Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

|  |
| --- |
|  |
|  |

ОТЧЕТ

по лабораторной работе

на тему:

Списки

|  |  |  |
| --- | --- | --- |
| Выполнил  Студент гр. 151001 |  | Н.В. Малявко |
| Проверил |  | Асс. Е.Е. Фадеева |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Минск, 2022

1. Теоретические сведения по теме лабораторной работы

*Указатель* — статическая переменная, в которой хранится численное значение (адрес); переменная, диапазон значений которой состоит из адресов ячеек памяти и специального значения — нулевого адреса.

Односвязный список - это динамическая структура данных, состоящая из узлов. Каждый узел будет иметь какое-то значение и указатель на следующий узел.

1. Задание на лабораторную работу

Постановка задачи  
Для заданного преподавателем варианта реализовать программу, хранящую данные в типизированном файле. Чтение данных из файла происходит в динамический список. Дальнейшие манипуляции данными выполняются в динамическом списке. Программа должна содержать пользовательское меню, состоящее из следующих пунктов:

1. Чтение данных из файла
2. Просмотр всего списка (если несколько списков – выпадает подменю с именами списков для просмотра)
3. Сортировка данных в соответствии с заданием
4. Поиск данных с использованием фильтров
5. Добавление данных в список (если несколько списков – выпадает подменю с именами списков для добавления)
6. Удаление данных из списка (если несколько списков – выпадает подменю с именами списков для удаления)
7. Редактирование данных (если несколько списков – выпадает подменю с именами списков для редактирования)
8. -
9. Выход из программы без сохранения изменений
10. Выход с сохранением изменений

В магазине имеется список поступивших в продажу CD/DVD дисков. Каждая запись списка содержит: тип хранимой информации (фильм, музыка, СОФТ (антивирусы, обучающие программы, путеводители и т.д.)), наименование, ФИО автора, цену и примечание (поле м.б. пустым). Требуется:

* Отсортировать внутри каждого типа информацию по наименованию либо по автору;
* Осуществлять поиск диска по автору, по наименованию;
* Предусмотреть возможность добавлять, удалять и корректировать записи из списка, а так же просматривать весь список.

1. Выполнение

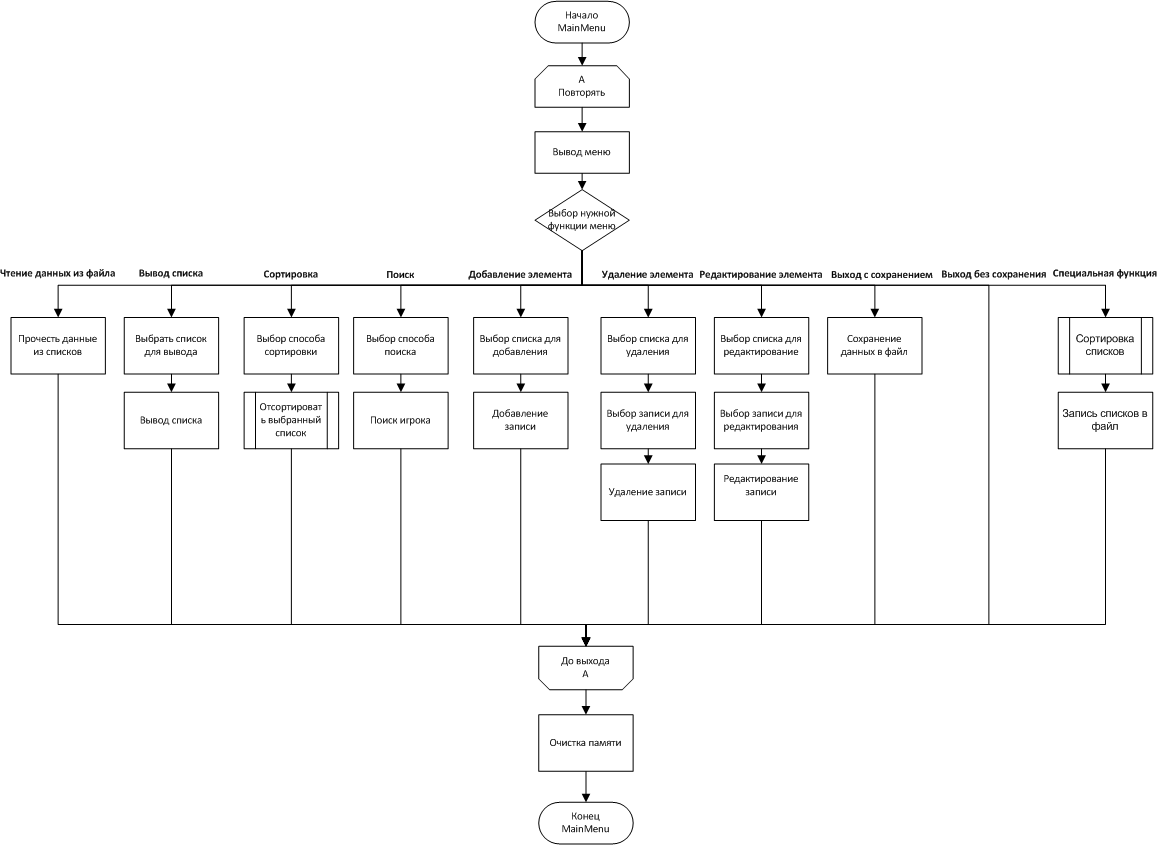
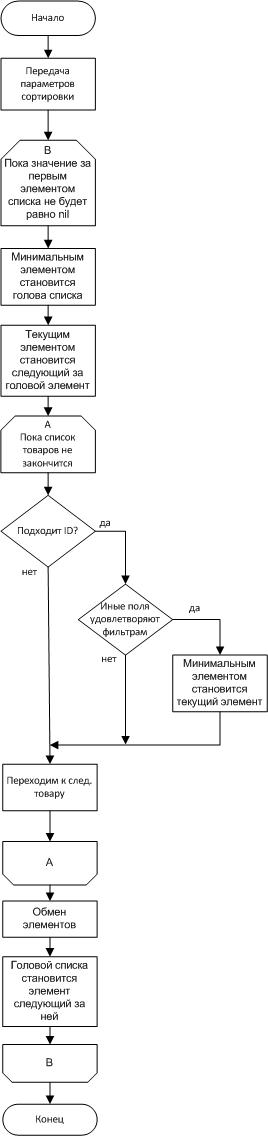


Рисунок 3.1 – Схема работы программы



Разработка алгоритма

***Типы данных для хранения информации о типе информации:***

pInfTypeElement = ^TInfTypeElement;

TInfType = record

ID\_INFO: integer;

NAME: string[20];

PARAM\_NAME\_1: string[20];

PARAM\_NAME\_2: string[20];

end;

TInfTypeElement = record

Data: TInfType;

Next: pInfTypeElement;

end;

***Типы данных для хранения информации об игроках:***

pProductElement = ^TProductElement;

TProduct = record

ID\_PRODUCT: integer;

ID\_INF\_TYPE: integer;

NAME: string[20];

AUTHOR: string[20];

PRICE: integer;

PARAM\_VALUE\_1: string[20];

PARAM\_VALUE\_2: string[20];

NOTE: string[20];

end;

TProductElement = record

Data: TProduct;

Next: pProductElement;

end;

***Описание функций и процедур***

* + function InputString(): string;

Функция проверки ввода строки

* + function InputNumber(): integer;

Функция проверки ввода числа

* + procedure PrintIsFound(var Found: boolean); inline;

Вспомогательная процедура для вывода результатов поиска.

* + procedure PrintProduct(const Temp: pProductElement); inline;

Процедура для вывода элемента списка товаров

* + procedure PrintInfType(const Temp: pInfTypeElement); inline;

Процедура для вывода элемента списка типов информации

* + procedure EditProduct(var El: pProductElement; const ID: integer);

Процедура редактирования товара

* + procedure EditInfType(var El: pInfTypeElement);

Процедура редактирования типа информации

* + function isProductInList(const Head: pProductElement; var SearchWord: string; var ID: integer): boolean;

Процедура проверки записи товаров на вхождение в список

* + function isInfTypeInList(const Head: pInfTypeElement; var SearchWord: string; var ID: integer): boolean;

Процедура проверки записи типа информации на вхождение в список

* + procedure InitLists(var HeadProduct: pProductElement; var HeadInfType: pInfTypeElement);

Процедура инициализации списков

* + procedure ReadProduct(var Head: pProductElement; var ProductFile: TProductFile; var MaxProductID: integer);

Считывает список товаров из типизированного файла

* + procedure ReadInfType(var Head: pInfTypeElement; var InfTypeFile: TInfTypeFile; var MaxInfTypeID: integer);

Считывает список типов информации из типизированного файла

* + procedure WriteProduct(const Head: pProductElement; var ProductFile: TProductFile);

Записывает список продуктов в типизированный файл

* + procedure WriteInfType(const Head: pInfTypeElement; var InfTypeFile: TInfTypeFile);

Записывает список типов информации в типизированный файл

* + procedure WriteProductTxt(const Head: pProductElement; var ProductTxt: TextFile);

Записывает список продуктов в текстовый файл

* + procedure WriteInfTypeTxt(const Head: pInfTypeElement; var InfTypeTxt: TextFile);

Записывает список типов информации в текстовый файл

* + procedure PrintProductList(const Head: pProductElement);

Процедура для вывода списка товаров

* + procedure PrintInfTypeList(const Head: pInfTypeElement);

Процедура для вывода списка типов информации

* + procedure swapProduct(a, b: pProductElement);

Процедура обмена элементов списка товаров

* + procedure swapInfType(a, b: pInfTypeElement);

Процедура обмена элементов списка типов информации

* + function cmpInfTypeName(a, b: pInfTypeElement): boolean;

Функция сравнения

* + function cmpProductName(a, b: pProductElement): boolean;

Функция сравнения

* + function cmpProductAuthor(a, b: pProductElement): boolean;

Функция сравнения

* + function cmpProductIDName(a, b: pProductElement): boolean;

Функция сравнения

* + function cmpProductIDAuthor(a, b: pProductElement): boolean;

Функция сравнения

* + procedure sortProduct(Head: pProductElement; const Comp: TcompProduct);

Сортирует список продуктов

* + procedure sortInfType(Head: pInfTypeElement; const Comp: TcompInfType);

Сортирует список типов информации

* + function equalProductName(a: pProductElement; b: string): boolean;

Функция сравнения

* + function equalProductAuthor(a: pProductElement; b: string): boolean;

Функция сравнения

* + procedure SearchProduct(const Head: pProductElement; Comp: TequalProduct; const SearchWord: string; var Found: boolean);

Поиск товаров в списке с фильтрами

* + procedure SearchProductByID(const Head: pProductElement; const ID: integer; var Found: boolean);

Поиск товаров в списке по ID

* + procedure SearchProductByInfID(const Head: pProductElement; const ID: integer; var Found: boolean);

Поиск товаров в списке по типу информации

* + procedure SearchInfTypeByName(const Head: pInfTypeElement; const SearchWord: string; var Found: boolean);

Поиск типа информации в списке по имени

* + procedure SearchInfTypeByID(const Head: pInfTypeElement; const ID: integer; var Found: boolean);

Поиск типа информации в списке по ID

* + function GetProduct(const Head: pProductElement; const ID: integer) : pProductElement;

Возвращает искомый элемент

* + function GetInfType(const Head: pInfTypeElement; const ID: integer) : pInfTypeElement;

Возвращает искомый элемент

* + procedure AddProduct(const HeadProduct: pProductElement; const ID: integer; var MaxProductID: integer);

Процедура добавления товара в список

* + procedure AddInfType(const Head: pInfTypeElement; var MaxInfType: integer);

Процедура добавления типа информации в список

* + procedure DeleteProduct(const Head: pProductElement; const ID: Cardinal);в

Удаляет товар из списка

* + procedure DeleteInfType(const Head: pInfTypeElement; const ID: Cardinal);

Удаляет тип информации из списка

* + procedure KillProduct(var Head: pProductElement);

Очищает память от списка товаров

* + procedure KillInfType(var Head: pInfTypeElement);

Очищает память от списка типов информации

3.2 Тестирование и отладка программы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Специфика тестирования | Номер теста | Вводимые данные | Ожидаемый результат | Полученный результат |
| Некорректный ввод в меню | 1 |  |  | Тест пройден |
| Чтение данных из файла | 2 |  |  | Тест пройден |
| Вывод списка  товаров | 3 |  |  | Тест пройден |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Вывод списка типов информации | 4 |  |  | Тест пройден |
| Сортировка товаров по имени | 5 |  |  | Тест пройден |
| Сортировка товаров по автору | 6 |  |  | Тест пройден |
| Сортировка товаров по имени и типу | 7 |  |  | Тест пройден |
| Сортировка товаров по автору и типу | 8 |  |  | Тест пройден |
| Сортировка типа информации по имени | 9 |  |  | Тест пройден |
| Поиск по списку существующего элемента | 10 |  |  | Тест пройден |
| Поиск по списку несуществующего элемента | 11 |  |  | Тест пройден |
| Добавление в список дисков при существующем типе информации | 12 |  |  | Тест пройден |
| Добавление в список дисков при несуществующем типе информации | 13 |  |  | Тест пройден |
| Удаление диска с существующим ID | 14 |  |  | Тест пройден |
| Удаление диска с несуществующим именем | 15 |  |  | Тест пройден |
| Удаление диска с несуществующим ID | 16 |  |  | Тест пройден |
| Удаление дисков с одинаковым именем | 17 |  | Удаление игрока по его ID | Тест пройден |
| Удаление типа информации при наличии дисков на складе | 18 |  | Тип данных удаляется, если не осталось дисков с этим типом информации | Тест пройден |
| Редактирование диска c одинаковыми именами | 19 |  | Изменяется только игрок с нужным ID | Тест пройден |
| Работа специальной функции | 20 |  | Вывод в текстовый файл | Тест пройден |
| Спец функция | 21 |  | Вывод соответствующего сообщений | Тест пройден |
| Выход без сохранения | 22 |  | Окончание программы | Тест пройден |
| Выход c сохранением | 23 |  | Окончание программы с записью данных | Тест пройден |

Таблица 3.2 Прохождение тестов программы