Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

Национальный исследовательский университет «МЭИ»

Типовой расчет «Моделирование типов»

Вариант 24

Задание выполнил:

Черкасов Александр Андреевич

Студент группы A-06-19

Проверил:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оценка:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Замечания:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва 2019 НИУ «МЭИ»

**Часть 1:**

**Задание:**

**Смоделировать** двумя способами новый тип «Бинарное дерево»:

1) на основе ссылочного типа «Нелинейный связный список»;

2) на основе динамического массива или типизированного/двоичного файла (на выбор студента).

Для этого создать модуль и описать новый тип данных и его структуру (на русском языке и на Delphi/C, без классов! Только записи/структуры и массив/файл) и базовые операции:

- проверка на пустоту дерева (поддерева, начинающегося с текущей вершины);

- создание пустого дерева или дерева из одного элемента (как удобнее);

- сделать текущим (по адресу в списке или индексу в массиве) корень дерева;

- сделать текущим левое поддерево (от текущей вершины);

- сделать текущим правое поддерево (от текущей вершины);

- сделать текущим родительскую вершину для текущей вершины (для удобства можно в способе 1 ввести третью связь – обратную связь с родительской вершиной, чтобы не искать ее по всему дереву – Трёхсвязный список);

- узнать значение текущей вершины;

- изменить значение текущей вершины;

- создать левое поддерево для текущей вершины;

- создать правое поддерево для текущей вершины;

**Выбранный 2й способ**: бинарный файл

Использованные структуры:

struct date //Дата

{

int day; //День

int month; //Месяц

int year; //Год

};

struct data //Данные

{

char\* FName; //Имя

char\* SName; //Фамилия

char\* LName; //Отчество

date dBirth; //Дата рождения

date dDeath; //Смерти (опционально)

char\* BPlace; //Место рождения(опционально)

};

Специфическая структура для дерева на основе списка

struct Node //Структура одной ветви

{

data info; //Данные

Node \*nextL; //Указатель на левый элемент

Node \*nextR; //Указатель на правый элемент

Node \*prev; //Указатель на предыдущий элемент

};

**Основные функции:**

**Названия:**

Список:

//Указатель на ветвь называем NodePtr

typedef Node\* NodePtr;

Файл:

//"Указатель" на ветвь содержит текущую позицию и указатель на файл, к которому принадлежит

typedef std::pair<int,FILE\*> NodePtr;

**Инициализация:**

Список:

void Init()

{

//Создаем на корне пустые элементы и присваиваем неотсортированному стоп-знак

head = NewNode(nullptr);

headKey = NewNode(nullptr);

head->info={(char\*)"\*\*"};

}

Файл:

void Init()

{

//Открываем соответствующие файлы

pTreeUnsorted = fopen("C:\\Users\\Public\\Documents\\treeUnsorted.bin","wb+");

pTreeSorted = fopen("C:\\Users\\Public\\Documents\\treeSorted.bin","wb+");

//Инициализируем корни

head = std::make\_pair(1,pTreeUnsorted);

headKey = std::make\_pair(1,pTreeSorted);

//Записываем в корни пустые данные

replDataM(&head,nullData);

replDataM(&headKey,nullData);

}

**Новый элемент:**

Список:

//Создание новой ветви, в зависимости от символа будет правым или левым или же "основой"

NodePtr NewNode(Node \*\*CurrentPosition, char Pos='0')

{

Node \*newNode = new Node;

newNode->nextL=nullptr;

newNode->nextR=nullptr;

//Создание пустых связей

//Проверка на наличие "предшественника"

newNode->prev = (CurrentPosition == nullptr ? nullptr : (\*CurrentPosition));

//Добавление связи в зависимости от того, куда неообходимо добавить новую ветвь

if (Pos=='L') (\*CurrentPosition)->nextL=newNode;

if (Pos=='R') (\*CurrentPosition)->nextR=newNode;

//Записываем пустые данные

newNode->info=nullData;

return newNode;

}

Файл:

//Создание новой ветви, в зависимости от символа будет правым или левым или же "основой"

NodePtr NewNode(NodePtr \*CurrentPosition, char Pos='0')

{

if (Pos=='L') return curL(\*CurrentPosition); //Возвращение «левой» позиции

else if (Pos=='R') return curR(\*CurrentPosition); //Возвращение «правой» позиции

else return (\*CurrentPosition); //Возвращение самого себя

}

**Замена данных:**

**Текущая ветвь**

Список:

void replDataM(NodePtr \*cur,data inf){

(\*cur)->info=inf;

}

Файл:

void replDataM(NodePtr \*cur,data inf) {

fseek((\*cur).second, (\*cur).first \* sizeof(data), SEEK\_SET); //Встаем на позицию ветви

fwrite(&inf, sizeof(data), 1, (\*cur).second);

}

**Левая ветвь**

Список:

void replDataL(NodePtr \*cur,data inf){

(\*cur)->nextL = NewNode(cur); //Создание элемента слева

(\*cur)->nextL->info=inf;

}

Файл:

void replDataL(NodePtr \*cur,data inf) {

fseek((\*cur).second, (\*cur).first \* 2 \* sizeof(data), SEEK\_SET); //Встаем на позицию ветви

fwrite(&inf, sizeof(data), 1, (\*cur).second);

}

**Правая ветвь**

Список:

void replDataR(NodePtr \*cur,data inf){

(\*cur)->nextR = NewNode(cur); //Создание элемента справа

(\*cur)->nextR->info=inf;

}

Файл:

void replDataR(NodePtr \*cur,data inf) {

fseek((\*cur).second, ((\*cur).first \* 2 + 1) \* sizeof(data), SEEK\_SET); //Встаем на позицию ветви

fwrite(&inf, sizeof(data), 1, (\*cur).second);

}

**Сделать текущим левое поддерево**

Список:

NodePtr curL(NodePtr cur){

return cur->nextL; //Возврат указателя на левое поддерево

}

Файл:

NodePtr curL(NodePtr cur){

return std::make\_pair(cur.first\*2,cur.second); //Возврат пары левая позиция, указатель на файл

}

**Сделать текущим правое поддерево**

Список:

NodePtr curR(NodePtr cur){

return cur->nextR; //Возврат указателя на правое поддерево

}

Файл:

NodePtr curR(NodePtr cur){

return std::make\_pair(cur.first\*2+1,cur.second); //Возврат пары правая позиция, указатель на файл

}

**Сделать текущим родительскую вершину:**

**Генеалогическое дерево**

Список:

//Указатель на корень Геологического дерева

NodePtr getStartUnsorted(){

return head;

}

Файл:

//Указатель на корень Геологического дерева

NodePtr getStartUnsorted(){

return head;

}

**Отсортированное дерево**

Список:

//Указатель на корень отсортированного дерева

NodePtr getStartSorted(){

return headKey;

}

Файл:

//Указатель на корень отсортированного дерева

NodePtr getStartSorted(){

return headKey;

}

**Удаление ветви**

Список:

//Уничтожение ветви или целого дерева

void destroy(NodePtr \*cur)

{

if (!(\*cur)->info.FName) { //Проверка, удалили ли данные раньше

/\*

\* Нужно в основном для отсортированного дерева

\* т.к. есть элемнты добавленные вручную, а остальные

\* удалены ранее в генеологическом дереве

\*/

delete[]((\*cur)->info.FName);

delete[]((\*cur)->info.SName);

delete[]((\*cur)->info.LName);

delete[]((\*cur)->info.BPlace);

(\*cur)->info.FName=nullptr; //для возможности определить был ли удален элемент ранее

}

//Если есть что-то слева или справа - уничтожаем

if ((\*cur)->nextL!= nullptr)

destroy(&((\*cur)->nextL));

if ((\*cur)->nextR!= nullptr)

destroy(&((\*cur)->nextR));

//Если не голова - удаляем текущий элемент и соответствующую связь

if (\*cur!=head and \*cur!=headKey){

Node \*temp = (\*cur)->prev;

if (temp->nextL==(\*cur)) temp->nextL=nullptr;

if (temp->nextR==(\*cur)) temp->nextR=nullptr;

delete [](\*cur);

(\*cur) = temp;

}

else {

//Очищаем голову и отдаем nullptr

delete [](\*cur);

(\*cur) = nullptr;

}

}

Файл:

void destroy(NodePtr \*cur) {

//Если уничтожаем корень, то просто закрываем файл и уничтожаем его

if (\*cur == head) {

fclose(pTreeUnsorted);

remove("C:\\Users\\Public\\Documents\\treeUnsorted.bin");

} else if (\*cur == headKey) {

fclose(pTreeSorted);

remove("C:\\Users\\Public\\Documents\\treeSorted.bin");

} else {

//Иначе просто записываем в нужный файл на соответствующее место пустые данные

fseek((\*cur).second, (\*cur).first \* sizeof(data), SEEK\_SET);

fwrite(&nullData, sizeof(data), 1, (\*cur).second);

NodePtr nextL = std::make\_pair((\*cur).first \* 2, (\*cur).second);

NodePtr nextR = std::make\_pair((\*cur).first \* 2 + 1, (\*cur).second);

//и переходим влево и вправо

if (!IsEmpty(nextL)) destroy(&nextL);

if (!IsEmpty(nextR)) destroy(&nextR);

}

}

**Часть 2:**

**Задание:**

**Создать приложение** (консольное, на C или Delphi), позволяющее выбрать и выполнить следующие действия над деревом (только с помощью выше перечисленных базовых операций! Без прямого обращения к элементам списка/массива/файла):

- удалить дерево (освободить память);

- создать упорядоченное по ключу (по возрастанию/убыванию одного или нескольких полей записи/структуры) бинарное дерево из текстового файла;

- создать неупорядоченное бинарное дерево (генеалогическое дерево) из текстового файла (Корень – Левое\_поддерево – Правое\_поддерево, \*\* – признак пустого поддерева);

- добавить один элемент, со значениями введенными в интерактивном режиме (с клавиатуры) в упорядоченное дерево по ключу (по возрастанию/убыванию одного или нескольких полей записи/структуры) для построения упорядоченного дерева; например, в ниже изображенное дерево добавляются числа 6,3,4,2,6,9 по возрастанию (неубыванию в случае добавления одинаковых значений);

- найти решение задачи из списка заданий по вариантам в конце этого файла (стр.16);

- вывести дерево 2-4мя разными способами: (Корень – Левое\_поддерево – Правое\_поддерево, Левое\_поддерево – Корень – Правое\_поддерево, Левое\_поддерево – Правое\_поддерево – Корень; графически(по желанию). \*\* – признак пустого поддерева);

После каждой операции выводить содержимого дерева в текстовый лог-файл для контроля за состоянием бинарного дерева после каждой операции с начала и до конца работы программы первым или последним из вышеуказанных 4-х способов.

Для каждого человека в генеалогическом дереве указаны следующие сведения: Фамилия, Имя, Отчество, дата рождения, а также при необходимости дата смерти и место рождения. **Для каждой вершины левое поддерево отвечает за отца, правое – за мать**

24. а) Создав соответствующее упорядоченное дерево, создать на год (с января по декабрь) план поминок усопших родственников. б) По генеалогическому дереву определить, есть ли в Вашем роду кто-нибудь, у кого обоих дедушек звали Иванами

**Формат входного файла:**

Данные

<FName>

<SName>

<LName>

**<**dBitrh.day**>**

**<**dBitrh.month**>**

**<**dBitrh.year**>**

**<**dDeath.day**>**

**<**dDeath.month**>**

**<**dDeath.year**>**

**<**BPlace>

При желании перейти на другую ветвь ставить стоп-знак

<\*\*>

**Описание меню**

**'v'** - вывести деревья на экран

Предлагает на выбор 3 возможных способа вывода дерева; функции имеют вид:

**void (NodePtr TreeHead,FILE\* pfile,int depth)**

NodePtr TreeHead – указатель на текущее положение в дереве;

FILE \*pfile – указатель на поток, в который записывается инфомация

int depth – глубина текущего положения в дереве

**'p'** - вывести план поминок

Функция имеет вид:

**void (NodePtr TreeHead, FILE \*pfile)**

NodePtr TreeHead – указатель на текущее положение в дереве;

FILE \*pfile – указатель на поток, в который записывается инфомация

**'a'** - добавить к дереву по ключу

Сперва вызывается

**data readData(char\* FName, FILE\* FileIn)**

char\* FName – Имя элемента. Вынесена в параметр, из-за использования при считывании данных с текстового файла

FILE\* FileIn – поток, с которого считываются данные

Возвращает данные в формате структуры data

Если будут ошибки с форматом даты или другая надпись вместо даты – вызов кода ошибки 2

Потом

**void AddToTreeKey(data currentData)**

data CurrentData -данные, которые нужно записать в дерево

**'t'** - выполнить поиск человека, у которого обоих дедушек зовут Иванами

Функция имеет вид:

**int (NodePtr CurrentPosition)**

NodePtr CurrentPosition – текущая позиция в дереве.

Возвращает 1, если найден человек, у которого обоих дедушек зовут Иванами

Иначе 0

**'r'** - пересоздать деревья

Сперва вызывается для обоих деревье

**void remove**(NodePtr \*cur)

NodePtr \*cur – «указатель на указатель» текущего положения дерева, которого нужно очистить

Снова инициализируются деревья

**void Init**()

Вызываются создания деревьев

**void MakeTree**(char\* filename)

char\* filename – имя входного файла

**void MakeTreeKey**()

**'x'** - завершение программы

Переключает флаг end основного цикла на 1 (цикл работает на !end)

**Функциональные тесты:**

Ремарка для удобства восприятия, все деревья выводятся в режиме Лево-Корень-Право

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Входные данные** | **Ожидаемый результат** | **Смысл теста** |
| **Аномалии** | | | |
| **0** | В параметрах при запуске указать неправильную папку логов | Сообщение «Невозможно создать Лог-файл»  Завершение программы | Проверка поведении программы на невозможность создать лог-файл |
| **1** | В параметрах при запуске указать не существующий входной файл | Сообщение «Невозможно прочитать входной файл»  Завершение программы | Проверка поведении программы на невозможность прочитать входные данные |
| **2** | Человек  Случайный  Корень  Точно  11  2000  1  2  2060  Doma  \*\*  \*\* | Сообщение «Неверный формат даты»  Завершение программы | Проверка поведении программы на неверный формат входных данных  День в дате рождения |
| **2.1** | Человек  Случайный  Корень  -10  11  2000  1  2  2060  Doma  \*\*  \*\* | Сообщение «Неверный формат даты»  Завершение программы | Проверка поведении программы на неверный формат входных данных  Отрицательное число в дне даты рождения |
| **2.2** | Человек  Случайный  Корень  35  11  2000  1  2  2060  Doma  \*\*  \*\* | Сообщение «Неверный формат даты»  Завершение программы | Проверка поведении программы на неверный формат входных данных  Число больше 31 в дне даты рождения |
| **3** | Человек  Случайный  Корень  11  Точно  2000  1  2  2060  Doma  \*\*  \*\* | Сообщение «Неверный формат даты»  Завершение программы | Проверка поведении программы на неверный формат входных данных  Месяц в дате рождения |
| **3.1** | Человек  Случайный  Корень  11  -1  2000  1  2  2060  Doma  \*\*  \*\* | Сообщение «Неверный формат даты»  Завершение программы | Проверка поведении программы на неверный формат входных данных  Отрицательное число в месяце даты рождения |
| **3.2** | Человек  Случайный  Корень  11  30  2000  1  2  2060  Doma  \*\*  \*\* | Сообщение «Неверный формат даты»  Завершение программы | Проверка поведении программы на неверный формат входных данных  Число больше 12 в месяце даты рождения |
| **4** | Человек  Случайный  Корень  11  2  Точно  1  2  2060  Doma  \*\*  \*\* | Сообщение «Неверный формат даты»  Завершение программы | Проверка поведении программы на неверный формат входных данных  Год в дате рождения |
| **4.1** | Человек  Случайный  Корень  1  2  -2000  11  2  2060  Doma  \*\*  \*\* | Сообщение «Неверный формат даты»  Завершение программы | Проверка поведении программы на неверный формат входных данных  Отрицательный год в дате рождения |
| **5** | Человек  Случайный  Корень  1  2  2000  Точно  11  2000  Doma  \*\*  \*\* | Генеалогическое дерево  -- Случайный Человек Корень Родился 1.2.2000 Родился в Doma  ----------------  Упорядоченное дерево  \*\* | Проверка поведении программы на неверный формат входных данных  День в дате смерти |
| **5.1** | Человек  Случайный  Корень  10  11  2000  -1  2  2060  Doma  \*\*  \*\* | Сообщение «Неверный формат даты»  Завершение программы | Проверка поведении программы на неверный формат входных данных  Отрицательное число в дне даты смерти |
| **5.2** | Человек  Случайный  Корень  10  11  2000  45  2  2060  Doma  \*\*  \*\* | Сообщение «Неверный формат даты»  Завершение программы | Проверка поведении программы на неверный формат входных данных  Число больше 31 в дне даты смерти |
| **6** | Человек  Случайный  Корень  1  2  2000  11  Точно  2000  Doma  \*\*  \*\* | Генеалогическое дерево  -- Случайный Человек Корень Родился 1.2.2000 Родился в Doma  ----------------  Упорядоченное дерево  \*\* | Проверка поведении программы на неверный формат входных данных  Месяц в дате смерти |
| **6.1** | Человек  Случайный  Корень  11  1  2000  1  -2  2060  Doma  \*\*  \*\* | Сообщение «Неверный формат даты»  Завершение программы | Проверка поведении программы на неверный формат входных данных  Отрицательное число в месяце даты смерти |
| **6.2** | Человек  Случайный  Корень  11  12  2000  1  20  2060  Doma  \*\*  \*\* | Сообщение «Неверный формат даты»  Завершение программы | Проверка поведении программы на неверный формат входных данных  Число больше 12 в месяце даты смерти |
| **7** | Человек  Случайный  Корень  1  2  2000  11  2  Точно  Doma  \*\*  \*\* | Генеалогическое дерево  -- Случайный Человек Корень Родился 1.2.2000 Родился в Doma  ----------------  Упорядоченное дерево  \*\* | Проверка поведении программы на неверный формат входных данных  Год в дате смерти |
| **7.1** | Человек  Случайный  Корень  1  2  2000  11  2  -200  Doma  \*\*  \*\* | Генеалогическое дерево  -- Случайный Человек Корень Родился 1.2.2000 Родился в Doma  ----------------  Упорядоченное дерево  \*\* | Проверка поведении программы на неверный формат входных данных  Отрицательный год в дате смерти |
| **8** | Человек  Корневой  Первый  1  3  1990  11  12  2040  --  \*\*  \*\*  Человек  Корневой  Первый  1  3  1990  11  12  2040  --  \*\*  \*\* | Сообщение «Попытка записи в существующий блок»  Завершение программы | Попытка записать в уже созданное генеалогическое дерево |
| **Построение дерева** | | | |
| **9** | \*\* | Генеалогическое дерево  \*\*  ----------------  Упорядоченное дерево  \*\*  Task  Невозможно выполнить поиск на пустом дереве  Нет такого человека, у которого обоих дедушек зовут Иванами | Пустые данные |
| **10** | Человек  Случайный  Корень  1  2  2000  11  2  2060  Doma  \*\*  \*\* | - Генеалогическое дерево  -- Случайный Человек Корень Родился 1.2.2000 Умер 11.2.2060 Родился в Doma  ----------------  Упорядоченное дерево  -- Случайный Человек Корень Родился 1.2.2000 Умер 11.2.2060 Родился в Doma | Один элемент с датой смерти |
| **11** | Человек  Случайный  Корень  1  2  2000  0  0  0  Doma  \*\*  \*\* | Генеалогическое дерево  -- Случайный Человек Корень Родился 1.2.2000 Умер 11.2.2060 Родился в Doma  ----------------  Упорядоченное дерево  \*\* | Один элемент без даты смерти |
| **12** | Человек  Тестовый  Корневой  11  11  2000  10  12  2060  Городе  Человек  Тестовый  Слева  16  11  2010  1  1  12  Городе  \*\*  \*\*  \*\* | Генеалогическое дерево  -- Тестовый Человек Слева Родился 16.11.2010 Умер 1.1.12 Родился в Городе  -- Тестовый Человек Корневой Родился 11.11.2000 Умер 10.12.2060 Родился в Городе  ----------------  Упорядоченное дерево  -- Тестовый Человек Слева Родился 16.11.2010 Умер 1.1.12 Родился в Городе  -- Тестовый Человек Корневой Родился 11.11.2000 Умер 10.12.2060 Родился в Городе | Построение левой ветки |
| **13** | Человек  Тестовый  Корневой  11  11  2000  1  2  2060  Городе  \*\*  Человек  Тестовый  Справа  16  11  2010  10  11  12  Городе  \*\*  \*\* | Генеалогическое дерево  -- Тестовый Человек Корневой Родился 11.11.2000 Умер 1.2.2060 Родился в Городе  -- Тестовый Человек Справа Родился 16.11.2010 Умер 10.11.12 Родился в Городе  ----------------  Упорядоченное дерево  -- Тестовый Человек Корневой Родился 11.11.2000 Умер 1.2.2060 Родился в Городе  -- Тестовый Человек Справа Родился 16.11.2010 Умер 10.11.12 Родился в Городе | Построение правой ветки |
| **Добавление элемента** | | | |
| **14** | Входной файл  \*\*  Консоль:  Человек  Корневой  Первый  1  3  1990  11  12  2040  -- | Генеалогическое дерево  \*\*  ----------------  Упорядоченное дерево  -- Человек Корневой Первый Родился 1.3.1990 Умер 11.12.2040  План поминок:  Человек Корневой Первый | Умер 11.12.2040 | Добавление элемента к пустому дереву |
| **15** | Входной файл  Консоль:  Человек  Корневой  Первый  1  3  1990  11  12  2040  --  \*\*  \*\*  Консоль:  Человек  Добавленный  Второй  1  3  1990  11  11  2040 | Генеалогическое дерево  \*\*  ----------------  Упорядоченное дерево  -- Человек Добавленный Второй Родился 1.3.1990 Умер 11.11.2040  -- Человек Корневой Первый Родился 1.3.1990 Умер 11.12.2040  План поминок:  Человек Добавленный Второй | Умер 11.11.2040  Человек Корневой Первый | Умер 11.12.2040 | Более ранняя дата смерти  Добавлен в левую подветвь |
| **16** | Входной файл  Консоль:  Человек  Корневой  Первый  1  3  1990  11  12  2040  --  \*\*  \*\*  Консоль:  Человек  Добавленный  Второй  1  3  1990  11  12  2040 | Генеалогическое дерево  \*\*  ----------------  Упорядоченное дерево  -- Человек Корневой Первый Родился 1.3.1990 Умер 11.12.2040  -- Человек Добавленный Второй Родился 1.3.1990 Умер 11.12.2040  План поминок:  Человек Корневой Первый | Умер 11.12.2040  Человек Добавленный Второй | Умер 11.12.2040 | Такая же дата смерти  Добавлен в правую подветвь |
| **17** | Входной файл  Консоль:  Человек  Корневой  Первый  1  3  1990  11  12  2040  --  \*\*  \*\*  Консоль:  Человек  Добавленный  Второй  1  3  1990  30  12  2040 | Генеалогическое дерево  \*\*  ----------------  Упорядоченное дерево  -- Человек Корневой Первый Родился 1.3.1990 Умер 11.12.2040  -- Человек Добавленный Второй Родился 1.3.1990 Умер 30.12.2040  План поминок:  Человек Корневой Первый | Умер 11.12.2040  Человек Добавленный Второй | Умер 30.12.2040 | Более поздняя дата смерти  Добавлен в правую подветвь |
| **Поиск человека, у которого обоих дедушек зовут Иванами** | | | |
| **18** | Человек  Случайный  Корень  1  2  2000  11  2  2060  Doma  \*\*  \*\* | Генеалогическое дерево  -- Человек Случайный Корень Родился 1.2.2000 Умер 11.2.2060  Task:  Нет такого человека, у которого обоих дедушек зовут Иванами | Поиск с недостающим количеством элементов |
| **19** | Человек  Тестовый  Корневой  11  11  2000  0  0  0  --  Человек  Тестовый  Слева  16  11  1980  0  0  0  --  Олег  Тестовый  СлеваЛева  12  5  1950  0  0  0  --  \*\*  \*\*  \*\*  Человек  Тестовый  Справа  16  10  1980  0  0  0  --  Олег  Тестовый  СлеваПрава  12  7  1951  0  0  0  --  \*\*  \*\*  \*\* | Генеалогическое дерево  -- Тестовый Олег СлеваЛева Родился 12.5.1950  -- Тестовый Человек Слева Родился 16.11.1980  -- Тестовый Человек Корневой Родился 11.11.2000  -- Тестовый Олег СлеваПрава Родился 12.7.1951  -- Тестовый Человек Справа Родился 16.10.1980  Task  Нет такого человека, у которого обоих дедушек зовут Иванами | Нет нужного элемента |
| **20** | Человек  Тестовый  Корневой  11  11  2000  0  0  0  --  Человек  Тестовый  Слева  16  11  1980  0  0  0  --  Иван  Тестовый  СлеваЛева  12  5  1950  0  0  0  --  \*\*  \*\*  \*\*  Человек  Тестовый  Справа  16  10  1980  0  0  0  --  Иван  Тестовый  СлеваПрава  12  7  1951  0  0  0  --  \*\*  \*\*  \*\* | Генеалогическое дерево  -- Тестовый Иван СлеваЛева Родился 12.5.1950  -- Тестовый Человек Слева Родился 16.11.1980  -- Тестовый Человек Корневой Родился 11.11.2000  -- Тестовый Иван СлеваПрава Родился 12.7.1951  -- Тестовый Человек Справа Родился 16.10.1980  Task  Найден человек, у которого обоих дедушек зовут Иванами | Есть нужный элемент |
| **21** | Человек  Тестовый  Корневой  11  11  2000  0  0  0  --  Человек  Тестовый  Слева  16  11  1980  0  0  0  --  Иван  Тестовый  СлеваЛева  12  5  1950  0  0  0  --  Иван  Тестовый  ЛЛЛ  12  5  1920  14  1  2000  --  \*\*  \*\*  \*\*  Олег  Тестовый  СправаЛева  12  5  1950  0  0  0  --  Иван  Тестовый  ЛСЛ  12  5  1920  14  1  2000  --  \*\*  \*\*  \*\*  Человек  Тестовый  Справа  16  11  1980  0  0  0  --  Иван  Тестовый  СлеваПрава  12  5  1950  0  0  0  --  \*\*  \*\*  \*\* | Генеалогическое дерево  -- Тестовый Иван ЛЛЛ Родился 12.5.1920 Умер 14.1.2000  -- Тестовый Иван СлеваЛева Родился 12.5.1950  -- Тестовый Человек Слева Родился 16.11.1980  -- Тестовый Иван ЛСЛ Родился 12.5.1920 Умер 14.1.2000  -- Тестовый Олег СправаЛева Родился 12.5.1950  -- Тестовый Человек Корневой Родился 11.11.2000  -- Тестовый Иван СлеваПрава Родился 12.5.1950  -- Тестовый Человек Справа Родился 16.11.1980  Task  Найден человек, у которого обоих дедушек зовут Иванами | Два нужных элемента  В логе (см. приложение 4) видно, что дальше первого элемента не шли |
| **22** | Человек  Тестовый  Корневой  11  11  2000  0  0  0  --  Человек  Тестовый  Слева  16  11  1980  0  0  0  --  Олегжа  Тестовый  СлеваЛева  12  5  1950  0  0  0  --  Иван  Тестовый  ЛЛЛ  12  5  1920  14  1  2000  --  \*\*  \*\*  \*\*  Олег  Тестовый  СправаЛева  12  5  1950  0  0  0  --  Иван  Тестовый  ЛСЛ  12  5  1920  14  1  2000  --  \*\*  \*\*  \*\*  Человек  Тестовый  Справа  16  11  1980  0  0  0  --  Иван  Тестовый  СлеваПрава  12  5  1950  0  0  0  --  \*\*  \*\*  \*\* | Генеалогическое дерево  -- Тестовый Иван ЛЛЛ Родился 12.5.1920 Умер 14.1.2000  -- Тестовый Олегжа СлеваЛева Родился 12.5.1950  -- Тестовый Человек Слева Родился 16.11.1980  -- Тестовый Иван ЛСЛ Родился 12.5.1920 Умер 14.1.2000  -- Тестовый Олег СправаЛева Родился 12.5.1950  -- Тестовый Человек Корневой Родился 11.11.2000  -- Тестовый Иван СлеваПрава Родился 12.5.1950  -- Тестовый Человек Справа Родился 16.11.1980  Task  Найден человек, у которого обоих дедушек зовут Иванами | Нужный элемент не первый |

**Код для решения задачи**

/\*

Ремарка. PutInLog(char\* info) определен в доп. заголовочном файле. Просто выводит информацию в лог-файл. Код в приложении

\*/

//Поиск человека, у которого обоих дедушек зовут Иванами

int Task(NodePtr CurrentPosition)

{

if (IsEmpty(CurrentPosition) or strcmp(getData(CurrentPosition).FName,"\*\*")==0)

{

printf("Невозможно выполнить поиск на путом дереве\n");

PutInLog((char\*)"Функция вызвана на путом дереве");

} else {

PutInLog((char \*) "Проверка элемента ");

ShowData(getData(CurrentPosition), pLog);

if (

(strcmp(getData(curL(CurrentPosition)).FName, "\*\*") != 0 and

strcmp(getData(curR(CurrentPosition)).FName, "\*\*") != 0)

and (!IsEmpty(curL(curL(CurrentPosition))) and !IsEmpty(curR(curL(CurrentPosition))))

) {

PutInLog((char \*) "У элемента найдены дедушки\n");

if (strcmp(getData(curL(curL(CurrentPosition))).FName, "Иван") == 0

and strcmp(getData(curR(curL(CurrentPosition))).FName, "Иван") == 0) {

PutInLog((char \*) "Элемент соответсвует поиску");

return 1;

}

PutInLog((char \*) "Не соответствует поиску\n");

return ((!IsEmpty(curL(CurrentPosition))) and (Task(curL(CurrentPosition)))

or (!IsEmpty(curR(CurrentPosition))) and

(Task(curR(CurrentPosition))));

}

PutInLog((char \*) "У элемента нет дедушек\n");

}

return 0;

}

//Инициализатор создания отсортированного дерева

void MakeTreeKey()

{

PutInLog((char\*)"Построение отсортированного дерева\n");

TreeSortedBranches(getStartUnsorted()); //Создание ветвей

PutInLog((char\*)"Создано дерево:\n");

ShowTree1(getStartSorted(), pLog);

PutInLog((char\*)"План поминок выглядит:\n");

ShowTree(getStartSorted(), pLog);

}

// Рекурсия для построения отсортированного дерева

void TreeSortedBranches(NodePtr CurrentPositionOfUnsorted)

{

data currentData = getData(CurrentPositionOfUnsorted);

if (strcmp(currentData.FName,"\*\*")!=0 and !IsEmpty(CurrentPositionOfUnsorted)) { //Проверка на пустоту

if (currentData.dDeath.day) {

AddToTreeKey(currentData);

}

if (!IsEmpty(curL(CurrentPositionOfUnsorted)) and

strcmp(getData(curL(CurrentPositionOfUnsorted)).FName, "\*\*") != 0)

//Если слева не пусто - вызываем для левой части

TreeSortedBranches(curL(CurrentPositionOfUnsorted));

if (!IsEmpty(curR(CurrentPositionOfUnsorted)) and

strcmp(getData(curR(CurrentPositionOfUnsorted)).FName, "\*\*") != 0)

//Если справа не пусто - вызываем для правой части

TreeSortedBranches(curR(CurrentPositionOfUnsorted));

}

else {

//Если пустое - то записываем стоп-знак

NodePtr headSorted = getStartSorted();

replDataM(&headSorted, {(char\*)"\*\*"});

}

}

**Заключение:**

**Приложение 1:**

**Приложение 2:**

**Приложение 3:**

**Приложение 4:**

Инициализированы модули

Построение обыкновенного дерева

Найден новый элемент

Прочитан новый элемент

Создана ветвь Тестовый Человек Корневой Родился 11.11.2000

Переход влево

Найден новый элемент

Прочитан новый элемент

Создана ветвь Тестовый Человек Слева Родился 16.11.1980

Переход влево

Найден новый элемент

Прочитан новый элемент

Создана ветвь Тестовый Иван СлеваСлева Родился 12.5.1950

Переход влево

Найден новый элемент

Прочитан новый элемент

Создана ветвь Тестовый Иван ЛЛЛ Родился 12.5.1920 Умер 14.1.2000

Переход влево

Прочитан cтоп-знак

Возврат на предыдущую позицию

Переход вправо

Прочитан cтоп-знак

Возврат на предыдущую позицию

Возврат на предыдущую позицию

Переход вправо

Прочитан cтоп-знак

Возврат на предыдущую позицию

Возврат на предыдущую позицию

Переход вправо

Найден новый элемент

Прочитан новый элемент

Создана ветвь Тестовый Олег СправаСлева Родился 12.5.1950

Переход влево

Найден новый элемент

Прочитан новый элемент

Создана ветвь Тестовый Иван ЛПЛ Родился 12.5.1920 Умер 14.1.2000

Переход влево

Прочитан cтоп-знак

Возврат на предыдущую позицию

Переход вправо

Прочитан cтоп-знак

Возврат на предыдущую позицию

Возврат на предыдущую позицию

Переход вправо

Прочитан cтоп-знак

Возврат на предыдущую позицию

Возврат на предыдущую позицию

Возврат на предыдущую позицию

Переход вправо

Найден новый элемент

Прочитан новый элемент

Создана ветвь Тестовый Человек Справа Родился 16.11.1980

Переход влево

Найден новый элемент

Прочитан новый элемент

Создана ветвь Тестовый Иван СлеваСправа Родился 12.5.1950

Переход влево

Прочитан cтоп-знак

Возврат на предыдущую позицию

Переход вправо

Прочитан cтоп-знак

Возврат на предыдущую позицию

Возврат на предыдущую позицию

Переход вправо

Конец файла

Создано дерево:

-- Тестовый Человек Корневой Родился 11.11.2000

-- Тестовый Человек Слева Родился 16.11.1980

-- Тестовый Иван СлеваСлева Родился 12.5.1950

-- Тестовый Иван ЛЛЛ Родился 12.5.1920 Умер 14.1.2000

-- Тестовый Олег СправаLedt Родился 12.5.1950

-- Тестовый Иван ЛПЛ Родился 12.5.1920 Умер 14.1.2000

-- Тестовый Человек Справа Родился 16.11.1980

-- Тестовый Иван СлеваСправа Родился 12.5.1950

Построение отсортированного дерева

Элемент: Тестовый Иван ЛЛЛ Родился 12.5.1920 Умер 14.1.2000

Элемент записан

Элемент: Тестовый Иван ЛПЛ Родился 12.5.1920 Умер 14.1.2000

Сравнение текущего элемента с Тестовый Иван ЛЛЛ Родился 12.5.1920 Умер 14.1.2000

Переход вправо

Элемент записан

Создано дерево:

-- Тестовый Иван ЛЛЛ Родился 12.5.1920 Умер 14.1.2000

-- Тестовый Иван ЛПЛ Родился 12.5.1920 Умер 14.1.2000

План поминок выглядит:

Тестовый Иван ЛЛЛ | Умер 14.1.2000

Тестовый Иван ЛПЛ | Умер 14.1.2000

Вызвано добавление элемента вручную

Элемент: Тестовый Человек Доп Родился 11.11.2020 Умер 11.11.2222

Сравнение текущего элемента с Тестовый Иван ЛЛЛ Родился 12.5.1920 Умер 14.1.2000

Переход вправо

Сравнение текущего элемента с Тестовый Иван ЛПЛ Родился 12.5.1920 Умер 14.1.2000

Переход вправо

Элемент записан

Отсортированное дерево:

-- Тестовый Иван ЛЛЛ Родился 12.5.1920 Умер 14.1.2000

-- Тестовый Иван ЛПЛ Родился 12.5.1920 Умер 14.1.2000

-- Тестовый Человек Доп Родился 11.11.2020 Умер 11.11.2222

Вывод деревьев на экран типом Лево-Корень-Право

Вывод плана поминок

Производится поиск элемента по ключу

Проверка элемента Тестовый Человек Корневой Родился 11.11.2000

У элемента найдены дедушки:

Тестовый Иван СлеваСлева Родился 12.5.1950

Тестовый Иван СлеваСправа Родился 12.5.1950

Элемент соответствует поиску

Вызвано пересоздание деревьев

Уничтожено обыкновенное дерево

Уничтожено отсортированное дерево

Создание из D:\Documents\GitHub\Labs\_Progs\Sem2\TC\Tests\in1.txt

Построение обыкновенного дерева

Найден новый элемент

Прочитан новый элемент

Создана ветвь Molekula Chelovek Odin Родился 11.11.2000 Умер 1.2.2060 Родился в Doma

Переход влево

Найден новый элемент

Прочитан новый элемент

Создана ветвь Molekula Chelovek DwaL Родился 16.11.2010 Родился в Uliza

Переход влево

Прочитан cтоп-знак

Возврат на предыдущую позицию

Переход вправо

Прочитан cтоп-знак

Возврат на предыдущую позицию

Возврат на предыдущую позицию

Переход вправо

Найден новый элемент

Прочитан новый элемент

Создана ветвь Molekula Chelovek DwaR Родился 16.10.1920 Умер 1.2.2000

Переход влево

Найден новый элемент

Прочитан новый элемент

Создана ветвь Molekula Chelovek TriL Родился 16.10.1910 Умер 2.1.1990

Переход влево

Прочитан cтоп-знак

Возврат на предыдущую позицию

Переход вправо

Прочитан cтоп-знак

Возврат на предыдущую позицию

Возврат на предыдущую позицию

Переход вправо

Найден новый элемент

Прочитан новый элемент

Создана ветвь Molekula Chelovek TriR Родился 16.10.1911 Умер 2.1.1991

Переход влево

Прочитан cтоп-знак

Возврат на предыдущую позицию

Переход вправо

Конец файла

Создано дерево:

-- Molekula Chelovek Odin Родился 11.11.2000 Умер 1.2.2060 Родился в Doma

-- Molekula Chelovek DwaL Родился 16.11.2010 Родился в Uliza

-- Molekula Chelovek DwaR Родился 16.10.1920 Умер 1.2.2000

-- Molekula Chelovek TriL Родился 16.10.1910 Умер 2.1.1990

-- Molekula Chelovek TriR Родился 16.10.1911 Умер 2.1.1991

Построение отсортированного дерева

Элемент: Molekula Chelovek Odin Родился 11.11.2000 Умер 1.2.2060 Родился в Doma

Элемент записан

Элемент: Molekula Chelovek DwaR Родился 16.10.1920 Умер 1.2.2000

Сравнение текущего элемента с Molekula Chelovek Odin Родился 11.11.2000 Умер 1.2.2060 Родился в Doma

Переход вправо

Элемент записан

Элемент: Molekula Chelovek TriL Родился 16.10.1910 Умер 2.1.1990

Сравнение текущего элемента с Molekula Chelovek Odin Родился 11.11.2000 Умер 1.2.2060 Родился в Doma

Переход влево

Элемент записан

Элемент: Molekula Chelovek TriR Родился 16.10.1911 Умер 2.1.1991

Сравнение текущего элемента с Molekula Chelovek Odin Родился 11.11.2000 Умер 1.2.2060 Родился в Doma

Переход влево

Сравнение текущего элемента с Molekula Chelovek TriL Родился 16.10.1910 Умер 2.1.1990

Переход вправо

Элемент записан

Создано дерево:

-- Molekula Chelovek Odin Родился 11.11.2000 Умер 1.2.2060 Родился в Doma

-- Molekula Chelovek TriL Родился 16.10.1910 Умер 2.1.1990

-- Molekula Chelovek TriR Родился 16.10.1911 Умер 2.1.1991

-- Molekula Chelovek DwaR Родился 16.10.1920 Умер 1.2.2000

План поминок выглядит:

Molekula Chelovek TriL | Умер 2.1.1990

Molekula Chelovek TriR | Умер 2.1.1991

Molekula Chelovek Odin | Умер 1.2.2060

Molekula Chelovek DwaR | Умер 1.2.2000

Вывод деревьев на экран типом Лево-Право-Корень

Вызвано завершение программы

Уничтожено обыкновенное дерево

Уничтожено отсортированное дерево

Завершение программы