Лабораторна робота

Тема: Динамічні структури даних. Дерева

Студента групи 141

Бужака Андрія

Варіант №4

Завдання: Розробити програму, яка створює бінарне дерево , елементами якого є дійсні числа, і обчислює добуток елементів цього дерева.

Код програми:

**Файл UMain.cpp**

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "UMain.h"

#include "UTree.h"

#pragma package(smart\_init)

#pragma resource "\*.dfm"

#include <vector.h>

//---------------------------------------------------------------------------

TfrmMain \*frmMain;

//---------------------------------------------------------------------------

\_\_fastcall TfrmMain::TfrmMain(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

int N=0;

TTree \*Tree=NULL;

//---------------------------------------------------------------------------

void \_\_fastcall TfrmMain::FormCreate(TObject \*Sender)

{

frmMain->stgElements->Cells[0][0]="Елементи";

}

//---------------------------------------------------------------------------

void \_\_fastcall TfrmMain::btnSizeClick(TObject \*Sender)

{

N=StrToInt(edtSize->Text);

frmMain->stgElements->RowCount=N+1;

frmMain->stgTree->RowCount=N;

for(int i=1;i<N;i++)

frmMain->stgTree->Cells[0][i]=IntToStr(i);

frmMain->stgTree->Cells[0][0]="Root";

CleanStringGrid(stgTree);

}

//---------------------------------------------------------------------------

void \_\_fastcall TfrmMain::btnDoClick(TObject \*Sender)

{

CleanStringGrid(stgTree);

for(int i=1;i<=N;i++)

AddNode(Tree,StrToFloat(frmMain->stgElements->Cells[0][i]));

PrintTree(Tree,0,stgTree);

frmMain->stgTree->RowCount=TreeHeight(Tree);

frmMain->edtDobutok->Text=FloatToStr(DobutokNodes(Tree));

DeleteTree(Tree);

}

//---------------------------------------------------------------------------

void \_\_fastcall TfrmMain::btnRandomClick(TObject \*Sender)

{

int i;

for(i=1;i<=N;i++)

frmMain->stgElements->Cells[0][i]=IntToStr(RandomValue(StrToInt(edtRandomLeft->Text),StrToInt(edtRandomRight->Text)));

}

//---------------------------------------------------------------------------

void \_\_fastcall TfrmMain::stgTreeDrawCell(TObject \*Sender, int ACol,

int ARow, TRect &Rect, TGridDrawState State)

{

int x, y;

x=Rect.Left+(Rect.Width()-stgTree->Canvas->TextWidth(stgTree->Cells[ACol][ARow]))/2;

y=Rect.Top+(Rect.Height()-stgTree->Canvas->TextHeight(stgTree->Cells[ACol][ARow]))/2;

if(ACol!=0)

{

stgTree->Canvas->FillRect(Rect);

stgTree->Canvas->TextOut(x,y,stgTree->Cells[ACol][ARow]);

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TfrmMain::stgElementsDrawCell(TObject \*Sender, int ACol,

int ARow, TRect &Rect, TGridDrawState State)

{

int x, y;

x=Rect.Left+(Rect.Width()-stgElements->Canvas->TextWidth(stgElements->Cells[ACol][ARow]))/2;

y=Rect.Top+(Rect.Height()-stgElements->Canvas->TextHeight(stgElements->Cells[ACol][ARow]))/2;

stgElements->Canvas->FillRect(Rect);

stgElements->Canvas->TextOut(x,y,stgElements->Cells[ACol][ARow]);

}

//---------------------------------------------------------------------------

**Файл UTree.cpp**

//---------------------------------------------------------------------------

#pragma hdrstop

#include "UTree.h"

#pragma package(smart\_init)

#include <vector.h>

//---------------------------------------------------------------------------

void AddNode(TTree \*&T,double d)

{

if(T==NULL)

{

T=new TTree;

T->left=NULL;

T->right=NULL;

T->data=d;

}

else

if(d<T->data)

AddNode(T->left,d);

else

AddNode(T->right,d);

}

//---------------------------------------------------------------------------

void DeleteTree(TTree \*&T)

{

if(T!=NULL)

{

DeleteTree(T->left);

DeleteTree(T->right);

delete T;

T=NULL;

}

}

//---------------------------------------------------------------------------

void PrintTree(TTree \*T,int Rang,TStringGrid \*&stg)

{

if(T!=NULL)

{

PrintTree(T->left,Rang+1,stg);

stg->Cells[stg->ColCount][Rang]=FloatToStr(T->data);

stg->ColCount++;

PrintTree(T->right,Rang+1,stg);

}

}

//---------------------------------------------------------------------------

void CleanStringGrid(TStringGrid \*&stg)

{

int i,j;

for(i=stg->FixedCols;i<stg->ColCount;i++)

for(j=stg->FixedRows;j<stg->RowCount;j++)

stg->Cells[i][j]="";

stg->ColCount=2;

}

//---------------------------------------------------------------------------

int TreeHeight(TTree \*T)

{

int k=0;

if(T!=NULL)

return max(TreeHeight(T->left),TreeHeight(T->right))+1;

return k;

}

//---------------------------------------------------------------------------

double DobutokNodes(TTree \*T)

{

double k=1;

if(T!=NULL)

return DobutokNodes(T->left)\*DobutokNodes(T->right)\*T->data;

return k;

}

//---------------------------------------------------------------------------

int RandomValue(int l,int r)

{

return random(r-l+1)+l;

}

//---------------------------------------------------------------------------

**Файл UTree.h**

//---------------------------------------------------------------------------

#ifndef UTreeH

#define UTreeH

//---------------------

#include <Classes.hpp>

#include <Grids.hpp>

//---------------------------------------------------------------------------

typedef struct OTree

{

double data;

OTree \*left;

OTree \*right;

}TTree;

//---------------------------------------------------------------------------

void AddNode(TTree \*&T,double d);

void DeleteTree(TTree \*&T);

void PrintTree(TTree \*T,int Rang,TStringGrid \*&stg);

void CleanStringGrid(TStringGrid \*&stg);

int TreeHeight(TTree \*T);

double DobutokNodes(TTree \*T);

int RandomValue(int l,int r);

//---------------------------------------------------------------------------

#endif

