index</b>	<b>13</b>



# Chapter 1

## Class Index

### 1.1 Class List

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

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



## Chapter 2

# File Index

### 2.1 File List

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

DCT/ <a href="#">gates.cpp</a> . . . . .	7
DCT/ <a href="#">gates.h</a> . . . . .	9
DCT/ <a href="#">main.cpp</a> . . . . .	11





## Chapter 3

# Class Documentation

### 3.1 gate Struct Reference

#### Public Attributes

- int **inx**
- int **iny**
- gatename **name**

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

- DCT/[gates.h](#)



# Chapter 4

## File Documentation

### 4.1 DCT/gates.cpp File Reference

```
#include <iostream>
#include <fstream>
#include <unordered_map>
#include <sstream>
#include <string>
#include "gates.h"
```

#### Functions

- void [load\\_data](#) (string &gatesname, unordered\_map< int, [gate](#) > &gates, vector< int > &outputnodes, int &inputamount)
- void [startevaluation](#) (string &line, unordered\_map< int, [gate](#) > &gates, ofstream &output, vector< int > &outputnodes, int &inputamount)
- bool [evaluate\\_gate](#) (const int &gatenum, unordered\_map< int, [gate](#) > &gates, ofstream &output)

#### 4.1.1 Function Documentation

##### 4.1.1.1 [evaluate\\_gate\(\)](#)

```
bool evaluate_gate (
    const int & gatenum,
    unordered_map< int, gate > & gates,
    ofstream & output )
```

The function evaluates logic gates in order to find the value of a given node.

#### Parameters

<i>gatenum</i>	Number of the node that is being evaluated.
<i>gates</i>	Unordered map which holds gates and inputs.
<i>output</i>	Output file.

**Returns**

The function returns the value of the node.

**Author**

Jakub Knapik

**4.1.1.2 load\_data()**

```
void load_data (
    string & gatesname,
    unordered_map< int, gate > & gates,
    vector< int > & outputnodes,
    int & inputamount )
```

The function loads data from files.

**Parameters**

<i>gatesname</i>	Name of the input file with input gates.
<i>gates</i>	Unordered map which holds gates and inputs.
<i>outputnodes</i>	Vector which holds output nodes.
<i>inputamount</i>	Counter holding the number of input nodes.

**Author**

Jakub Knapik

**4.1.1.3 startevaluation()**

```
void startevaluation (
    string & line,
    unordered_map< int, gate > & gates,
    ofstream & output,
    vector< int > & outputnodes,
    int & inputamount )
```

The function starts the evaluation of the output nodes, and outputs the result.

**Parameters**

<i>line</i>	Values for input nodes.
<i>gates</i>	Unordered map which holds gates and inputs.
<i>output</i>	Output file.
<i>outputnodes</i>	Vector which holds output nodes.
<i>inputamount</i>	Counter holding the number of input nodes.

Author

Jakub Knapik

## 4.2 DCT/gates.h File Reference

```
#include <iostream>
#include <unordered_map>
```

### Classes

- struct [gate](#)

### Enumerations

- enum **gatename** {  
    **IN** , **AND** , **NAND** , **OR** ,  
    **NOR** , **XOR** , **XNOR** , **NEG** }

### Functions

- void [load\\_data](#) (string &gatesname, unordered\_map< int, [gate](#) > &gates, vector< int > &outputnodes, int &inputamount)
- void [startevaluation](#) (string &line, unordered\_map< int, [gate](#) > &gates, ofstream &output, vector< int > &outputnodes, int &inputamount)
- bool [evaluate\\_gate](#) (const int &gatenum, unordered\_map< int, [gate](#) > &gates, ofstream &output)

#### 4.2.1 Function Documentation

##### 4.2.1.1 [evaluate\\_gate\(\)](#)

```
bool evaluate_gate (
    const int & gatenum,
    unordered_map< int, gate > & gates,
    ofstream & output )
```

The function evaluates logic gates in order to find the value of a given node.

#### Parameters

<i>gatenum</i>	Number of the node that is being evaluated.
<i>gates</i>	Unordered map which holds gates and inputs.
<i>output</i>	Output file.

**Returns**

The function returns the value of the node.

**Author**

Jakub Knapik

**4.2.1.2 load\_data()**

```
void load_data (
    string & gatesname,
    unordered_map< int, gate > & gates,
    vector< int > & outputnodes,
    int & inputamount )
```

The function loads data from files.

**Parameters**

<i>gatesname</i>	Name of the input file with input gates.
<i>gates</i>	Unordered map which holds gates and inputs.
<i>outputnodes</i>	Vector which holds output nodes.
<i>inputamount</i>	Counter holding the number of input nodes.

**Author**

Jakub Knapik

**4.2.1.3 startevaluation()**

```
void startevaluation (
    string & line,
    unordered_map< int, gate > & gates,
    ofstream & output,
    vector< int > & outputnodes,
    int & inputamount )
```

The function starts the evaluation of the output nodes, and outputs the result.

**Parameters**

<i>line</i>	Values for input nodes.
<i>gates</i>	Unordered map which holds gates and inputs.
<i>output</i>	Output file.
<i>outputnodes</i>	Vector which holds output nodes.
<i>inputamount</i>	Counter holding the number of input nodes.

## Author

Jakub Knapik

## 4.3 gates.h

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

```
1
2 #include <iostream>
3 #include <unordered_map>
4 using namespace std;
5
6 #ifndef gates_h
7 #define gates_h
8 enum gatename{IN, AND, NAND, OR, NOR, XOR, XNOR, NEG};
9 struct gate
10 {
11     int inx;
12     int iny;
13     gatename name;
14 };
22 void load_data(string& gatesname, unordered_map<int, gate>& gates, vector<int>& outputnodes, int&
    inputamount);
31 void startevaluation(string& line, unordered_map<int, gate>& gates, ofstream& output, vector<int>&
    outputnodes, int& inputamount);
39 bool evaluate_gate(const int& gatenum, unordered_map<int, gate>& gates, ofstream& output);
40
41 #endif // GATES_H
```

## 4.4 DCT/main.cpp File Reference

```
#include <iostream>
#include <fstream>
#include <unordered_map>
#include <vector>
#include <string>
#include "gates.h"
```

### Functions

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





# Index

DCT/gates.cpp, [7](#)  
DCT/gates.h, [9](#), [11](#)  
DCT/main.cpp, [11](#)

evaluate\_gate  
    gates.cpp, [7](#)  
    gates.h, [9](#)

gate, [5](#)

gates.cpp  
    evaluate\_gate, [7](#)  
    load\_data, [8](#)  
    startevaluation, [8](#)

gates.h  
    evaluate\_gate, [9](#)  
    load\_data, [10](#)  
    startevaluation, [10](#)

load\_data  
    gates.cpp, [8](#)  
    gates.h, [10](#)

startevaluation  
    gates.cpp, [8](#)  
    gates.h, [10](#)