Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Основы алгоритмизации и программирования

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

на тему

ПРОГРАММНОЕ СРЕДСТВО МОНИТОРИНГА И ОТОБРАЖЕНИЯ СЕТЕВОЙ СТРУКТУРЫ ОТГРУЗКИ ТОВАРОВ СО СКЛАДОВ В МАГАЗИНЫ

БГУИР КР 6-05-0612-01 122 ПЗ

Студент: гр. 451001 Соболь Н.Г.

Руководитель:

асс. Фадеева Е.Е.

Минск 2025

1. **Анализ прототипов, литературных источников и формирование требований к проектируемому программному средству**
   1. **Обзор литературы**

**1.1.1** [1] algorithmica [Электронный ресурс]. – Режим доступа: <https://ru.algorithmica.org/>

1.1.2 [2] e-maxx [Электронный ресурс]. – Режим доступа: <http://e-maxx.ru/algo>

Здесь подробно описываются различные алгоритмы, в том числе алгоритмы хеширования и построения декартового дерева.

**1.1.3** [3] docwiki.embarcadero [Электронный ресурс]. – Режим доступа: <https://docwiki.embarcadero.com/>

Подробное описание работы с delphi и RAD Studio. Документация по процедурам и функциям в языке delphi. Подробное описание создания оконных приложений при помощи RAD Studio.

**1.1.4** [4] Гаджинский, А. М. Логистика. Учебник – Режим доступа: <https://www.booksite.ru/fulltext/logist/text.pdf>

В учебнике собрата теория по логистике, которую можно применить в программном средстве.

* 1. **Примеры решения аналогичных задач**

Рассмотрим уже существующие программные средства, решающие задачу мониторинга и отображения сетевой структуры отгрузки товаров со складов в магазины:

1. Zabbix

Достоинства:

А) Бесплатное программное обеспечение;

Б) Мощные возможности мониторинга и уведомлений;

В) Есть API;

Г) Гибкая настройка.

Недостатки:

А) Высокий порог входа;

Б) Требуется тяжёлая настройка.

2) Nagios

Достоинства:

А) Высокая гибкость и настраиваемость;

Б) Множество плагинов.

Недостатки:

А) Сложная настройка и управление.

3) SAP ERP

Достоинства:

А) Высокая функциональность;

Б) Интеграция с другими модулями ERP.

Недостатки:

А) Высокая стоимость;

Б) Сложность внедрения.

4) Oracle NetSuite

Достоинства:

А) Облачная платформа;

Б) Простота использования.

Недостатки:

А) Ограниченные возможности кастомизации.

* 1. **Требования к проектируемому программному средству**

**1.3.1** Назначение разработки:

Проектируемое программное средство предназначено для мониторинга и отображения сетевой структуры отгрузки товаров со складов в магазины в реальном времени. Оно должно обеспечивать отслеживание отгрузок, а также генерировать таблицы для анализа логистической цепочки. Система должна обеспечивать высокую степень автоматизации, снижать вероятность ошибок и ускорять принятие управленческих решений.

**1.3.2** Состав выполняемых функций**:**

1. Мониторинг отгрузок в режиме реального времени;
2. Визуализация сетевой структуры отгрузок;
3. Формирование таблиц по остаткам товаров;
4. Сохранение данных в файл, возможность чтения из файлов;
5. Формирование таблиц по отгрузке товаров;
6. Анализ информации об уровне запасов в магазинах и на складах.

**1.3.3** Входные данные:

1. Информация о складах: название склада, адрес (улица, дом), вместимость.
2. Информация о магазинах: название склада, адрес (улица, дом), вместимость.
3. Данные о товарах: название, категория, объём единицы товара (количество условных единиц)
4. Данные об отгрузках: название, тип отправителя, название отправителя, идентификатор отправителя, тип получателя, название получателя, идентификатор получателя, название товара, артикул товара, количество товара для отгрузки;

**1.3.4** Выходные данные:

1. Графическое отображение сетевой структуры отгрузок в виде карты;
2. Формирование таблиц по остаткам товаров во всех магазинах и на складах;
3. Формирование таблиц с существующими отгрузками;
4. Формирование таблиц по остаткам товаров в конкретном магазине или в конкретном складе.

**1.3.5** Требования к временным характеристикам:

1. Обновление данных в режиме реального времени;
2. Задержка между обновлением данных и их отображением на экране не должна превышать 1 секунды.

**1.3.6** Требования к надёжности:

Программное средство должно быть устойчиво к программным ошибкам и сбоям, в том числе проверять все входные данные на корректность.

* + 1. Условия эксплуатации:

Программное средство должно поддерживать работу на ОС Windows.

* + 1. Язык и среда разработки:

Язык разработки: Delphi, благодаря его возможности создания высокопроизводительных и надежных приложений с графическим интерфейсом.

Среда разработки: Embarcadero RAD Studio, так как она предлагает мощные инструменты для визуального проектирования и отладки приложений на Delphi.

* + 1. Дополнительные требования:

Интерфейс должен быть простым и понятным для пользователей с разным уровнем технической подготовки. Это включает в себя использование понятных терминов, логичную структуру меню и подсказки для сложных операций. Для отображения сетевой структуры отгрузок необходимо использовать графические элементы (карта, линии на карте, множество панелей), которые позволяют быстро анализировать информацию.

**2** **Анализ требований к ПС и** **разработка функциональных требований**

**2.1 Теоретический анализ, математическое обоснование и доказательства, модели технических объектов и результаты моделирования**

* + 1. Декартово дерево (treap, дерамида)

Декартово дерево – структура данных, которое объединило в себе бинарное дерево поиска (tree) и бинарную кучу (heap). Эта структура данных хранит в себе пары (X, Y), так, что дерамида является бинарным деревом поиска по X и бинарной пирамидой по Y. Таким образом, если в текущей вершине лежит значение (X0, Y0), то в левом поддереве будут лежать элементы со значением X < X0, а в правом со значением X > X0. А также и в левом, и в правом поддереве будут располагаться Y < Y0. В дальнейшей реализации X – ключ, Y – приоритет. Приоритет будет генерироваться случайно, обеспечивая сложность работы O(logN) в среднем. В итоге декартово дерево позволяет выполнять следующие операции:

1. Insert(X, Y) – Вставка элемента в дерево. O(logN) в среднем;

2. Find(X) – Поиск элемента в дереве. O(logN) в среднем;

3. Erase(X) – Удаление элемента из дерева. O(logN) в среднем;

4. Merge(Root1, Root2) – Объединение деревьев. O(Mlog(N/M)) в среднем;

5. Split(Root, X) – Разделение одного дерева на 2 по ключу X. O(logN) в среднем.

* 1. **Описание функциональности ПС**

**2.2.1** Управление логистическими объектами (складами и магазинами)

Пользователь может добавлять новые объекты (склад или магазин), при этом для них он выбирает уникальное название, положение на карте, адрес и вместимость. Во время добавления система сверяет, что имя еще не используется, и что вводимые данные подчиняются правилам (длина строки, разрешённые символы). Реализована также возможность изменения уже существующих объектов: изменение названия, вместимости, адреса. Существует возможность удаления. В процессе удаления система проверяет, существуют ли у объекта отгрузки, связанные с ним, и запрещает его удаление, если такие зависимости имеются. Все объекты записываются в декартово дерево, в котором обеспечивается быстрый поиск, удаление, изменение объекта.

Организован фильтр, при помощи которого можно выбирать, какие объекты нужно отобразить, а какие нужно скрыть.

**2.2.2** Управление ассортиментом товаров

В системе также предусмотрена функция добавления товаров на конкретный склад или магазин. При добавлении нового товара пользователь вводит его название, количество, объем и категорию. Система проверяет, что эти поля заполнены корректно (например, количество и объём числовые, идентификатор уникален). После подтверждения товар записывается в список товаров соответствующего объекта.

Для просмотра всех товаров в системе предусмотрен табличный вывод, в котором отображаются: объект, в котором лежит товар, тип объекта, вместимость этого объекта, количество занятого и зарезервированного места, название товара, категория товара, артикул, количество, место, занятое товаром.

**2.2.3** Планирование и исполнение отгрузок

Пользователь может создать новую отгрузку, выбрав отправителя (склад или магазин), получателя (склад или магазин), товар и количество условных единиц. Перед созданием, система проверяет, что у отправителя есть достаточное количество желаемого товара на балансе. После подтверждения отгрузка записывается в список отгрузок.

Также для каждой действующей отгрузки система визуально показывает маршрут на карте, связывая отправную и конечную точки.

**2.2.3** Фильтрация и поиск данных

Система предоставляет средства фильтрации отображения объектов на карте по следующим критериям: тип объекта, адрес объекта, вместимость объекта, занятое место в объекте.

После указания нужных параметров пользователь нажимает «Применить», и на карте отображаются только те объекты, которые удовлетворяют условиям.

**2.2.4** Визуализация логистической сети

На основной карте отображаются все склады и магазины условными символами (разный цвет для складов и магазинов). При наведении мыши на объект на карте можно увидеть его краткое описание.

Для каждой активной отгрузки на карте рисуется линия с направлением от точки отправления до точки прибытия. При наведении курсора мыши на отгрузку можно увидеть её подробные сведения: кто отправитель, кто получатель, какой товар, сколько штук было отправлено.

Вся визуализация автоматически обновляется при создании, обработке или удалении отгрузок, а также при добавлении или удалении объектов, чтобы карта всегда отражала актуальное состояние сети.

**2.2.5** Сохранение и восстановление состояния

Система поддерживает сохранение полного состояния базы данных (списки объектов, товары, отгрузки, балансы) в файлы. При следующем запуске приложения можно загрузить ранее сохранённые файлы, и система восстановит все объекты, товары, отгрузки и отобразит их на карте.

При загрузке данных прежние списки очищаются, затем последовательно восстанавливаются объекты с их товарами и балансовыми остатками, а затем восстанавливается список отгрузок с соответствующими маршрутами. После завершения загрузки карта автоматически перерисовывается, чтобы отобразить все объекты и текущие маршруты.

* 1. **Спецификация функциональных требований**
     1. Визуализация сетевой структуры отгрузок

Сетевая структура отгрузок (склады, магазины, маршруты) отображается в виде карты. Карта автоматически подстраивается под разрешение экрана пользователя.

Элементы карты:

-склады, которые обозначаются бордовыми кругами;

-магазины, которые обозначаются синими кругами;

-маршруты, которые отображаются черными линиями.

* + 1. Построение прямого маршрута

Функция построения маршрута на карте принимает на вход список всех линий и текущую отгрузку. Функция добавляет линию, содержащую данную отгрузку к списку всех лини. Визуально на карте результат выглядит как непрерывная прямая линия, соединяющая отправителя и получателя.

В дальнейшем, при перемещении курсора вычисляется расстояние между курсором и линиями. Если это расстояние меньше определенной константы, на экран выводится информация о данной отгрузке.

* + 1. Добавление/редактирование/удаление объектов

Должно быть реализовано несколько функций, каждая из которых будет отвечать за добавление, редактирование или удаление определенного объекта (магазина / склада). На ввод подаётся информация о новом или уже существующем объекте, в зависимости от того, какая операция будет выполняться. Все данные проходят валидацию. Функция ничего не возвращает, а только добавляет объект в список, редактирует уже существующий объект или удаляет объект из списка.

**3 Проектирование программного средства**

 **3.1 Разработка алгоритма ПС и алгоритмов отдельных модулей**

Рисунок 3.1.1 – Схема работы quickSort

Алгоритм быстрой сортировки реализован рекурсивно. Изначально в процедуру передается массив, параметр для сортировки, левая и правая граница. Если левая граница больше или равна правой, то алгоритм завершает работу. В противном случае алгоритм вызывает функцию выбора опорного элемента и помещает все элементы, которые меньше опорного, слева, остальные – справа. Далее процедура вызывается рекурсивно для левой и правой части массива.



Рисунок 3.1.2 – Схема работы partition

На ввод функции передается массив, границы и параметр, по которому производится сортировка. Далее случайным образом выбирается опорный элемент. Запускаем цикл, пока левая граница меньше правой. Внутри цикла происходит перестановка элементов. Все элементы, которые меньше опорного перемещаются в левую часть, остальные – в правую. В конце функция возвращает индекс левой границы.



Рисунок 3.1.3 – Схема работы dijkstra

На ввод функция получает граф, заданный списком смежности, количество вершин, начальную вершину, конечную вершину. Создается массив кратчайших расстояний, изначально заполненный числами, заведомо большими, чем максимально возможное расстояние в графе. Расстояние до стартовой вершины инициализируется нулём. Создаётся очередь с приоритетом (куча), в вершине которой всегда будет храниться вершина, расстояние до которой на данной итерации наименьшее. В очередь добавляется начальная вершина. Запускается цикл, пока в очереди есть хотя бы 1 элемент. На каждой итерации выбирается элемент с вершины очереди и удаляется из нее. Перебираются все рёбра, которые исходят из выбранной вершины. Для каждой вершины, в которую ведет текущее рассматриваемое ребро, совершается попытка найти ответ, который лучше, чем уже существующий. Если такой ответ существует, мы обновляем массив расстояний и добавляем вершину и расстояние до нее в приоритетную очередь. В конце происходит восстановление ответа и его возврат.



Рисунок 3.1.4 – Схема работы filter

Функция принимает на вход данные и параметр, по которому будет производиться фильтрация. Создаётся список ответа. Далее цикл проходит по каждому элементу в массиве входных данных и, если текущий элемент подходит по параметру, он добавляется в список ответа. В конце функция возвращает сформированный список ответа.



Рисунок 3.1.5 – Схема работы addShop

Процедура начинается с цикла, который проверяет корректность входных данных. Пользователь должен ввести название и расположение магазина. После завершения цикла процедура добавляет магазин в список магазинов.



Рисунок 3.1.6 – Схема работы addWarehouse

Процедура начинается с цикла, который проверяет корректность входных данных. Пользователь должен ввести название, расположение и вместимость склада. После завершения цикла процедура добавляет склад в список складов.

Рисунок 3.1.7 – Схема работы addItem

Процедура начинается с цикла, который проверяет корректность входных данных. Пользователь должен ввести название, место добавления, количество, категорию товара. После завершения цикла процедура добавляет товар в соответствующее место.



Рисунок 3.1.8 – Схема работы rmShop

Процедура начинается с цикла, который проверяет корректность входных данных. Пользователь должен ввести название магазина. После завершения цикла процедура удаляет магазин из списка магазинов.



Рисунок 3.1.9 – Схема работы rmWarehouse

Процедура начинается с цикла, который проверяет корректность входных данных. Пользователь должен ввести название склада. После завершения цикла процедура удаляет склад из списка складов.



Рисунок 3.1.10 – Схема работы linearRegression

Функция принимает на вход массивы координат x и y, коэффициент A и коэффициент B. Функция должна вычислить коэффициенты для уравнения y = Ax + B, по которой и будет производиться прогнозирование. После ввода начинается цикл по значениям массивов. Происходит вычисление четырёх значений: сумма всех X, сумма всех Y, сумма произведений X\*Y, сумма квадратов всех X. После цикла происходит вычисление знаменателя по формуле , где n – количество координат, sumX – сумма всех X, sumX2 – сумма квадратов всех X. Далее происходит вычисление коэффициентов A и B.



Рисунок 3.1.11 – Схема работы editShop

Процедура начинается с цикла, который проверяет корректность входных данных. Пользователь должен ввести название магазина. После завершения цикла происходит выбор поля, которое необходимо поменять. Пользователь вводит новое значение поля и подтверждает или отменяет свой выбор.



Рисунок 3.1.12 – Схема работы editWarehouse

Процедура начинается с цикла, который проверяет корректность входных данных. Пользователь должен ввести название склада. После завершения цикла происходит выбор поля, которое необходимо поменять. Пользователь вводит новое значение поля и подтверждает или отменяет свой выбор.



Рисунок 3.1.13 – Схема работы editItem

Процедура начинается с цикла, который проверяет корректность входных данных. Пользователь должен ввести название товара и место, в котором его нужно поменять (список товаров / конкретный склад / конкретный магазин). После завершения цикла происходит выбор поля, которое необходимо поменять. Пользователь вводит новое значение поля и подтверждает или отменяет свой выбор.