Требования к отчету:

Часть 1

Задание на Часть 1. (Смоделировать… /Создать новый тип…)

Определение моделируемой структуры данных.

Способ 1

Описание структуры нового типа, схема, программный код.

Описание порядка выполнения всех операций и их программный код.

Способ 2

Описание структуры нового типа, схема, программный код.

Описание порядка выполнения всех операций и их программный код.

В Части 1 описание способов 1 и 2 можно сделать последовательно (один за другим: сначала все операции по способу 1, затем все операции по способу 2), либо для каждой из операций описать сразу оба способа ее реализации перед описанием следующей операции.

Часть 2

Задание на Часть 2. (На основе описанного нового типа решить…)

Условие задачи (по варианту).

Формат входного текстового файла.

Описание интерфейса (меню) созданного приложения.

Функциональные тесты.

Программный код процедуры для решения задачи (фильтр/поиск)

Заключение

Что было сделано в данной работе.

Сравнение двух способов: недостатки и достоинства (по отношению друг к другу) использованных в работе структур: Списков, Файлов и Массивов, выявленные при моделировании на основе них нового типа.

Вывод: Какой способ оказался проще, удобней? Почему?

Приложения

Приложение 1. Программный код первого модуля (способ 1)

Приложение 2. Программный код второго модуля (способ 2)

Приложение 3. Программный код приложения, работающего с любым из двух написанных модулей (имена типов и операций должны совпадать в модулях).

Приложение 4. Пример получившегося лог-файла (С выполнением всех возможных действий).

**Задание В – «БИНАРНОЕ ДЕРЕВО»**

**Часть В1. Смоделировать двумя способами** новый тип «Бинарное дерево»:

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

Корень дерева

Г

Б

Е

NIL

Д

В

NIL

NIL

NIL

NIL

NIL

NIL

A

При обходе «в глубину» (Корень-Левое-Правое поддерево) в файле нужно записать:

Г

Б

А

\*\*

\*\*

В

\*\*

\*\*

Д

\*\*

Е

\*\*

\*\*

При выводе Левое\_поддерево – Корень – Правое\_поддерево получим:

\*\*А\*\*Б\*\*В\*\*Г\*\*Д\*\*Е\*\* (\*\* - пустая вершина)

(без вывода пустых вершин) получим: АБВГДЕ

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

Б

Г

Д

А

В

\*

Е

\*

\*

\*

\*

\*

Номера дочерних вершин рассчитываются по формулам  
2N и 2N+1 для родительских вершин с номерами N =1,2,3,4,…

При расчете индексов/позиций в файле с 0 нужна поправка +1 к индексу родительской вершины

\*

\*

\*

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;

**Часть В2. 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.

Корень

При выводе Левое\_поддерево – Корень – Правое\_поддерево получим возрастающую(неубывающую) последовательность: 2,3,4,6,6,9

Корень дерева

6

3

9

NIL

6

4

NIL

NIL

NIL

NIL

NIL

NIL

2

Корень

6

3

4

NIL

NIL

NIL

NIL

2

Корень дерева

6

NIL

NIL

Первое число 6

Число 3 меньше 6

6

3

NIL

NIL

NIL

Число 4 меньше 6, но больше 3

Число 2 меньше 6, и меньше 3

NIL

Число 6 не меньше 6

Число 9 больше 6 и второй 6

**Список задач по заданию В – «Бинарное дерево».**

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.** Например, дерево Ивана (указаны для упрощения только имена):

Корень дерева  
(адрес)

Иван

Пётр

Ольга

NIL

Мария

Елена

NIL

NIL

NIL

NIL

NIL

NIL

Алексей

Генеалогическое дерево Ивана.

Отец: Пётр, родителями которого являются Алексей и Елена (дедушка и бабушка Ивана);

Мать: Мария, у которой отец не указан в дереве, а мать Ольга (бабушка Ивана).

10. а) Создав соответствующее упорядоченное дерево, вывести информацию обо всех людях от самого старшего до самого младшего по дате рождения. б) В генеалогическом дереве найти всех предков, кто родился в Москве. прожил не менее 70 лет.