|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
| Федеральное государственное бюджетное  образовательное учреждение высшего образования «Новосибирский государственный технический университет» | | | |
|  | | | |
| Кафедра прикладной математики | | | |
|  | | | |
| Практическое задание № 3 | | | |
| по дисциплине «Структуры данных и алгоритмы» | | | |
| **ДЕРЕВЬЯ** | | | |
|  | | | |
|  | Группа | ПМ-95 |
| Бригада | 8 |
| Алгоритм | Костыря Сергей |
| Реализация | Краус Александр |
| Тестирование | Лаворов Дмитрий |
| Преподаватель | Тракимус Юрий Викторович |
| Дата | 28.05.2020 |
|  | | | |
| Новосибирск | | | |

**1. Условие задачи**

В заданном бинарном дереве подсчитать число его листьев и напечатать их значения: при прямом обходе дерева.

**2. Анализ задачи**

Метод решения: при решении используется прямой обход бинарного дерева.

2.1 Создается бинарное дерево struct treeNode.

2.2 Начинается прямой обход (preOrder) дерева с помощью подпрограммы.

2.3 Считывается количество листьев и записывается в переменную Leaves.

**3. Структура входных и выходных данных**

На вход подаются рандомные числа находящиеся в дереве. И выводятся количество листьев после обхода.

**4. Текст программы**

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <conio.h>

#include <locale.h>

struct treeNode {

struct treeNode\* leftPtr; //указатель на левое поддерево

int data; //значение в узле

struct treeNode\* rightPtr; //указывает на правое поддерево

};

typedef struct treeNode TreeNode; //синоним

typedef TreeNode\* TreeNodePtr; //синоним

//прототипы

void insertNode(TreeNodePtr\* treePtr, int value);

int preOrder(TreeNodePtr treePtr);

int Leaves(TreeNodePtr rootPtr);

int main(void)

{

setlocale(LC\_CTYPE, "Russian");

int i; //счетчик для цикла 1-10

int item; //переменная для случайных значений

TreeNodePtr rootPtr = NULL;

TreeNodePtr LeavesPtr = NULL;

srand(time(NULL));

puts("Число, помещаемое в дерево:");

for (i = 1; i <= 12; i++) {

item = rand() % 99;

printf("%3d", item);

insertNode(&rootPtr, item);

}

//обход дерева с предварительной выборкой(прямой) preOrder

puts("\n\nПредварительный обход:");

printf("\nЧисло ветвей: %3d \n", preOrder(rootPtr));

printf("Число листьев: %3d \n", Leaves(rootPtr));

system("pause");

return 0; //показывает завершение

}

//вставить узел в дерево

void insertNode(TreeNodePtr\* treePtr, int value)

{

//если дерево пусто

if (\*treePtr == NULL) {

\*treePtr = (TreeNode\*)malloc(sizeof(TreeNode));

//если память выделана,присвоить данные

if (\*treePtr != NULL) {

(\*treePtr)->data = value;

(\*treePtr)->leftPtr = NULL;

(\*treePtr)->rightPtr = NULL;

}

else {

printf("%d не вставленно. Не достаточно памяти. \n", value);

free(\*treePtr);

}

}

//дерево не пусто

else {

//вставляемые данные меньше, чем в текущем узле

if (value < (\*treePtr)->data) {

insertNode(&((\*treePtr)->leftPtr), value);

}

//вставляемые данные больше, чем в текущем узле

else if (value > (\*treePtr)->data) {

insertNode(&((\*treePtr)->rightPtr), value);

}

else {//дубликаты инорируются

printf("%s", " - дубликат");

}

}

}

//обход дерева с предварительной выборкой

int preOrder(TreeNodePtr treePtr)

{

static int nRet = 0;

//если дерево не пусто обойти его

if (treePtr != NULL) {

++nRet;

printf("%3d", treePtr->data);

preOrder(treePtr->leftPtr);

preOrder(treePtr->rightPtr);

}

//\_getch();

return nRet;

}

//подсчет листьев

int Leaves(TreeNodePtr rootPtr)

{

static int nRet = 0;

if (rootPtr != NULL)

{

if ((rootPtr->leftPtr == NULL) && (rootPtr->rightPtr == NULL))

nRet++;

Leaves(rootPtr->leftPtr);

Leaves(rootPtr->rightPtr);

}

return nRet;

}

**5. Набор тестов**

|  |  |  |
| --- | --- | --- |
| **№** | **Дано** | **Результат** |
| **1** | Число, помещаемое в дерево:  80 4 68 71 2 50 77 63 70 11 6 21  Предварительный обход:  80 4 2 68 50 11 6 21 63 71 70 77 | Число ветвей: 12  Число листьев: 6 |
| **2** | Число, помещаемое в дерево:  72 9 34 71 36 13 10 68 79 21 23 66  Предварительный обход:  72 9 34 13 10 21 23 71 36 68 66 79 | Число ветвей: 12  Число листьев: 4 |
| **3** | Число, помещаемое в дерево:  94 29 5 35 41 74 37 62 97 37 - дубликат 93 13  Предварительный обход:  94 29 5 13 35 41 37 74 62 93 97 | Число ветвей: 11  Число листьев: 5 |