**Частное учреждение образования**

**«Гродненский колледж бизнеса и права»**

**Лабораторная работа № 8**

**по дисциплине**

**«Структуры и алгоритмы обработки данных»**

**Тема:** Решение задач с применением красно-черных деревьев

для учащихся 2 курса специальности

2-40 01 01 «Программное обеспечение информационных технологий»

**ЛАБОРАТОРНАЯ РАБОТА № 8**

Тема: Решение задач с применением красно-черных деревьев.

Цель:

Образовательная**:**

* Обучить основным алгоритмам обхода графа и научиться решать задачи обхода графа на основе поиска в ширину и поиска в глубину,

Развивающая:

* научить анализировать алгоритмы обхода графа и научить решать задачи обхода графа на основе поиска в ширину и поиска в глубину,
* создать условия для развития способности четко формулировать свои мысли.

Воспитательная:

* воспитывать в обучающихся средствами урока уверенность в своих силах;

воспитывать сознательное и серьёзного отношения обучающихся к учебной дисциплине, убеждая их в том, что полученные знания пригодятся им в будущей деятельности.

Задачи: Освоение теоретического материала и выполнение индивидуального задания.

**ЗАДАЧИ**

Условие: Ввести 10-15 целых чисел и построить из них красно-черное дерево поиска.

Алгоритм: Предоставлен преподавателю в письменном виде.

Решение:

**program** Project1;

**uses** crt;

**type**

TData = Integer;

TPNode = ^TNode;

TNode = **record**

Data : TData;

PLeft, PRight : TPNode;.

**end**;

**procedure** ReadNode(**var** aPNode : TPNode; **const** aLr : Integer; aCode : String);

**var**

PNode : TPNode;

Data : TData;

Code : Integer;

S : String;

**begin**

**if** aPNode = nil **then**

aCode := '0'

**else**

aCode := aCode + '-' + IntToStr(aLr);

Data := 0;

**repeat**

Write('Узел ', aCode, ' Значение: ');

Readln(S);

Code := 0;

**if** S <> '' **then begin**

Val(S, Data, Code);

**if** Code <> 0 **then**

Writeln('Не верный ввод');

**end**;

**until** Code = 0;

//Если пользователь выбрал отмену при создании узла, то выходим.

**if** S = '' **then begin**

Writeln('ОТМЕНА');

**Exit**;

**end**;

//Создаём узел, записываем в него данные и в зависимости от флага aLr

//прикрепляем его к левой или к правой ветви родительского узла aPNode.

**New**(PNode);

PNode^.Data := Data;

PNode^.PLeft := nil;

PNode^.PRight := nil;

**if** aPNode <> nil **then begin**

**if** aLr = 1 **then**

**begin**

textcolor(4);

write(' Красный ');

aPNode^.PLeft := PNode

**end**

**else**

**begin**

textcolor(0);

write(' Чёрный ');

aPNode^.PRight := PNode;

**end**;

**end else**

**begin**

textcolor(4);

write('Красный');

aPNode := PNode;

**end**;

Writeln(' Узел создан');

//Рекурсивный вызов для создания левого узла.

ReadNode(PNode, 1, aCode);

//Рекурсивный вызов для создания правого узла.

ReadNode(PNode, 2, aCode);

**end**;

**var**

PTree : TPNode;

Data : TData;

Cmd, Code : Integer;

S : String;

**begin**

textbackground(15);

textcolor(0);

writeln(' 0');

writeln(' / \');

textcolor(4);

writeln(' 0-1 0-2');

writeln(' / \ / \');

textcolor(0);

writeln(' 0-1-1 0-1-2 0-2-1 0-2-2');

writeln(' / \ / \');

textcolor(4);

writeln(' 0-1-1-1 0-1-1-2 0-2-2-1 0-2-2-2');

PTree := nil;

**repeat**

//Меню

textcolor(0);

Writeln('Выберите действие:');

Writeln('1: Создание дерева');

Writeln('2: Выход');

Write('Введите команду: ');

Readln(Cmd);

**case** Cmd **of**

1:

**begin**

Writeln('Ввод дерева.');

Writeln('Что бы отменить нажмите Enter');

ReadNode(PTree, -1, '');

Writeln('Дерево СОздано')

**end**;

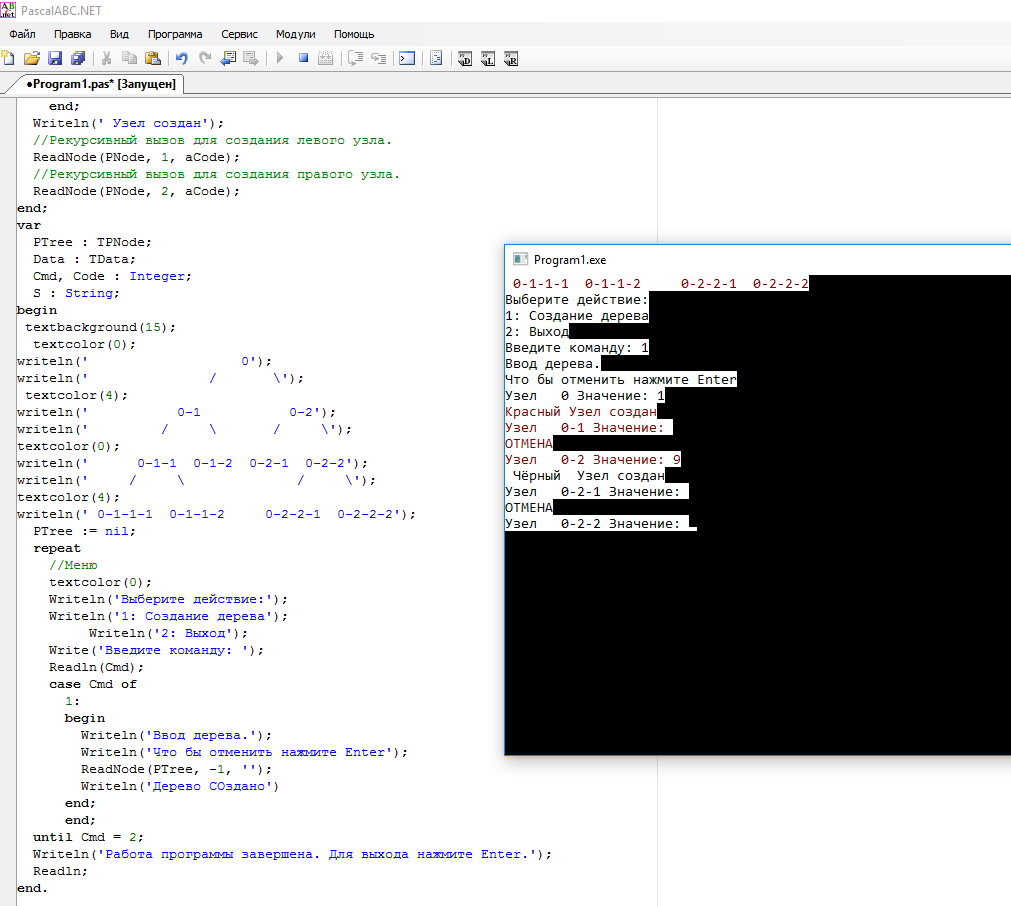
**end**;

**until** Cmd = 2;

Writeln('Работа программы завершена. Для выхода нажмите Enter.');

Readln;

**end**.



**ОТВЕТЫ НА КОНТРОЛЬНЫЕ ВОПРОСЫ**

1. Красный предок, красный "дядя": Ситуацию красный-красный иллюстрирует рис. У нового узла X предок и "дядя" оказались красными. Простое перекрашивание избавляет нас от красно-красного нарушения. После перекраски нужно проверить "дедушку" нового узла (узел B), поскольку он может оказаться красным. Обратите внимание на распространение влияния красного узла на верхние узлы дерева. В самом конце корень мы красим в черный цвет корень дерева. Если он был красным, то при этом увеличивается черная высота дерева.
2. Это свойство выполняется только в красно-черных деревьях.
3. 3 и 4. Во многих реализациях структуры дерева возможно, чтобы узел имел только одного потомка и листовой узел содержал данные. В этих предположениях реализовать красно-чёрное дерево возможно, но изменятся несколько свойств и алгоритм усложнится. По этой причине данная статья использует «фиктивные листовые узлы», которые не содержат данных и просто служат для указания, где дерево заканчивается. Эти узлы часто опускаются при графическом изображении, в результате дерево выглядит противоречиво с вышеизложенными принципами, но на самом деле противоречия нет. Следствием этого является то, что все внутренние (не являющиеся листовыми) узлы имеют два потомка, хотя один из них может быть нулевым листом. Свойство 5 гарантирует, что красный узел обязан иметь в качестве потомков либо два чёрных нулевых листа, либо два чёрных внутренних узла. Для чёрного узла с одним потомком нулевым листовым узлом и другим потомком, не являющимся таковым, свойства 3, 4 и 5 гарантируют, что последний должен быть красным узлом с двумя чёрными нулевыми листьями в качестве потомков.

5 и 6. Пускай высота дерева h, минимальное количество вершин N. Тогда:

для АВЛ-дерева N(h-1)+N(h-2)+1} N(h)=N(h-1)+N(h-2)+1. Поскольку N(0)=0} N(0)=0, N(1)=1} N(1)=1, N(h) растёт как последовательность Фибоначчи, следовательноN(h)=Theta{h} N(h)=\Theta lambda ^{h}, гдеlambda =5+1)lambda ={5}+1)/2 1,62   
для красно-чёрного дерева N(h)\ 2^{(h-1)/2}=Theta {2}^{h} N(h)2^(h-1)/2}=\Theta {2}^{h)   
Следовательно, при том же количестве листьев красно-чёрное дерево может быть выше АВЛ-дерева, но не более чем в лямбда.