Текстовод Уникальность https://textovod.com/unique

Отчет проверки уникальности текста

Дата проверки: 2023-06-22 22:32:25

Уникальность 67%

Хорошо. Подойдет для большинства текстов.

Текст

#pragma once

#include < iomanip>
#include < iostream>
#include < string>

#include "colors.h"

template < typename T> class ThreadedBST {
public:
struct Node {
T value;
// Parent only used to beautifully display nodes, it is not used in anything else
Node *left, *right, *parent;
bool is_left_threaded, is_right_threaded;

Node(T value, Node *parent, Node *left, Node *right)
: value(value), left(left), right(right), parent(parent), is_left_threaded(true),
is_right_threaded(true) {
}
};

private:
ThreadedBST< T> :: Node *tree = nullptr;

// Colored "left" and "right" words
const std: :string left_string = COLOR_YELLOW + "left" + COLOR_RESET;

```
const std: :string right_string = COLOR_CYAN + "right" + COLOR_RESET;
const std: :string moving_left_string = " - moving " + this-> left_string;
const std: :string moving_right_string = " - moving " + this-> right_string;
const std: :string is_thread = COLOR_RED + " [thread]" + COLOR_RESET;

// Colored "value" output
std: :string valueString(T value) {
return "'" + COLOR_YELLOW + std: :to_string(value) + COLOR_RESET + "'";
}

// Shows colored node. Green - parent, Yellow - left of parent, Cyan - right of parent
void showNode(ThreadedBST< T> :: Node *node) {
const int width = 2;
std: :cout < < " - ";

if (! node-> parent) {
std: :cout < < COLOR_GREEN;
} else if (node-> parent-> left == node) {
std: :cout < < COLOR_YELLOW;
} else {
std: :cout < < COLOR_CYAN;
}

std: :cout < < std: :setw(width) < < node-> value < < COLOR_RESET < < " [" < <
COLOR_YELLOW < < "l: ";

if (node-> is_left_threaded & & node-> left) {
std: :cout < < std: :setw(width) < < node-> left-> value;
} else {
std: :cout < < std: :string(width, ' ');
}

std: :cout < < COLOR_RESET < < " | " < < COLOR_CYAN < < "r: ";

if (node-> is_right_threaded & & node-> right) {
std: :cout< < std: :setw(width) < < node-> right-> value;
} else {
std: :cout < < std: :string(width, ' ');
}

std: :cout < < COLOR_RESET < < "]" < < std: :endl;
```

```
}

public:
void insert(T value) {
if (! this-> tree) {
std: :cout < < COLOR_GREEN < < "Tree is empty, creating the root" < <
COLOR_RESET < < std: :endl;

this-> tree = new ThreadedBST< T> :: Node(value, nullptr, nullptr, nullptr);
return;
}

std: :cout < < "Searching for the suitable space: " < < std: :endl;
ThreadedBST< T> :: Node *parent = nullptr;
ThreadedBST< T> :: Node *current = tree;

while (current) {
parent = current;

if (current-> value > value) {
std: :cout < < moving_left_string;

if (current-> is_left_threaded) {
std: :cout < < is_thread < < std: :endl;
break;
}

std: :cout < < std: :endl;
current = current-> left;
} else if (current-> value < value) {
std: :cout < < moving_right_string;

if (current-> is_right_threaded) {
std: :cout < < is_thread < < std: :endl;
break;
}

std: :cout < < std: :endl;
current = current-> right;
} else {
```

```
std: :cout < < COLOR_RED < < "The same value is found. Returning" < <
COLOR_RESET < < std: :endl;

return;
}
}

if (parent-> value > value) {
std: :cout < < "Inserting new node " < < this-> left_string < < " with the " < <
valueString(value) < < std: :endl;
parent-> is_left_threaded = false;
parent-> left = new ThreadedBST< T> :: Node(value, parent, parent-> left, parent);
} else if (parent-> value < value) {
std: :cout < < "Inserting new node " < < this-> right_string < < " with the " < <
valueString(value) < < std: :endl;
parent-> is_right_threaded = false;
parent-> right = new ThreadedBST< T> :: Node(value, parent, parent, parent-> right);
}
}

ThreadedBST< T> :: Node *search(T value) {
if (! this-> tree) {
std: :cout < < "The tree is empty, nothing to search" < < std: :endl;
return nullptr;
}

ThreadedBST< T> :: Node *current = this-> tree;
while (current) {
if (current-> value > value) {
std: :cout < < moving_left_string;

if (current-> is_left_threaded) {
std: :cout < < is_thread < < std: :endl;
break;
}

std: :cout < < std: :endl;
current = current-> left;
} else if (current-> value < value) {
std: :cout < < moving_right_string;
```

```
if (current-> is_right_threaded) {
std: :cout < < is_thread < < std: :endl;
break;
}

std: :cout < < std: :endl;
current = current-> right;
} else {
std: :cout < < "The value " < < this-> valueString(value) < < COLOR_GREEN < < "
was found" < < COLOR_RESET
< < std: :endl;

return current;
}
}

std: :cout < < "The value " < < this-> valueString(value) < < COLOR_RED < < " was
not found" < < COLOR_RESET
< < std: :endl;

return nullptr;
}

ThreadedBST< T> :: Node *inorderSuccessor(ThreadedBST< T> :: Node *node) {
if (node-> is_right_threaded) {
return node-> right;
}

node = node-> right;
while (node-> is_left_threaded == false) {
node = node-> left;
}

return node;
}

void show() {
if (! this-> tree) {
std: :cout < < "The tree is empty, nothing to show" < < std: :endl;
return;
}
```

```
ThreadedBST< T> :: Node *current = this-> tree;

while (! current-> is_left_threaded) {
current = current-> left;
}

while (current) {
showNode(current);

if (current-> is_right_threaded) {
current = current-> right;
continue;
}

current = current-> right;
while (current-> is_left_threaded == false) {
current = current-> left;
}
}
}
};
```

Источники

- https://programbox.ru/2022/01/06/%D0%BA%D0%B0%D0%BA-%D1%81%D0%BE%D0%B7%D0%B4%D0%B0%D1%82%D1%8C-%D0%BA%D0%BB%D0%B0%D1%81%D1%81-%D0%B4%D0%B2%D0%BE%D0%B8%D1%87%D0%BD%D0%BE%D0%B3%D0%BE-%D0%B4%D0%B5%D1%80%D0%B5%D0%B2%D0%B0/ (13%)
- https://www.CyberForum.ru/cpp-beginners/thread2175991-page2.html (11%)
- https://www.programmersought.com/article/5771313648/ (10%)
- https://bytes.com/topic/c/answers/694010-using-std-cout (9%)
- https://russianblogs.com/article/1327503951/ (9%)
- https://all-learning.com/smart-pointers-in-c/ (9%)
- https://ru.hexlet.io/courses/algorithms-trees/lessons/binary/theory_unit (9%)
- https://infourok.ru/test-po-discipline-osnovi-algoritmizacii-i-programmirovaniya-3057358.html (8%)
- https://www.digitalocean.com/community/tutorials/getline-in-c-plus-plus (7%)
- https://metanit.com/cpp/tutorial/8.1.php (7%)

- https://www.bogotobogo.com/cplusplus/binarytree.php (7%)
- https://topuch.com/programma-peredaet-upravlenie-v-sluchae-esli-znachenie-peremen/index.html (7%)
- https://www.learncpp.com/cpp-tutorial/stdstring_view-part-2/ (6%)
- https://pastebin.com/yeqX73C6 (5%)
- https://www.inapps.net/update-getting-started-with-c-and-influxdb/ (4%)
- https://thenewstack.io/getting-started-with-c-and-influxdb/ (4%)
- https://forum.vingrad.ru/topic-347441.html (3%)
- https://github.com/MIPT-DAFE-CS/765-Algo_s2-classwork/blob/master/sem1/assignment3/task3/task-3_2.cxx (3%)
- https://www.algotree.org/algorithms/tree_graph_traversal/pre_in_post_order/ (3%)
- https://sodocumentation.net/cplusplus/topic/488/std--string (3%)
- https://learntutorials.net/ru/cplusplus/topic/488/%D1%81%D1%82%D0%B0%D0%BD%D%D1%81%D1%82%D1%80%D0%BE%D0%BA%D0%B0 (3%)
- https://learnc.info/adt/binary_tree_traversal.html (2%)