Тема курсовой работы

Учет топлива на автозаправочной станции. Имеется несколько видов топлива на АЗС. Учет продаж (суммовой и количественный) ведется в разрезах продавцов и видов топлива. Пополнение запасов происходит путем закупки требуемых материалов у сторонних контрагентов..

Предлагаемые структуры

Исходя из требований к отчетам, можно выделить5 сущностей, необходимых для их построения: **топливо**, **учет продаж**, **продавец, закупки топлива, контрагент** у которого закупают топливо.

В каждой таблице должно быть уникальное поле, по которому можно отделить одну запись от другой. В качестве такого поле используют идентификатор (т.е. по сути, нумерацию строк в таблице, причем номера не повторяются).

1. **Топливо** – сущность, хранящая информацию об виде топлива, его цены и количества.

· id топлива– целочисленный тип данных для хранения уникального поля сущности .

· Вид топлива – строковый тип данных.

· Цена топлива – вещественный тип данных.

· Количество топлива – вещественный тип данных.

2. **Учет продаж** – сущность, хранящая информацию о учете продаж топлива, номер продажи, номер топлива, номер продавца, и количество проданного за одну продажу.

· id заказа– целочисленный тип данных для хранения уникального поля сущности.

· id топлива – целочисленный тип данных для хранения поля сущности Топлива в учете продаж (так как id топлива является целочисленным полем).

· id продавца – строковый тип данных.

· Количество – целочисленный тип данных.

3. **Продавец** – сущность, хранящая информацию о продавце, номер продавца, продающего топливо, ФИО продавца.

· id продавца – идентификатор, целочисленный тип данных.

· ФИО – строковый тип данных.

4. **Закупки топлива** –сущность, хранящая информацию о закупках топлива, номер закупки, номер контрагента, номер топлива, количество закупленного топлива.

· id закупки – идентификатор, целочисленный тип данных.

· id контрагента – целочисленный тип данных.

· id топлива –целочисленный тип данных.

· Количество – целочисленный тип данных.

5. **Контрагент** – сущность, хранящая информацию про контрагента (поставщика), номер контрагента, продающего топливо, ФИО контрагента.

· id – идентификатор, целочисленный тип данных.

· ФИО – строковый тип данных.

Общая схема зависимости сущностей

У нас получается три независимых сущности и две сущности, которые зависят от двух независимых (так как в них фигурируют данные из других сущностей, а именно идентификаторы).

Получается такая схема зависимостей:

J:\Education\work 11.2016\diagram.png

**То, что красным – это дополнительная информация в итоговое оформление она не должна входить.**

Тема курсовой работы

Конкурс красоты. Участницы проходят ряд соревнований, и на каждом из них члены жюри выставляют оценки. Назвать первую тройку победителей в каждом соревновании и итоговую тройку победителей конкурса. Также, определить, кому каждый член жюри отдавал предпочтение в конкурсе.

Топливо

Для сущности «**топливо**» в качестве используемой коллекции будем использовать **односвязный список**. Добавление будет происходить в **хвост списка** (Тип списка и вариант добавления может быть иным, на усмотрение студента).

Добавление

При добавлении новой записи в список пользователь должен ввести следующие данные:

· Вид топлива

· Цена топлива

· Количество топлива

После ввода данных необходимо произвести следующие проверки:

· В поля «Цена» и «Количество» должно быть записано вещественное значение.

Если проверка прошла успешно, генерируем идентификатор для новой записи по следующему принципу:

· если в списке есть записи,

o то находим максимальное значение идентификатора из этих записей, прибавляем к нему 1 и запоминаем полученное значение, как идентификатор добавляемой записи,

o иначе присваиваем идентификатору значение 1.

Другие варианты заполнения идентификатора: ввод пользователем, поиск минимального свободного или иной, по желанию студента

Созданный элемент заносим в список.

Редактирование

Редактировать разрешено все поля, кроме идентификатора, так как его изменение может нарушить логические связи между сущностями.

Перед редактированием пользователь вводит номер идентификатора записи, которую хочет изменить. Если записи с таким номером нет, то сообщить об этом пользователю, иначе вызывать операцию редактирования.

Проверки новых значений, заносимых в поля, аналогичны проверкам значений при добавлении записи в список.

Сохранение новых значений в записи.

Варианты редактирования могут быть разными, от редактирования всей записи разом, до изменения только выбранных полей.

Удаление

Для удаления записи из списка пользователь должен ввести номер идентификатора удаляемой записи. Если записи с таким номером нет, то сообщить об этом пользователю, иначе удалить элемент из списка и из памяти.

Эта сущность связана с сущностями «Учет продаж» и «Закупки топлива», поэтому прежде чем удалить запись из списка необходимо выполнить проверку, используется ли эта запись в списках «Учет продаж» и «Закупки топлива». Если данная запись там используется, то запретить удаление записи. Другой вариант: заменить значение поля «Id участницы» в списке «Оценка» в нужных записях на определенное значение и удалить после этого запись из списка «Участниц».

Соревнования

По аналогии

Члены жюри

По аналогии

Оценки

По аналогии

Формирование отчетов

Победители в соревнованиях

Общий алгоритм: для каждого соревнования найти всех участников и выбрать первых трех, у кого наибольшие баллы и вывести их фамилии с баллами на экран.

Формулировка «для каждого» сразу говорит об использовании цикла, поэтому соревнования будем перебирать через цикл. Далее в рамках этого цикл необходимо просмотреть все записи из списка «Оценки» и выбрать те из них, в которых будет встречаться идентификатор соревнования, по которому в данный момент мы производим выборку. Прежде чем далее продолжать уточнение алгоритма, необходимо определиться, как будет выбирать первых трех. Варианта два: или составляем отдельный список всех участниц этого соревнования и их баллов, сортируем его и выводим первых трех; или заводим 3 переменные, в которых будем хранить id участниц и их баллы и записывать в них участниц с наибольшими баллами. Выберем первый вариант, причем в качестве списка воспользуемся стандартной коллекцией map, в которую будем записывать id участницы как ключ и ее балл, как значение. И так идем по списку «Оценки», как только мы встречаем запись в списке «Оценки», которая удовлетворяет нашему поиску, то в коллекцию map добавляем новую запись (можно еще предварительно выполнить проверку, а не встречалась ли эта участница в этом map уже). После прохода по всему списку «Оценки» сортируем map по убыванию и выводим первые 3 записи на экран.

Алгоритм будет следующим:

1. Цикл по списку соревнований

1.1.Выводим название соревнования

1.2.Создаем map

1.3.Цикл по списку оценок

1.3.1. Если идентификатор текущего элемента соревнования совпадает с идентификатором соревнования в текущей записи оценок

1.3.1.1. Если в map не было еще записи с таким идентификатором участницы

1.3.1.1.1. Добавляем в map новую запись (идентификатор участницы, ее балл за соревнование)

1.4.Вызываем функцию сортировки для map

1.5.Цикл для вывода первых трех записей (т.е. в цикле будет 3 шага)

1.5.1. Цикл по списку участниц

1.5.1.1. Если ключ текущей записи map совпадает с идентификатором текущей записи из списка участниц

1.5.1.1.1. Выводим ФИО участницы и ее номер

1.5.1.1.2. Выводим значение из map (т.е. ее балл)

Победители в конкурсе

Общий алгоритм: для каждой участницы пройти по всем записям списка «Оценки» и просуммировать ее баллы за каждое соревнование.

Здесь сразу заведем map (ключ – идентификатор участницы, значение – ее балл). Здесь два варианта работы: или мы добавляем участницу, проходим по списку «Оценок», находим записи с ней и сразу суммируем баллы; или сначала добавляем всех участниц в map, потом один раз проходим по списку «Оценок» и прибавляем баллы каждой участнице (в map можно легко обратиться к записи, зная ее ключ). Второй вариант предпочтительней, так как там всего 1 раз будем проходить по списку «Оценки». Далее сортируем map и выводим первые три записи.

Алгоритм будет следующим:

1. Создаем map

2. Цикл по списку «Участниц»

2.1.Добавляем в map новую запись (идентификатор участницы, 0 в качестве балла)

3. Цикл по списку «Оценок»

3.1.К значению элемента map (ключ – это идентификатор участницы из текущего элемента списка «Оценки») прибавляем значение балла для текущего элемента списка «Оценки».

4. Сортируем map

5. Цикл для вывода первых трех записей (т.е. в цикле будет 3 шага)

5.1.Цикл по списку участниц

5.1.1. Если ключ текущей записи map совпадает с идентификатором текущей записи из списка участниц

5.1.1.1. Выводим ФИО участницы и ее номер

5.1.1.2. Выводим значение из map (т.е. ее балл)

Предпочтения членов жюри

Общий алгоритм: для каждого члена жюри составить список, сколько баллов он начислил каждой участнице, и выбрать из них ту, у, который этот балл максимальный.

Опять же воспользуемся map. Варианта опять же два: или один раз сформировать map по участницам и при каждом переходе к новому члену жюри этот map обнулять (каждой участнице присвоить значение баллов равное 0); или при каждом переходе к новому члену жюри создавать новый map и заново его заполнять участницами из списка «Участниц». В обоих вариантах нам нужно будет проходить циклы несколько раз. Выберем первый вариант, так как там задаем map всего один раз, т.е. будем экономить память.

Алгоритм будет следующим:

1. Создаем map

2. Цикл по участницам

2.1.Добавляем в map новую запись (идентификатор участницы, 0 в качестве балла)

3. Цикл по членам жюри

3.1.Выводим ФИО члена жюри

3.2.Цикл по оценкам

3.2.1. К значению элемента map (ключ – это идентификатор участницы из текущего элемента списка «Оценки») прибавляем значение балла для текущего элемента списка «Оценки».

3.3.Сортируем map

3.4.Цикл по списку участниц

3.4.1. Если ключ текущей записи map совпадает с идентификатором текущей записи из списка участниц

3.4.1.1. Выводим ФИО участницы и ее номер

3.4.1.2. Выводим значение из map (т.е. ее балл)

3.4.2. Обнуляем значение текущей записи map

Хранение данных в файлах

Данные удобнее хранить в двух вариантах:

1. В одном файле все списки (пользователь вводит путь до файла).

В таком случае, перед каждым списком идет указание количества записей в списке для удобства чтения записей из файла (можно и без этого)

2. В одной папке отдельные файлы для каждой сущности (соответственно пользователь вводит только путь до папки, а файлы в папке имеют заранее определенные имена, указать какие именно)

Не менее важно: порядок чтения данных из файла/файлов. Сперва считываются данные сущностей, которые не зависят от других (для данного варианта это участницы, соревнования и члены жюри), далее сущности, которые зависят только от сущностей первой волны (для данного примера, оценки) и так далее, пока не будут считаны все списки. При чтении из файла необходимо производить все те же проверки, как и при добавлении записей пользователем.

Порядок записи будет напрямую зависеть от порядка чтения.

Расписать порядок записи/чтения данных, в каком виде они буду храниться в файле/файлах.

**То, что пишется красным или дополнительная информация или иные варианты работы, тогда нужно выбрать один из предлагаемых вариантов.**

Общее задание

1. Провести анализ предметной области: выделить основные сущности (не менее 4), поля этих сущностей и взаимосвязи между ними.

2. **Защитить полученную структуру (не переходить к следующим этапам, пока не выполнен этот).**

3. На основе сущностей создать структуры (в коде на языке С++).

4. Разработать алгоритмы добавления, редактирования, изменения в сущностях; формирования требуемых отчетов; записи и чтения данных из файлов

5. **Защитить алгоритмы (не переходить к следующим этапам, пока не выполнен этот).**

6. Создать списки для работы с сущностями.

7. Создать класс, хранящий в себе списки всех описанных сущностей и дополнительные характеристики предметной области. У класса должны быть методы для добавления, редактирования и изменения любой сущности, чтения и записи в файл (разработать структуру хранящихся в файле записей) и методы формирования отчетности.

8. В основной программе **создать массив из 5 элементов класса**. Разработать меню со следующими пунктами:

Ø **Первый уровень**: выбор элемента из массива, с которым планируется работать.

Ø **Второй уровень**: список действий:

· Загрузить из файла

· Сущность 1

· Сущность 2

· Сущность 3

· Сущность 4

· Сформировать отчет 1

· Сформировать отчет 2 и т.д.

· Сохранить в файл

· Перейти к другому элементу (вернутся к первому уровню).

Ø **Третий уровень** (для сущности): добавить, изменить, удалить, вывести на экран.

9. При загрузке данных из файла необходимо вывести все загруженные данные на экран.

10.При добавлении, изменении или удалении элементов из коллекции необходимо проводить проверку на корректность зависимых от этой структуры данных (если вводятся цифры, проверка, что вводятся цифры; если указываются идентификаторы, то проверка, что такие есть). После операции вывести на экран измененный список.

11.Продемонстрировать готовую программу. Проверка будет на соответствие общему заданию, заданию по варианту, описанным алгоритмам.

12.**На 5**: выступить на выставке программных продуктов и конференции со своим программным продуктом (в докладе к конференции описать цели и задачи, методы решения, полученный результат и перспективы развития).

13.Сформировать пояснительную записку к курсовой работе.

Оформление пояснительной записки

Записка должна содержать следующие разделы:

**Раздел 1. Постановка задачи и анализ исходных данных**

Описание разработанной структуры.

**Раздел 2. Разработка методов**

Описание разработанных алгоритмов.

**Раздел 3. Реализация**

2.1 Описание функции добавления в список (выбирается одна любая сущность, у которой есть зависимость от других сущностей)

- блок-схема алгоритма;

- текст фрагмента программы.

2.2 Описание функции редактирования из списка (выбирается одна любая сущность, у которой есть зависимость от других сущностей)

- блок-схема алгоритма;

- текст фрагмента программы.

2.3 Описание функции удаления из списка (выбирается одна любая сущность, от которой зависят другие)

- блок-схема алгоритма;

- текст фрагмента программы.

2.4 Описание функции чтения из файла

- блок-схема алгоритма;

- текст фрагмента программы.

2.5 Описание функции запись в файл

- блок-схема алгоритма;

- текст фрагмента программы.

2.6 Описание функции создания отчета 1

- блок-схема алгоритма;

- текст фрагмента программы.

2.7 Описание функции создания отчета 2

- блок-схема алгоритма;

- текст фрагмента программы.

2.8 И т. д. для всех отчетов.

**Раздел 4. Эксперименты.**

Провести эксперименты. Ввести данные, указать ожидаемый результат и то, что выведется программой (скриншоты). Описать полученные результаты.

**Заключение**

Описать, какие технологии, стандартные контейнеры, методы и прочее были изучены. Продумать пути улучшения.

**Приложение. Листинг программного кода**

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет «Информационных систем и технологий»

Кафедра «Информационные системы»

Дисциплина «Программирование»

**КУРСОВОЙ ПРОЕКТ (РАБОТА)**

Тема «Тема работы»\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Выполнил студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /И. И. Иванов/

подпись инициалы, фамилия

Курс 1 Группа ИСЭбд – 11

Направление/ специальность «Прикладная информатика (в экономике)»

Руководитель ассистент кафедры «Информационные системы» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Эгов Евгений Николаевич \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата сдачи:

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_\_г.

Дата защиты:

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_\_г.

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

Ульяновск

2014 г.

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет «Информационных систем и технологий»

Кафедра «Информационные системы»

Дисциплина «Программирование»

**ЗАДАНИЕ НА КУРСОВОЙ ПРОЕКТ (РАБОТУ)**

студенту ИСЭбд – 11 . Иванова Ивана Ивановича \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Тема проекта (работы) «Тема работы»\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Срок сдачи законченного проекта (работы) «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_\_г.

Исходные данные к проекту (работе) задание преподавателя кафедры «Информационные системы», Эгова Е. Н.\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ .

Содержание пояснительной записки (перечень подлежащих разработке вопросов)

Описание задачи, описание алгоритмов решения, описания разработки и эксплуатации системы .

Перечень графического материала (с точным указанием обязательных чертежей)

Диаграмма связей, блок-схемы

Руководитель ассистент кафедры «ИС» \_\_\_\_\_\_\_\_\_\_\_\_\_ /Е. Н. Эгов/

подпись

«\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_г

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ **/**И. И.Иванов/

подпись

«\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_г

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

**ОТЗЫВ**

**руководителя на курсовой проект (работу)**

студента Иванова Ивана Ивановича

Факультет ИСТ группа ИСЭбд – 11 курс 1

Дисциплина «Программирование» .

Тема проекта (работы) «Тема работы»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель ассистент кафедры «ИС» \_\_\_\_\_\_\_\_\_\_\_\_\_ /Е. Н. Эгов/

подпись инициалы, фамилия

«\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_г.

1. **Учет топлива на автозаправочной станции.** Имеется несколько видов топлива на АЗС. Учет продаж (суммовой и количественный) ведется в разрезах продавцов и видов топлива. Пополнение запасов происходит путем закупки требуемых материалов у сторонних контрагентов.