Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации

Сибирский государственный университет телекоммуникаций и информатики

Кафедра ПМиК

Лабораторная Работа №5

По дисциплине «СиАОД»

Построение СДП двоичного дерева.

Работу выполнил

студент 2 курса группы ИП-015

Шеменьков В.В.

Работу проверил

старший преподаватель кафедры ПМиК:

Солодов П.С.

Новосибирск 2021

Цель: Научиться удалять из СДП вершину.
Задачи:

Лабораторная работа №3

1). Разработать процедуру удаления из случайного дерева поиска (СДП) вершины с заданным ключом.
2). Реализовать удаление из СДП 10 вершин, задаваемых с клавиатуры, распечатывать обход дерева слева направо после каждой удаленной вершины.

4). Дополнительное задание (на 5+):
На базе СДП построить словарь частот встречаемости ключевых слов в тексте программы на Си.

Итоги работы:

Скриншоты вывода по заданию:
Задание 2:

```
5 8 14 20 22 28 33 57 64 C:68  69 70 77 80 81 86 88 95 98 99
Input delete Key: 20
5 8 14 22 28 33 57 64 C:68  69 70 77 80 81 86 88 95 98 99
Input delete Key: 22
5 8 14 28 33 57 64 C:68  69 70 77 80 81 86 88 95 98 99
Input delete Key: 28
5 8 14 33 57 64 C:68  69 70 77 80 81 86 88 95 98 99
Input delete Key: 5
8 14 33 57 64 C:68  69 70 77 80 81 86 88 95 98 99
Input delete Key: 99
8 14 33 57 64 C:68  69 70 77 80 81 86 88 95 98
Input delete Key: 95
8 14 33 57 64 C:68  69 70 77 80 81 86 88 98
Input delete Key: 88
8 14 33 57 64 C:68  69 70 77 80 81 86 98
Input delete Key: 98
8 14 33 57 64 C:68  69 70 77 80 81 86
Input delete Key: 8
14 33 57 64 C:68  69 70 77 80 81 86
Input delete Key: 86
14 33 57 64 C:68  69 70 77 80 81
```

Вывод:
Для данной работы я создал новую процедуру удаление из СДП. Вывод немного изменил, а именно перед корневым узлом дописал С:, а так на обход слева на право.

Код программы:

```
// Двоичное дерево на C++
#include <stdlib.h>
#include <iostream>

using namespace std;

struct Vertex
```

```
{
	int value;
	int index;
	struct Vertex* left, * right;
};
Vertex* root;
void SDP(int val, Vertex *&q,int number)
{
	if (q == NULL)
	{//Нашли место для добавления
		q = new Vertex;
		(q)->left = NULL;
		(q)->right = NULL;
		(q)->value = val;
		(q)->index = number;
		return;
	}
	if ((q)->value > val)
		// Добавляем в левое поддерево
		SDP(val, q->left, number);
	else
		// Добавляем в правое поддерево
		SDP(val, q->right, number);
}
void deleteSDP(int key, Vertex*& root) {
	Vertex** p = &root;

	while (p != NULL) {
		if ((*p)->value < key) p = &((*p)->right);
		else
			if ((*p)->value > key) p = &((*p)->left);
			else
				break;
	}
	if (p != NULL) {
		Vertex* q = *p;
		if (q->left == NULL) *p = q->right;
		else if (q->right == NULL) *p = q->left;
		else {
			Vertex* r;
			Vertex* s;
			r = q->left;
			s = q;
			while (r->right) {
				s = r;
				r = r->right;
			}
			s->right = r->left;
			r->left = q->left;
			r->right = q->right;
			*p = r;
		}
		delete q;
	}
}
void printSDP(Vertex* &q) {
	if (q != NULL) {
		printSDP(q->left);
		if(q->index == 0)
			std::cout <<"C:" << q->value << "   ";
		else
			std::cout << q->value <<  " " ;
		printSDP(q->right);
	}
}
```

```
int main()
{
	srand(time(0));
	int m, t = 0, Key;
	const int huge = 21;
	int massiv[huge];
	for (int i = 0; i < huge - 1; i++) {
		massiv[i] = rand() % 100;
		for (int j = 0; j < i; j++) {
			if (massiv[i] == massiv[j]) {
				i--;
				break;;
			}
		}
}
	for ( int i = 0; i < huge - 1; i++){
		SDP(massiv[i], root, i);
	}
	printSDP(root);
	for (int i = 0; i < 10; i++) {
		std::cout <<std::endl<< "Input delete Key: ";
		std::cin >> Key;
		deleteSDP(Key, root);
		printSDP(root);
	}
	return 0;
}
```