Czerwono-Czarni

1.00

Wygenerowano przez Doxygen 1.8.16

1 Indeks klas 1

1.1 Lista klas . . . 1

2 Indeks plików 3

2.1 Lista plików . . . 3

3 Dokumentacja klas 5

3.1 Dokumentacja struktury TreeNode . . . 5

3.1.1 Opis szczegółowy . . . 5

3.1.2 Dokumentacja atrybutów składowych . . . 5

3.1.2.1 _Color . . . 6

3.1.2.2 left . . . 6

3.1.2.3 parent . . . 6

3.1.2.4 right . . . 6

3.1.2.5 value . . . 6

4 Dokumentacja plików 7

4.1 Dokumentacja pliku funkcje.cpp . . . 7

4.1.1 Dokumentacja funkcji . . . 8

4.1.1.1 Add() . . . 8

4.1.1.2 Add_Recovery() . . . 8

4.1.1.3 ChangeToColor() . . . 10

4.1.1.4 Deallocate() . . . 10

4.1.1.5 Delete() . . . 10

4.1.1.6 Delete_Recovery() . . . 11

4.1.1.7 FileGraph() . . . 11

4.1.1.8 FilePrint() . . . 11

4.1.1.9 Find() . . . 12

4.1.1.10 GetParent() . . . 12

4.1.1.11 GetSibling() . . . 12

4.1.1.12 Graph() . . . 13

4.1.1.13 GraphToFile() . . . 13

4.1.1.14 lookForSign() . . . 13

4.1.1.15 Print() . . . 14

4.1.1.16 PrintToFile() . . . 14

4.1.1.17 ReadFromFile() . . . 14

4.1.1.18 rotate_left() . . . 14

4.1.1.19 rotate_right() . . . 15

4.1.1.20 StartCase() . . . 15

4.1.1.21 Successor() . . . 15

4.2 Dokumentacja pliku funkcje.h . . . 16

4.2.1 Dokumentacja funkcji . . . 17

4.2.1.1 Add() . . . 17

4.2.1.2 Add_Recovery() . . . 17
4.2.1.3 ChangeToColor() . . . 18
4.2.1.4 Deallocate() . . . 18
4.2.1.5 Delete() . . . 18
4.2.1.6 Delete_Recovery() . . . 19
4.2.1.7 FileGraph() . . . 19
4.2.1.8 FilePrint() . . . 19
4.2.1.9 Find() . . . 20
4.2.1.10 GetParent() . . . 20
4.2.1.11 GetSibling() . . . 20
4.2.1.12 Graph() . . . 21
4.2.1.13 GraphToFile() . . . 21
4.2.1.14 lookForSign() . . . 21
4.2.1.15 Print() . . . 22
4.2.1.16 PrintToFile() . . . 22
4.2.1.17 ReadFromFile() . . . 22
4.2.1.18 rotate_left() . . . 23
4.2.1.19 rotate_right() . . . 23
4.2.1.20 StartCase() . . . 23
4.2.1.21 Successor() . . . 24
4.3 Dokumentacja pliku main.cpp . . . 24
4.3.1 Dokumentacja funkcji . . . 24
4.3.1.1 main() . . . 24
4.4 Dokumentacja pliku struktury.h . . . 25
4.4.1 Dokumentacja definicji typów . . . 25
4.4.1.1 T . . . 25
4.4.2 Dokumentacja typów wyliczanych . . . 25
4.4.2.1 Color . . . 25

Indeks **27**

Rozdział 1

Indeks klas

1.1 Lista klas

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

TreeNode
Struktura węzła drzewa czerwono-czarnego.
5

Rozdział 2

Indeks plików

2.1 Lista plików

Tutaj znajduje się lista wszystkich plików z ich krótkimi opisami:

funkcje.cpp 7
funkcje.h 16
main.cpp 24
struktury.h 25

Rozdział 3

Dokumentacja klas

3.1 Dokumentacja struktury TreeNode

Struktura węzła drzewa czerwono-czarnego.

`#include <struktury.h>`

Atrybuty publiczne

- Color _Color {}
  Kolor typu wyliczeniowego, określający kolor węzła.
- T value
  Wartość typu domyślnego T, przechowywana w węźle.
- TreeNode * left {}
  Wskaźnik na węzeł lewego poddrzewa.
- TreeNode * right {}
  Wskaźnik na węzeł prawego poddrzewa.
- TreeNode * parent {}
  Wskaźnik na węzeł-rodzic poddrzewa.

3.1.1 Opis szczegółowy

Struktura węzła drzewa czerwono-czarnego.

3.1.2 Dokumentacja atrybutów składowych

3.1.2.1 _Color

```
Color TreeNode::_Color {}
```

Kolor typu wyliczeniowego, określający kolor węzła.

3.1.2.2 left

```
TreeNode* TreeNode::left {}
```

Wskaźnik na węzeł lewego poddrzewa.

3.1.2.3 parent

```
TreeNode* TreeNode::parent {}
```

Wskaźnik na węzeł-rodzic poddrzewa.

3.1.2.4 right

```
TreeNode* TreeNode::right {}
```

Wskaźnik na węzeł prawego poddrzewa.

3.1.2.5 value

```
T TreeNode::value
```

Wartość typu domyślnego T, przechowywana w węźle.

Dokumentacja dla tej struktury została wygenerowana z pliku:

- struktury.h

Rozdział 4

Dokumentacja plików

4.1 Dokumentacja pliku funkcje.cpp

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

Funkcje

- TreeNode ∗ GetParent (TreeNode ∗node)

  Funkcja znajdująca i zwracająca rodzica podanego węzła.
- TreeNode ∗ GetSibling (TreeNode ∗node)

  Funkcja znajdująca i zwracająca rodzeństwo podanego węzła.
- void ChangeToColor (TreeNode ∗node, Color color)

  Funkcja zmieniająca kolor węzła na podany.
- void rotate_left (TreeNode ∗&node)

  Lokalna zmiana struktury poddrzewa (w lewo).
- void rotate_right (TreeNode ∗&node)

  Lokalna zmiana struktury poddrzewa (w lewo).
- void Add_Recovery (TreeNode ∗z)

  Funkcja przywracająca własności drzewa Czerwono-Czarnego po wykonaniu operacji dodawania nowego węzła do drzewa.
- void Add (TreeNode ∗&root, const T &value)

  Funkcja dodająca do drzewa węzeł o podanej wartości.
- TreeNode ∗ Find (TreeNode ∗root, const T &value)

  Funkcja szukająca w drzewie węzła o podanej wartości. Jeżeli węzeł zostanie odnaleziony funkcja zwraca wskaźnik na niego - w przeciwnym wypadku zwracany jest nullptr.
- TreeNode ∗ Successor (TreeNode ∗node)

  Funkcja odnajdująca następcę danego węzła. Jeżeli węzeł zostanie odnaleziony, funkcja zwraca wskaźnik na niego, w przeciwnym wypadku zwracany jest nullptr.
- void StartCase (TreeNode ∗node, bool &isCase2)

  Funkcja obejmująca przestawienie odpowiednich wskaźników i dealokowanie odpowiedniej pamięci - pamięci węzła y.
- void Delete_Recovery (TreeNode ∗&node)

Funkcja przywracająca własności drzewa czerwono-czarnego z widoku usuniętego wcześniej węzła-następcy węzła przeznaczonego do usunięcia.

- void Delete (TreeNode ∗&root, const T &value)

  Funkcja odnajduje węzeł o podanej wartości, usuwa go i dealokuje zaalokowaną wcześniej pamięć.
- void Print (TreeNode ∗root)

  Funkcja przyjmująca jako parametr korzeń drzewa. Wypisuje wartości wszystkich węzłów drzewa w kolejności in-order.
- void Graph (TreeNode ∗root, int indent=0)

  Funkcja wypisuje wartości wszystkich węzłów drzewa w kolejności rosnącej z uwzględnieniem wysokości drzewa i kolorów poszczególnych węzłów.
- void Deallocate (TreeNode ∗&root)

  Funkcja całkowicie usuwa drzewo i dealokuje zajętą przezeń pamięć.
- void PrintToFile (TreeNode ∗root, ofstream &file)
- void GraphToFile (TreeNode ∗root, ofstream &file, int indent=0)
- size_t lookForSign (const std::string &line, char searched)

  Funkcja odnajdująca pozycję podanego znaku w łańcuchu znakowym. Używana jest w funkcji ReadFromFile, by usunąć z pliku wszystkie komentarze, puste linie ora linie, które zawierają niepoprawnie wprowadzone komendy.
- void FilePrint (TreeNode ∗&root, const std::string &file_name, const bool &toOverwrite)

  Funkcja wypisująca do pliku o podanej nazwie wartości wszystich węzłów drzewa w porządku rosnącym.
- void FileGraph (TreeNode ∗&root, const std::string &file_name, const bool &toOverwrite)

  Funkcja wypisująca do pliku o podanej nazwie wizualizację wszystkich węzłów znajdujących się w drzewie, z uwzględnieniem ich koloru, np: [33] - węzeł czarny o wartości 33, (13) - węzeł czerwony o wartości 13.
- void ReadFromFile (const std::string &file_name)

  Funkcja odczytująca plik "linia po linii" i wykonująca odpowiednie polecenia.

4.1.1 Dokumentacja funkcji

4.1.1.1 Add()

```
void Add (
            TreeNode *& root,
            const T & value )
```

Funkcja dodająca do drzewa węzeł o podanej wartości.

Parametry

root	wskaźnik na korzeń drzewa.
T	wartość.

4.1.1.2 Add_Recovery()

```
void Add_Recovery (
            TreeNode * z )
```

Funkcja przywracająca własności drzewa Czerwono-Czarnego po wykonaniu operacji dodawania nowego węzła do drzewa.

Parametry

z	wskaźnik na węzeł drzewa.

4.1.1.3 ChangeToColor()

```
void ChangeToColor (
            TreeNode * node,
            Color color )
```

Funkcja zmieniająca kolor węzła na podany.

Parametry

node	wskaźnik na węzeł drzewa, którego kolor będzie zmieniony.
color	kolor, na jaki węzeł powinien być zmieniony (o ile węzeł istnieje).

4.1.1.4 Deallocate()

```
void Deallocate (
            TreeNode *& root )
```

Funkcja całkowicie usuwa drzewo i dealokuje zajętą przezeń pamięć.

Parametry

root	wskaźnik na korzeń drzewa.

4.1.1.5 Delete()

```
void Delete (
            TreeNode *& root,
            const T & value )
```

Funkcja odnajduje węzeł o podanej wartości, usuwa go i dealokuje zaalokowaną wcześniej pamięć.

Parametry

root	wskaźnik na korzeń drzewa.
value	wartość węzła do usunięcia.

4.1.1.6 Delete_Recovery()

```
void Delete_Recovery (
            TreeNode *& node )
```

Funkcja przywracająca własności drzewa czerwono-czarnego z widoku usuniętego wcześniej węzła-następcy węzła przeznaczonego do usunięcia.

Parametry

node	węzeł drzewa czerwono-czarnego.

4.1.1.7 FileGraph()

```
void FileGraph (
            TreeNode *& root,
            const std::string & file_name,
            const bool & toOverwrite )
```

Funkcja wypisująca do pliku o podanej nazwie wizualizację wszystkich węzłów znajdujących się w drzewie, z uwzględnieniem ich koloru, np: [33] - węzeł czarny o wartości 33, (13) - węzeł czerwony o wartości 13.

Parametry

root	wskaźnik na korzeń drzewa.
file_name	pełna nazwa pliku, do którego mają być wypisywane wszystkie wartości węzłów drzewa.
toOverwrite	parametr określający czy wartości w pliku mają być nadpisane.

4.1.1.8 FilePrint()

```
void FilePrint (
            TreeNode *& root,
            const std::string & file_name,
            const bool & toOverwrite )
```

Funkcja wypisująca do pliku o podanej nazwie wartości wszystich węzłów drzewa w porządku rosnącym.

Parametry

root	wskaźnik na korzeń drzewa.
file_name	pełna nazwa pliku, do którego mają być wypisywane wszystkie wartości węzłów drzewa.
toOverwrite	parametr określający czy wartości w pliku mają być nadpisane.

4.1.1.9 Find()

```
TreeNode* Find (
            TreeNode * root,
            const T & value )
```

Funkcja szukająca w drzewie węzła o podanej wartości. Jeżeli węzeł zostanie odnaleziony funkcja zwraca wskaźnik na niego - w przeciwnym wypadku zwracany jest nullptr.

Parametry

root	wskaźnik na korzeń drzewa.
value	wartosc.

Zwraca

węzeł o podanej wartości lub nullptr.

4.1.1.10 GetParent()

```
TreeNode* GetParent (
            TreeNode * node )
```

Funkcja znajdująca i zwracająca rodzica podanego węzła.

Parametry

node	wskaźnik na odpowiedni węzeł drzewa.

Zwraca

rodzic podanego węzła lub nullptr.

4.1.1.11 GetSibling()

```
TreeNode* GetSibling (
            TreeNode * node )
```

Funkcja znajdująca i zwracająca rodzeństwo podanego węzła.

Parametry

node	wskaźnik na odpowiedni węzeł drzewa.

Zwraca

rodzeństwo węzła lub nullptr.

4.1.1.12 Graph()

```
void Graph (
            TreeNode * root,
            int indent = 0 )
```

Funkcja wypisuje wartości wszystkich węzłów drzewa w kolejności rosnącej z uwzględnieniem wysokości drzewa i kolorów poszczególnych węzłów.

Parametry

root	wskaźnik na korzeń drzewa.
indent	wcięcie - wyrażane w ilości znaków białych.

4.1.1.13 GraphToFile()

```
void GraphToFile (
            TreeNode * root,
            ofstream & file,
            int indent = 0 )
```

4.1.1.14 lookForSign()

```
size_t lookForSign (
            const std::string & line,
            char searched )
```

Funkcja odnajdująca pozycję podanego znaku w łańcuchu znakowym. Używana jest w funkcji ReadFromFile, by usunąć z pliku wszystkie komentarze, puste linie ora linie, które zawierają niepoprawnie wprowadzone komendy.

Parametry

line	łańcuch znakowy.
searched	znak, którego pozycja ma zostać odnaleziona.

Zwraca

pozycję odnalezionego znaku (jeżeli nie wystąpił - zwracana jest długość podanego łańcucha znakowego).

4.1.1.15 Print()

```
void Print (
            TreeNode * root )
```

Funkcja przyjmująca jako parametr korzeń drzewa. Wypisuje wartości wszystkich węzłów drzewa w kolejności in-order.

Parametry

root	wskaźnik na korzeń drzewa.

4.1.1.16 PrintToFile()

```
void PrintToFile (
            TreeNode * root,
            ofstream & file )
```

4.1.1.17 ReadFromFile()

```
void ReadFromFile (
            const std::string & file_name )
```

Funkcja odczytująca plik "linia po linii" i wykonująca odpowiednie polecenia.

Parametry

file_name	pełna nazwa pliku.

4.1.1.18 rotate_left()

```
void rotate_left (
            TreeNode *& node )
```

Lokalna zmiana struktury poddrzewa (w lewo).

Parametry

node	wskaźnik na węzeł drzewa.

Ostrzeżenie

prawe poddrzewo węzła musi istnieć!

4.1.1.19 rotate_right()

```
void rotate_right (
            TreeNode *& node )
```

Lokalna zmiana struktury poddrzewa (w lewo).

Parametry

node	wskaźnik na węzeł drzewa.

Ostrzeżenie

lewe poddrzewo węzła musi istnieć!

4.1.1.20 StartCase()

```
void StartCase (
            TreeNode * node,
            bool & isCase2 )
```

Funkcja obejmująca przestawienie odpowiednich wskaźników i dealokowanie odpowiedniej pamięci - pamięci węzła y.

Parametry

node	wskaźnik na węzeł drzewa.
isCase2	sprawdzenie, czy istnieje potrzeba dalszego przywracania własności drzewa czerwono-czarnego.

4.1.1.21 Successor()

```
TreeNode* Successor (
            TreeNode * node )
```

Funkcja odnajdująca następcę danego węzła. Jeżeli węzeł zostanie odnaleziony, funkcja zwraca wskaźnik na niego, w przeciwnym wypadku zwracany jest nullptr.

Parametry

node	wskaźnik na węzeł drzewa.

Zwraca

następca węzła.

4.2 Dokumentacja pliku funkcje.h

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

Funkcje

- void Add (TreeNode ∗&root, const T &value)

  Funkcja dodająca do drzewa węzeł o podanej wartości.
- void Add_Recovery (TreeNode ∗z)

  Funkcja przywracająca własności drzewa Czerwono-Czarnego po wykonaniu operacji dodawania nowego węzła do drzewa.
- void rotate_left (TreeNode ∗&node)

  Lokalna zmiana struktury poddrzewa (w lewo).
- void rotate_right (TreeNode ∗&node)

  Lokalna zmiana struktury poddrzewa (w lewo).
- TreeNode ∗ Find (TreeNode ∗root, const T &value)

  Funkcja szukająca w drzewie węzła o podanej wartości. Jeżeli węzeł zostanie odnaleziony funkcja zwraca wskaźnik na niego - w przeciwnym wypadku zwracany jest nullptr.
- void StartCase (TreeNode ∗node, bool &isCase2)

  Funkcja obejmująca przestawienie odpowiednich wskaźników i dealokowanie odpowiedniej pamięci - pamięci węzła y.
- TreeNode ∗ Successor (TreeNode ∗node)

  Funkcja odnajdująca następcę danego węzła. Jeżeli węzeł zostanie odnaleziony, funkcja zwraca wskaźnik na niego, w przeciwnym wypadku zwracany jest nullptr.
- void Delete (TreeNode ∗&root, const T &value)

  Funkcja odnajduje węzeł o podanej wartości, usuwa go i dealokuje zaalokowaną wcześniej pamięć.
- TreeNode ∗ GetParent (TreeNode ∗node)

  Funkcja znajdująca i zwracająca rodzica podanego węzła.
- TreeNode ∗ GetSibling (TreeNode ∗node)

  Funkcja znajdująca i zwracająca rodzeństwo podanego węzła.
- void ChangeToColor (TreeNode ∗node, Color color)

  Funkcja zmieniająca kolor węzła na podany.
- void Deallocate (TreeNode ∗&root)

  Funkcja całkowicie usuwa drzewo i dealokuje zajętą przezeń pamięć.
- void Delete_Recovery (TreeNode ∗&node)

  Funkcja przywracająca własności drzewa czerwono-czarnego z widoku usuniętego wcześniej węzła-następcy węzła przeznaczonego do usunięcia.

- void Print (TreeNode ∗root)

  Funkcja przyjmująca jako parametr korzeń drzewa. Wypisuje wartości wszystkich węzłów drzewa w kolejności in-order.
- void Graph (TreeNode ∗root, int indent=0)

  Funkcja wypisuje wartości wszystkich węzłów drzewa w kolejności rosnącej z uwzględnieniem wysokości drzewa i kolorów poszczególnych węzłów.
- void ReadFromFile (const std::string &file_name)

  Funkcja odczytująca plik "linia po linii" i wykonująca odpowiednie polecenia.
- void FilePrint (TreeNode ∗&root, const std::string &file_name, const bool &toOverwrite)

  Funkcja wypisująca do pliku o podanej nazwie wartości wszystich węzłów drzewa w porządku rosnącym.
- void FileGraph (TreeNode ∗&root, const std::string &file_name, const bool &toOverwrite)

  Funkcja wypisująca do pliku o podanej nazwie wizualizację wszystkich węzłów znajdujących się w drzewie, z uwzględnieniem ich koloru, np: [33] - węzeł czarny o wartości 33, (13) - węzeł czerwony o wartości 13.
- void PrintToFile (TreeNode ∗root, std::ofstream &file)

  Główny algorytm służący wypisywaniu do pliku wszystkich wartości węzłów drzewa w kolejności rosnącej.
- void GraphToFile (TreeNode ∗root, std::ofstream &file, int indent=0)

  Główny algorytm służący wypisywaniu do pliku wizualizacji pozycji wszystkich węzłów w drzewie czerwono-czarnym.
- size_t lookForSign (const std::string &line, char searched)

  Funkcja odnajdująca pozycję podanego znaku w łańcuchu znakowym. Używana jest w funkcji ReadFromFile, by usunąć z pliku wszystkie komentarze, puste linie ora linie, które zawierają niepoprawnie wprowadzone komendy.

4.2.1 Dokumentacja funkcji

4.2.1.1 Add()

```
void Add (
            TreeNode *& root,
            const T & value )
```

Funkcja dodająca do drzewa węzeł o podanej wartości.

Parametry

root	wskaźnik na korzeń drzewa.
T	wartość.

4.2.1.2 Add_Recovery()

```
void Add_Recovery (
            TreeNode * z )
```

Funkcja przywracająca własności drzewa Czerwono-Czarnego po wykonaniu operacji dodawania nowego węzła do drzewa.

Parametry

z	wskaźnik na węzeł drzewa.

4.2.1.3 ChangeToColor()

```
void ChangeToColor (
            TreeNode * node,
            Color color )
```

Funkcja zmieniająca kolor węzła na podany.

Parametry

node	wskaźnik na węzeł drzewa, którego kolor będzie zmieniony.
color	kolor, na jaki węzeł powinien być zmieniony (o ile węzeł istnieje).

4.2.1.4 Deallocate()

```
void Deallocate (
            TreeNode *& root )
```

Funkcja całkowicie usuwa drzewo i dealokuje zajętą przezeń pamięć.

Parametry

root	wskaźnik na korzeń drzewa.

4.2.1.5 Delete()

```
void Delete (
            TreeNode *& root,
            const T & value )
```

Funkcja odnajduje węzeł o podanej wartości, usuwa go i dealokuje zaalokowaną wcześniej pamięć.

Parametry

root	wskaźnik na korzeń drzewa.
value	wartość węzła do usunięcia.

4.2.1.6 Delete_Recovery()

```
void Delete_Recovery (
            TreeNode *& node )
```

Funkcja przywracająca własności drzewa czerwono-czarnego z widoku usuniętego wcześniej węzła-następcy węzła przeznaczonego do usunięcia.

Parametry

node	węzeł drzewa czerwono-czarnego.

4.2.1.7 FileGraph()

```
void FileGraph (
            TreeNode *& root,
            const std::string & file_name,
            const bool & toOverwrite )
```

Funkcja wypisująca do pliku o podanej nazwie wizualizację wszystkich węzłów znajdujących się w drzewie, z uwzględnieniem ich koloru, np: [33] - węzeł czarny o wartości 33, (13) - węzeł czerwony o wartości 13.

Parametry

root	wskaźnik na korzeń drzewa.
file_name	pełna nazwa pliku, do którego mają być wypisywane wszystkie wartości węzłów drzewa.
toOverwrite	parametr określający czy wartości w pliku mają być nadpisane.

4.2.1.8 FilePrint()

```
void FilePrint (
            TreeNode *& root,
            const std::string & file_name,
            const bool & toOverwrite )
```

Funkcja wypisująca do pliku o podanej nazwie wartości wszystich węzłów drzewa w porządku rosnącym.

Parametry

root	wskaźnik na korzeń drzewa.
file_name	pełna nazwa pliku, do którego mają być wypisywane wszystkie wartości węzłów drzewa.
toOverwrite	parametr określający czy wartości w pliku mają być nadpisane.

4.2.1.9 Find()

```
TreeNode* Find (
            TreeNode * root,
            const T & value )
```

Funkcja szukająca w drzewie węzła o podanej wartości. Jeżeli węzeł zostanie odnaleziony funkcja zwraca wskaźnik na niego - w przeciwnym wypadku zwracany jest nullptr.

Parametry

root	wskaźnik na korzeń drzewa.
value	wartosc.

Zwraca

węzeł o podanej wartości lub nullptr.

4.2.1.10 GetParent()

```
TreeNode* GetParent (
            TreeNode * node )
```

Funkcja znajdująca i zwracająca rodzica podanego węzła.

Parametry

node	wskaźnik na odpowiedni węzeł drzewa.

Zwraca

rodzic podanego węzła lub nullptr.

4.2.1.11 GetSibling()

```
TreeNode* GetSibling (
            TreeNode * node )
```

Funkcja znajdująca i zwracająca rodzeństwo podanego węzła.

Parametry

node	wskaźnik na odpowiedni węzeł drzewa.

Zwraca

rodzeństwo węzła lub nullptr.

4.2.1.12 Graph()

```
void Graph (
            TreeNode * root,
            int indent = 0 )
```

Funkcja wypisuje wartości wszystkich węzłów drzewa w kolejności rosnącej z uwzględnieniem wysokości drzewa i kolorów poszczególnych węzłów.

Parametry

root	wskaźnik na korzeń drzewa.
indent	wcięcie - wyrażane w ilości znaków białych.

4.2.1.13 GraphToFile()

```
void GraphToFile (
            TreeNode * root,
            std::ofstream & file,
            int indent = 0 )
```

Główny algorytm służący wypisywaniu do pliku wizualizacji pozycji wszystkich węzłów w drzewie czerwono-czarnym.

Parametry

root	wskaźnik na korzeń drzewa.
file	strumień plikowy.
indent	wcięcie - wyrażane w ilości znaków białych.

4.2.1.14 lookForSign()

```
size_t lookForSign (
            const std::string & line,
            char searched )
```

Funkcja odnajdująca pozycję podanego znaku w łańcuchu znakowym. Używana jest w funkcji ReadFromFile, by usunąć z pliku wszystkie komentarze, puste linie ora linie, które zawierają niepoprawnie wprowadzone komendy.

Parametry

line	łańcuch znakowy.
searched	znak, którego pozycja ma zostać odnaleziona.

Zwraca

pozycję odnalezionego znaku (jeżeli nie wystąpił - zwracana jest długość podanego łańcucha znakowego).

4.2.1.15 Print()

```
void Print (
            TreeNode * root )
```

Funkcja przyjmująca jako parametr korzeń drzewa. Wypisuje wartości wszystkich węzłów drzewa w kolejności in-order.

Parametry

root	wskaźnik na korzeń drzewa.

4.2.1.16 PrintToFile()

```
void PrintToFile (
            TreeNode * root,
            std::ofstream & file )
```

Główny algorytm służący wypisywaniu do pliku wszystkich wartości węzłów drzewa w kolejności rosnącej.

Parametry

root	wskaźnik na korzeń drzewa.
file	strumień plikowy.

4.2.1.17 ReadFromFile()

```
void ReadFromFile (
            const std::string & file_name )
```

Funkcja odczytująca plik "linia po linii" i wykonująca odpowiednie polecenia.

Parametry

file_name	pełna nazwa pliku.

4.2.1.18 rotate_left()

```
void rotate_left (
            TreeNode *& node )
```

Lokalna zmiana struktury poddrzewa (w lewo).

Parametry

node	wskaźnik na węzeł drzewa.

Ostrzeżenie

prawe poddrzewo węzła musi istnieć!

4.2.1.19 rotate_right()

```
void rotate_right (
            TreeNode *& node )
```

Lokalna zmiana struktury poddrzewa (w lewo).

Parametry

node	wskaźnik na węzeł drzewa.

Ostrzeżenie

lewe poddrzewo węzła musi istnieć!

4.2.1.20 StartCase()

```
void StartCase (
            TreeNode * node,
            bool & isCase2 )
```

Funkcja obejmująca przestawienie odpowiednich wskaźników i dealokowanie odpowiedniej pamięci - pamięci węzła y.

Parametry

node	wskaźnik na węzeł drzewa.
isCase2	sprawdzenie, czy istnieje potrzeba dalszego przywracania własności drzewa czerwono-czarnego.

4.2.1.21 Successor()

```
TreeNode* Successor (
            TreeNode * node )
```

Funkcja odnajdująca następcę danego węzła. Jeżeli węzeł zostanie odnaleziony, funkcja zwraca wskaźnik na niego, w przeciwnym wypadku zwracany jest nullptr.

Parametry

node	wskaźnik na węzeł drzewa.

Zwraca

następca węzła.

4.3 Dokumentacja pliku main.cpp

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

Funkcje

- int main (int argc, char ∗argv[ ])

4.3.1 Dokumentacja funkcji

4.3.1.1 main()

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

4.4 Dokumentacja pliku struktury.h

Komponenty

- struct TreeNode

  Struktura węzła drzewa czerwono-czarnego.

Definicje typów

- typedef double T

  Szablon typu zmiennej (domyślnie: double).

Wyliczenia

- enum Color { Color::red, Color::black }

  Typ wyliczeniowy, opisujący kolor węzła.

4.4.1 Dokumentacja definicji typów

4.4.1.1 T

```
typedef double T
```

Szablon typu zmiennej (domyślnie: double).

4.4.2 Dokumentacja typów wyliczanych

4.4.2.1 Color

```
enum Color  [strong]
```

Typ wyliczeniowy, opisujący kolor węzła.

Wartości wyliczeń

red	
black	

Indeks

_Color
TreeNode, 5

Add
funkcje.cpp, 8
funkcje.h, 17
Add_Recovery
funkcje.cpp, 8
funkcje.h, 17

black
struktury.h, 25

ChangeToColor
funkcje.cpp, 10
funkcje.h, 18
Color
struktury.h, 25

Deallocate
funkcje.cpp, 10
funkcje.h, 18
Delete
funkcje.cpp, 10
funkcje.h, 18
Delete_Recovery
funkcje.cpp, 11
funkcje.h, 19

FileGraph
funkcje.cpp, 11
funkcje.h, 19
FilePrint
funkcje.cpp, 11
funkcje.h, 19
Find
funkcje.cpp, 12
funkcje.h, 20
funkcje.cpp, 7
Add, 8
Add_Recovery, 8
ChangeToColor, 10
Deallocate, 10
Delete, 10
Delete_Recovery, 11
FileGraph, 11
FilePrint, 11
Find, 12
GetParent, 12
GetSibling, 12
Graph, 13
GraphToFile, 13
lookForSign, 13
Print, 14
PrintToFile, 14
ReadFromFile, 14
rotate_left, 14
rotate_right, 15
StartCase, 15
Successor, 15
funkcje.h, 16
Add, 17
Add_Recovery, 17
ChangeToColor, 18
Deallocate, 18
Delete, 18
Delete_Recovery, 19
FileGraph, 19
FilePrint, 19
Find, 20
GetParent, 20
GetSibling, 20
Graph, 21
GraphToFile, 21
lookForSign, 21
Print, 22
PrintToFile, 22
ReadFromFile, 22
rotate_left, 23
rotate_right, 23
StartCase, 23
Successor, 24

GetParent
funkcje.cpp, 12
funkcje.h, 20
GetSibling
funkcje.cpp, 12
funkcje.h, 20
Graph
funkcje.cpp, 13
funkcje.h, 21
GraphToFile
funkcje.cpp, 13
funkcje.h, 21

left
TreeNode, 6
lookForSign
funkcje.cpp, 13
funkcje.h, 21

main
    main.cpp, 24
main.cpp, 24
    main, 24

parent
    TreeNode, 6
Print
    funkcje.cpp, 14
    funkcje.h, 22
PrintToFile
    funkcje.cpp, 14
    funkcje.h, 22

ReadFromFile
    funkcje.cpp, 14
    funkcje.h, 22
red
    struktury.h, 25
right
    TreeNode, 6
rotate_left
    funkcje.cpp, 14
    funkcje.h, 23
rotate_right
    funkcje.cpp, 15
    funkcje.h, 23

StartCase
    funkcje.cpp, 15
    funkcje.h, 23
struktury.h, 25
    black, 25
    Color, 25
    red, 25
    T, 25
Successor
    funkcje.cpp, 15
    funkcje.h, 24

T
    struktury.h, 25
TreeNode, 5
    _Color, 5
    left, 6
    parent, 6
    right, 6
    value, 6

value
    TreeNode, 6