Министерство образования Республики Беларусь  
Учреждение образования  
«Брестский государственный технический университет»  
Кафедра интеллектуальных информационных технологий

Отчет

По дисциплине «Объектное моделирование интеллектуальных систем»

По лабораторной работе № 4-6

«Проектирование приложений с использованием UML»

Выполнил:  
Студент 3 курса  
Группы ИИ-22  
Копанчук Е. Р.

Проверил:  
Монтик Н. С.

Брест 2023

**Цель работы:** изучить содержание типовых этапов объектно-ориентированной разработки программ. Изучить использование диаграмм UML при разработке программ. Выполнить объектно-ориентированный анализ с элементами проектирования и реализации приложения с документированием результатов на языке UML.

**Постановка задачи:**

1. Тип ЗАПИСЕЙ: е – группа, староста, численность студентов.

2. Тип ХРАНЕНИЯ СПИСКА записей: в – как произвольный файл.

3. Тип СПИСКА: б – упорядоченный (отсортированный) по заданному полю (полям) записей.

4. Тип КАРКАСА приложения (интерфейса): а – на базе окна типа главное с управлением задачами (прецедентами) с помощью меню;

5. Тип организации АВТОРИЗАЦИИ пользователей: а – без регистрации и авторизации пользователей;

6. Типы ОПЕРАЦИЙ со СПИСКОМ: в – создание нового списка с ”ручным” заполнением начальным содержимым; г – вывод содержимого списка в виде перечня записей; д – просмотр содержимого списка ”запись за записью”; е – поиск первой записи списка (по заданному номеру, по шаблону);

7. Типы ОПЕРАЦИЙ с ЗАПИСЬЮ: в – удаление записи (с конца списка, с начала списка, указанной (-ых) записей, в том числе найденной (-ых) записей).

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Вариант** | **Параметры** | | | | | | |
| **1** | **2** | **3** | **4\*** | **5** | **6** | **7** |
| 5 | е | в (произвольный файл) | б | а | а (без авторизации) | в, д, г, е | в |

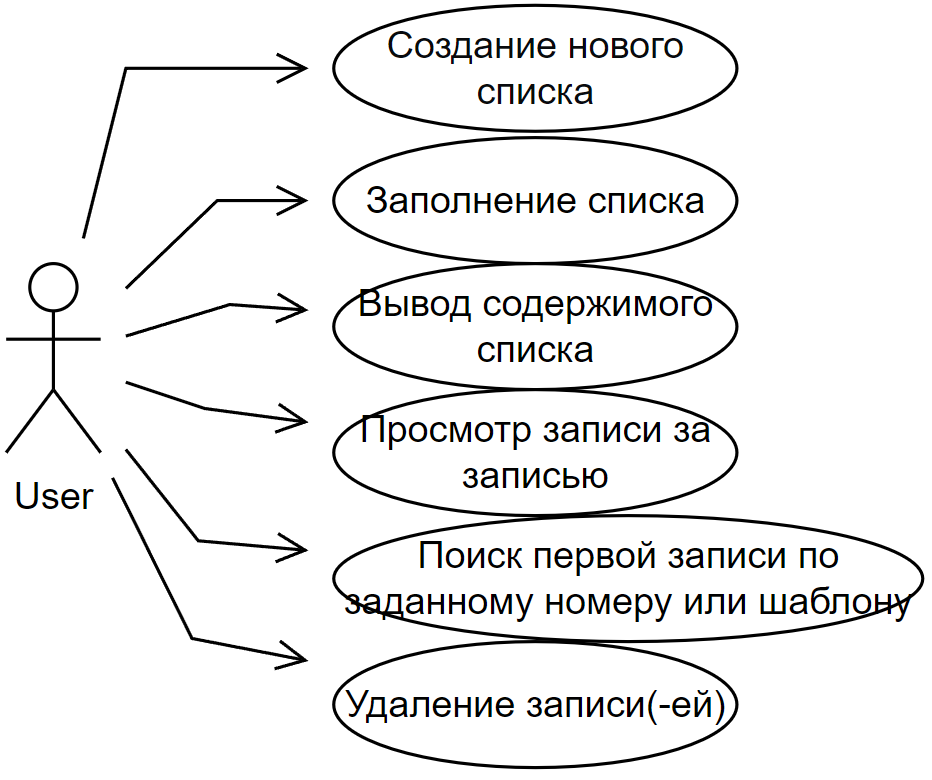
**Перечень этапов:**

|  |
| --- |
| **Выполняемые работы** |
| **ЭТАП 1 – Объектно-ориентированный анализ предметной области** |
| 1. Описание предметной области, автоматизируемых задач (результаты интервью) |
| 2. Выявление и описание прецедентов (построение диаграммы и описание потоков) |
| 3. Прототипирование приложения:  - первоначальное выявление интерфейсных форм (сопутствующих классов);  - дизайн форм;  - описание диаграммы состояний приложения;  - описание диаграмм состояний для прецедентов;  - описание диаграмм состояний для отдельных форм |
| 4. Выявление, первоначальное описание классов, классов предметной области, диаграммы классов предметной области на уровне “первичных” отношений |
| 5. Первичное описание структуры приложения и обработчиков |
| **ЭТАП 2 - Объектно-ориентированное проектирование** |
| 6. Описание диаграмм последовательностей, видов деятельности прецедентов |
| 7. Уточненное описание структуры классов, диаграммы классов предметной области |
| 8. Описание диаграмм видов деятельности методов |
| Реализация интерфейса приложения (макетирование) |
| **ЭТАП 3 – Объектно-ориентированное проектирование и реализация** |
| 10. Описание структуры приложения – диаграммы пакетов, компонентов |
| 11. Описание диаграммы развертывания приложения |
| 12. Реализация приложения, тестирование, документирование |

**1.**

Список записей заданного типа с возможностью осуществления операций как с самим списком, так и с составляющими его записями. Варианты организации списка, типы записей, а также варианты работы с ними приведены выше. Для работы со списком может потребоваться регистрация и авторизация пользователей. Информация (сам список, логины) может храниться в “системном” или произвольных файлах.

**2.**



|  |  |
| --- | --- |
| **Поток** | **Описание** |
| Создание нового списка | Пользователь выбирает опцию создания нового списка. Система запрашивает у пользователя тип записей (группа, староста, численность студентов). В случае выбора произвольного файла в качестве типа хранения, система запрашивает у пользователя путь к файлу. Система создает новый список с указанными характеристиками. |
| Заполнение списка | Пользователь выбирает опцию "Ручное заполнение" списка. Система запрашивает у пользователя данные для каждой записи (группа, староста, численность студентов). Система добавляет новую запись в список. Пользователь может повторять этот шаг для добавления нескольких записей. |
| Вывод содержимого списка | Пользователь выбирает опцию вывода содержимого списка. Система отображает содержимое списка в виде перечня записей, упорядоченных по заданному полю (если список был отсортирован). |
| Просмотр записи за записью | Пользователь выбирает опцию просмотра содержимого списка "запись за записью". Система последовательно отображает каждую запись списка. |
| Поиск первой записи по заданному номеру или шаблону | Пользователь выбирает опцию поиска первой записи списка. Пользователь указывает номер или шаблон для поиска. Система ищет первую запись, соответствующую заданным критериям, и отображает её. |
| Удаление записи(-ей) | Пользователь выбирает опцию удаления записи(-ей). Пользователь указывает, какую запись нужно удалить (с конца списка, с начала списка или указанные записи). Система удаляет указанную(-ые) запись(-и) из списка. |

**3.**

Используем язык программирования C# и Windows Forms для создания оконного интерфейса.

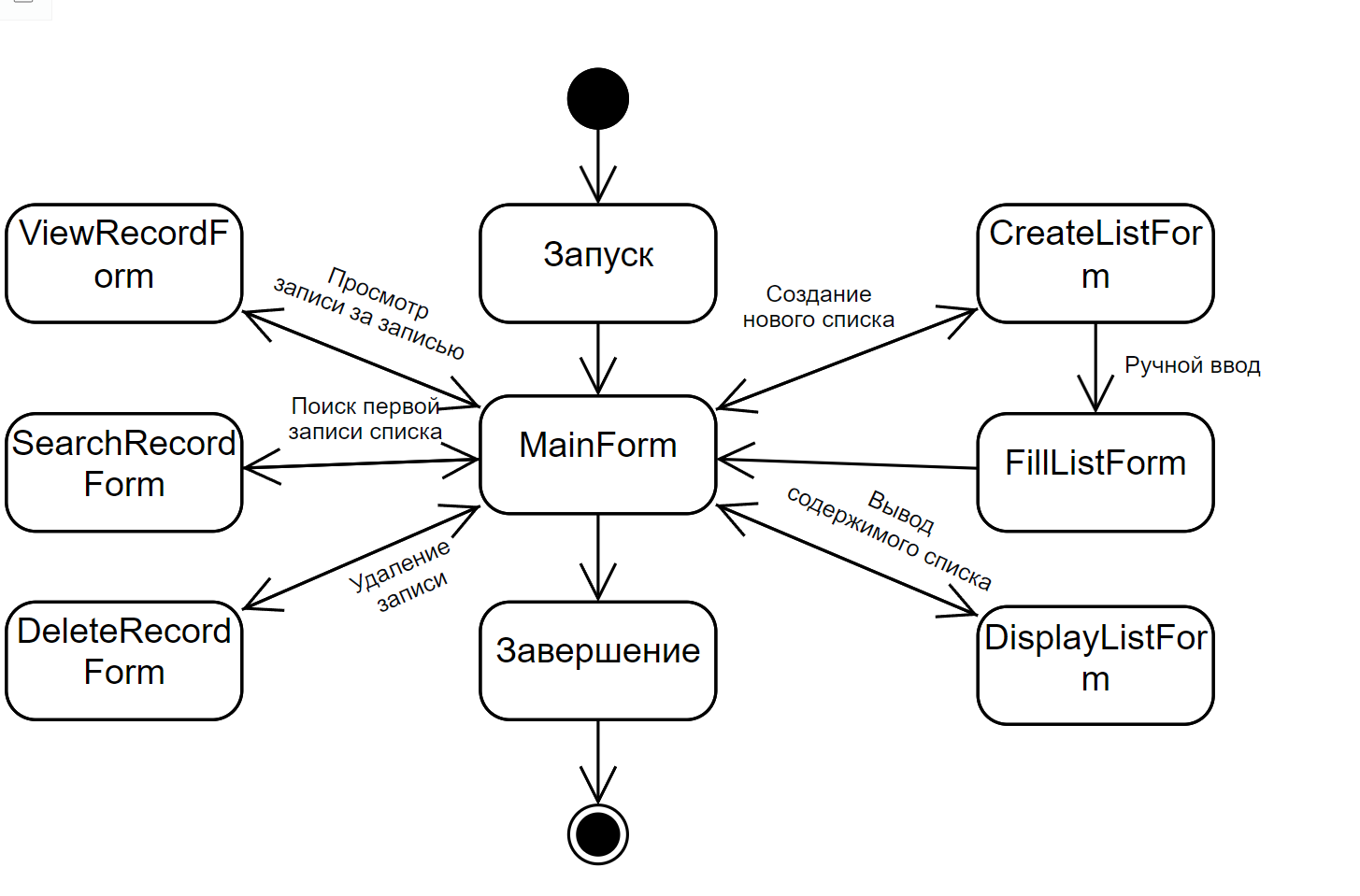
**Интерфейсные формы и сопутствующие классы:**

* + Форма "Главное меню" (MainForm): отображает список доступных задач и предоставляет кнопки для выбора операций с записями.
  + Форма "Создание списка" (CreateListForm): позволяет пользователю выбрать тип записей и тип хранения списка.
  + Форма "Заполнение списка" (FillListForm): позволяет пользователю вводить данные для каждой записи и добавлять их в список.
  + Форма "Вывод списка" (DisplayListForm): отображает содержимое списка в виде перечня записей, упорядоченных по выбранному полю.
  + Форма "Просмотр записей" (ViewRecordForm): позволяет пользователю последовательно просматривать каждую запись списка.
  + Форма "Поиск записи" (SearchRecordForm): позволяет пользователю указать номер или шаблон для поиска и отображает первую найденную запись.
  + Форма "Удаление записи" (DeleteRecordForm): позволяет пользователю выбрать метод удаления записи (с конца списка, с начала списка, указанную запись) и удалить ее.

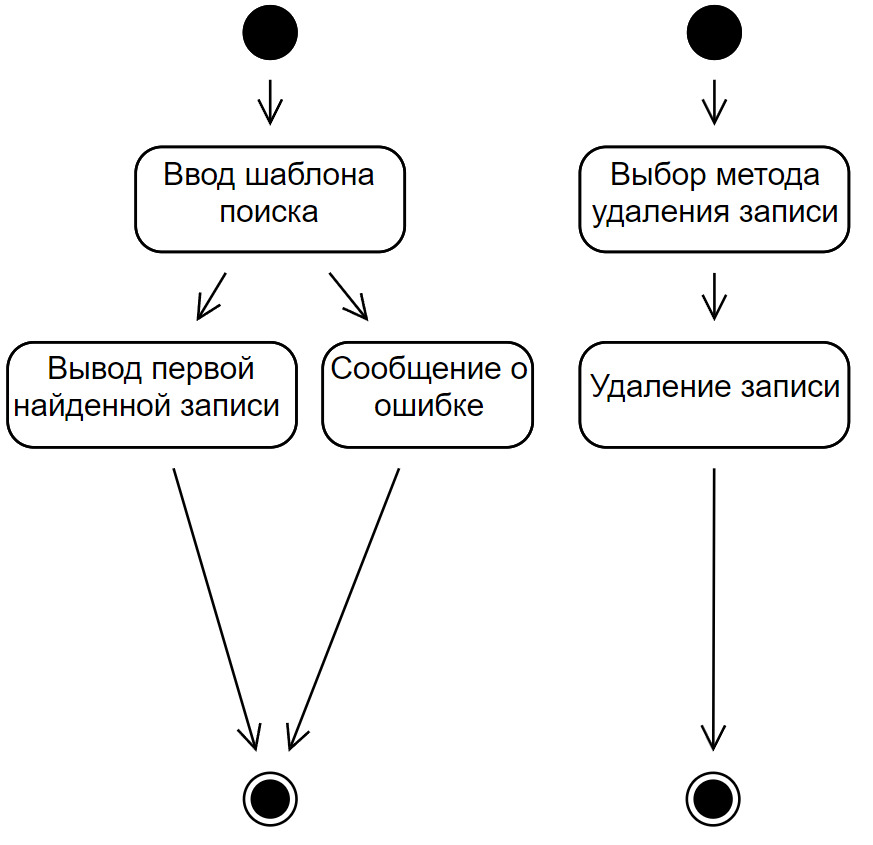
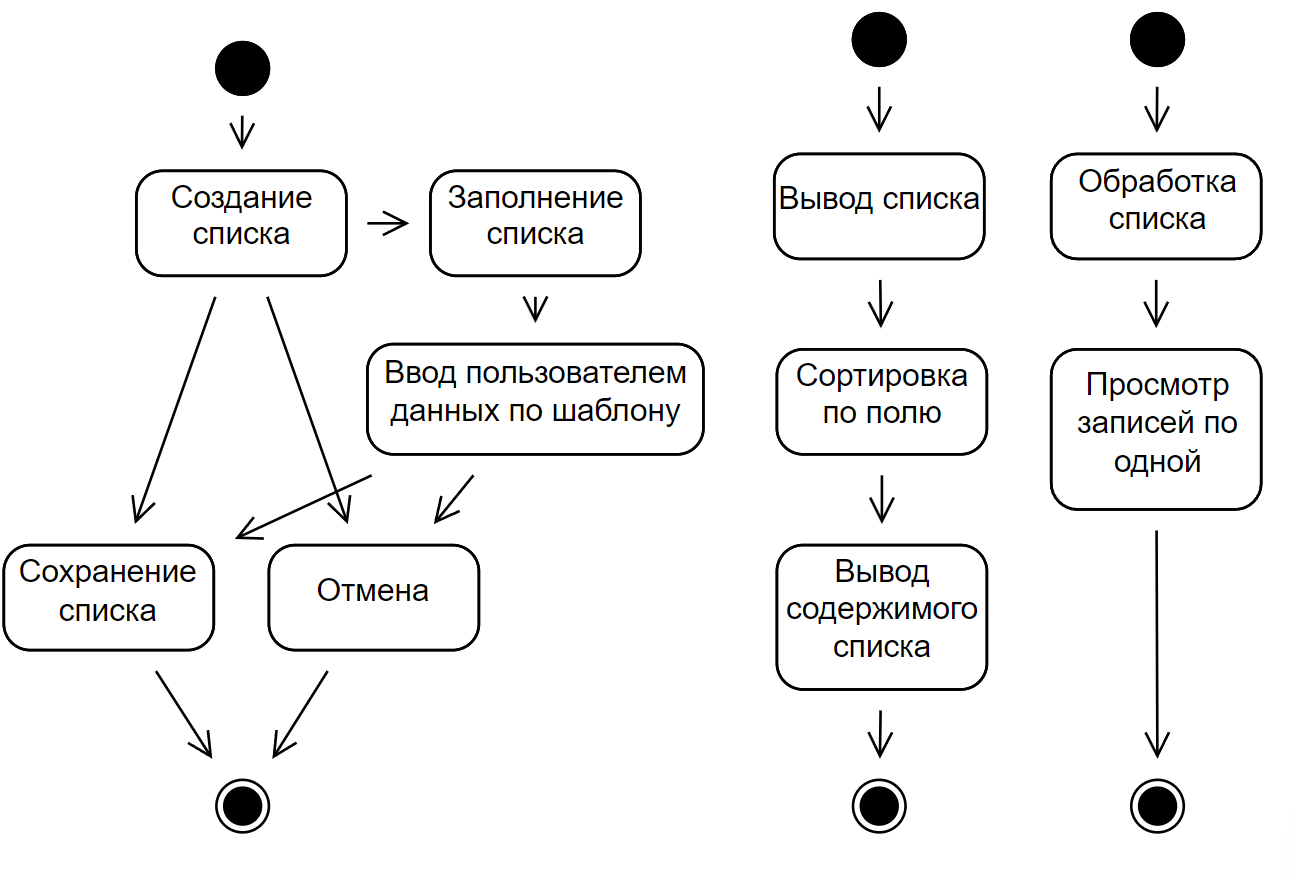
**Дизайн форм:**

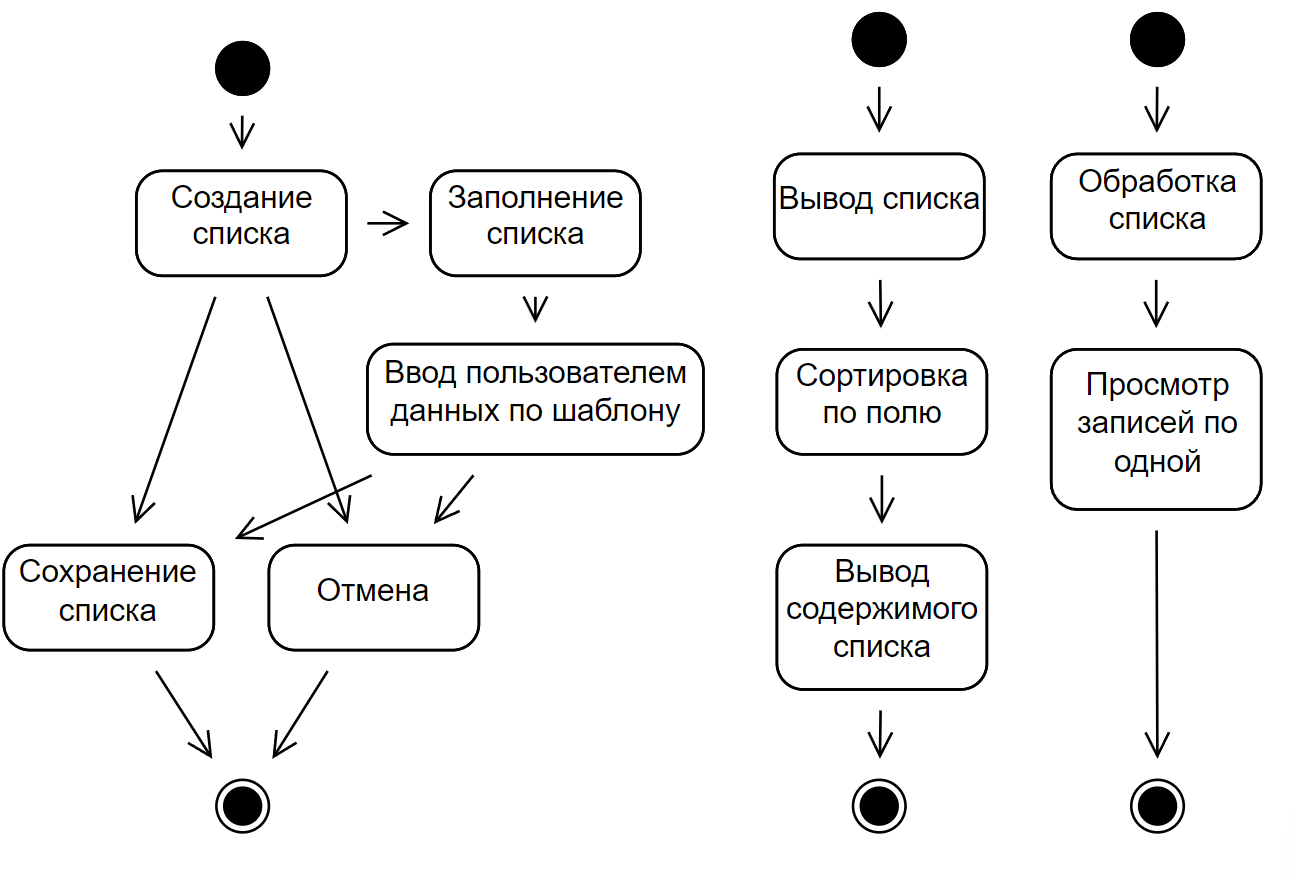
* + Форма MainForm: содержит меню с кнопками для выбора задач и операций со списком записей.
  + Форма CreateListForm: содержит элементы управления для выбора типа записей и типа хранения списка.
  + Форма FillListForm: содержит текстовые поля для ввода данных записей и кнопку для добавления их в список.
  + Форма DisplayListForm: отображает содержимое списка в виде ListBox или DataGridView, с возможностью выбора поля для сортировки.
  + Форма ViewRecordForm: содержит элементы управления для просмотра каждой записи списка.
  + Форма SearchRecordForm: содержит поле ввода номера или шаблона для поиска, кнопку для выполнения поиска и отображения найденной записи.
  + Форма DeleteRecordForm: содержит элементы управления для выбора метода удаления записи и кнопку удаления.

**Диаграмма состояний приложения:**



**Диаграммы состояний для прецедентов:**

**** 

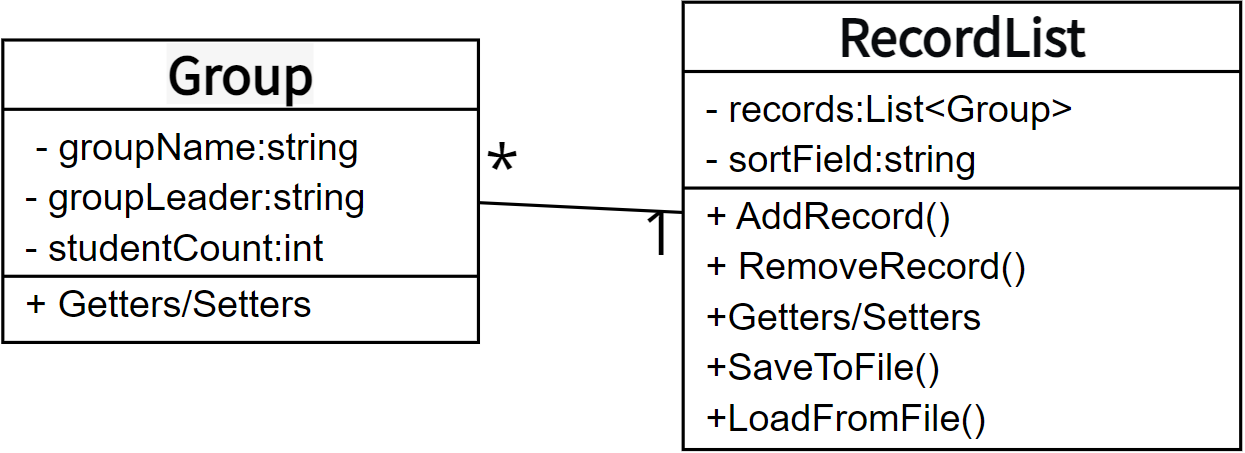


**4.**

**Выявление и первоначальное описание классов предметной области:**

1. Класс "Group" (Группа):
   * Поля:
     + string groupName: название группы
     + string groupLeader: имя и фамилия старосты
     + int studentCount: численность студентов в группе
     + Getters/Setters: геттеры и сеттеры для переменных
2. Класс "RecordList" (Список записей):
   * Поля:
     + List records: список записей типа "Группа"
     + string sortField: поле, по которому осуществляется сортировка списка
   * Методы:
     + void AddRecord(Group group): добавляет новую запись группы в список
     + void RemoveRecord(Group group): удаляет указанную запись группы из списка
     + void SaveToFile (Group group): сохраняет измененный список в файл
     + void LoadFromFile (Group group): загружает список из файла
3. Классы интерфейсных форм и сопутствующие классы:
   * MainForm: форма главного меню
   * CreateListForm: форма создания списка записей
   * FillListForm: форма заполнения списка записей
   * DisplayListForm: форма вывода списка записей
   * ViewRecordForm: форма просмотра записей
   * SearchRecordForm: форма поиска записей
   * DeleteRecordForm: форма удаления записей

**Диаграммы классов предметной области на уровне “первичных” отношений:**

****

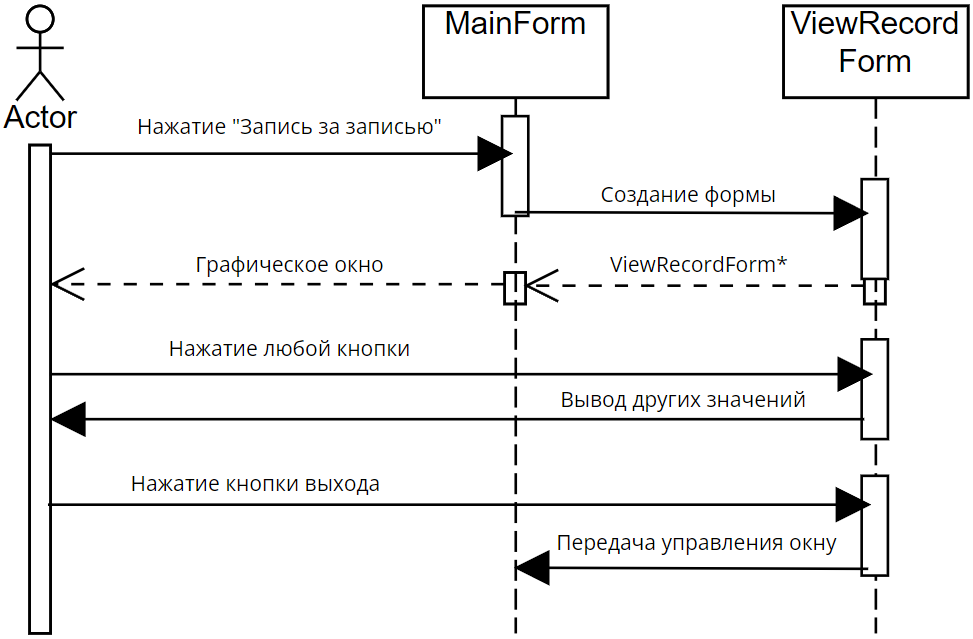
**5.**

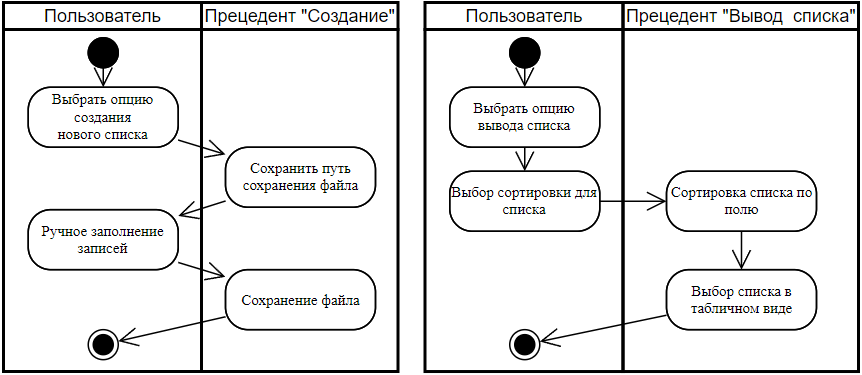
**Первичное описание структуры приложения и обработчиков:**

* MainForm: отображает основное меню с возможностью выбора задач, например, "Создание списка", "Вывод списка", "Просмотр записей", "Поиск записи", "Удаление записи".
* CreateListForm: пользователь выбирает тип записей и тип хранения списка. После выбора происходит создание объекта RecordList.
* FillListForm: пользователь вводит данные для каждой группы, после чего новая запись добавляется в объект RecordList.
* DisplayListForm: отображает список записей в виде перечня, упорядоченного по выбранному полю из объекта RecordList.
* ViewRecordForm: поочередно отображает каждую запись из объекта RecordList, позволяя пользователю просмотреть детали каждой группы.
* SearchRecordForm: пользователь указывает номер или шаблон для поиска записи. Поиск происходит в объекте RecordList, и первая найденная запись отображается на форме.
* DeleteRecordForm: пользователь выбирает метод удаления записи (с конца списка, с начала списка, указанную запись). Запись удаляется из объекта RecordList.

**6.**

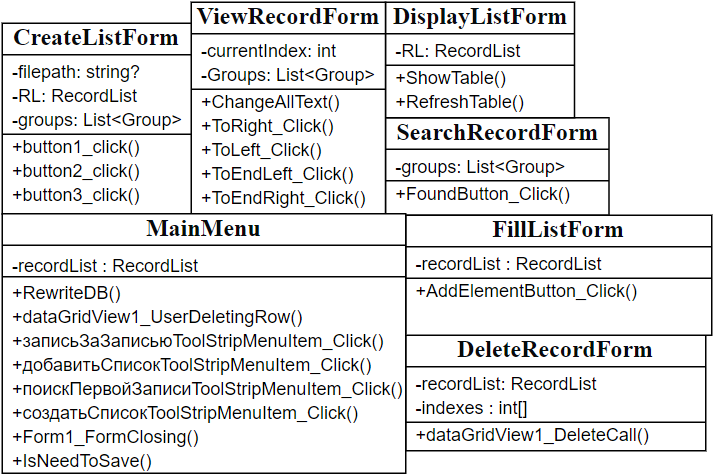
**Описание диаграмм последовательностей и видов деятельности прецедентов:**

****

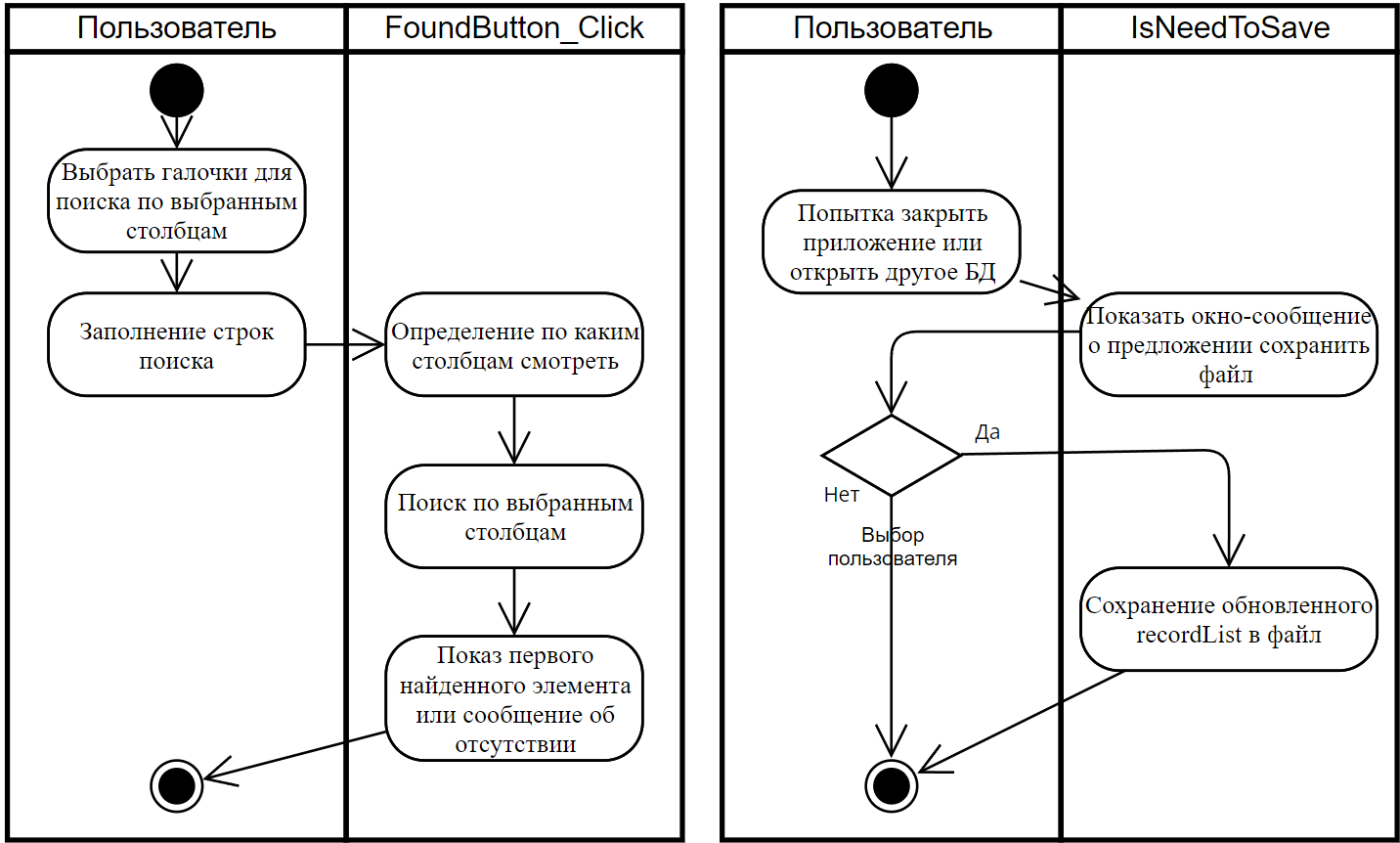
****

**7.**

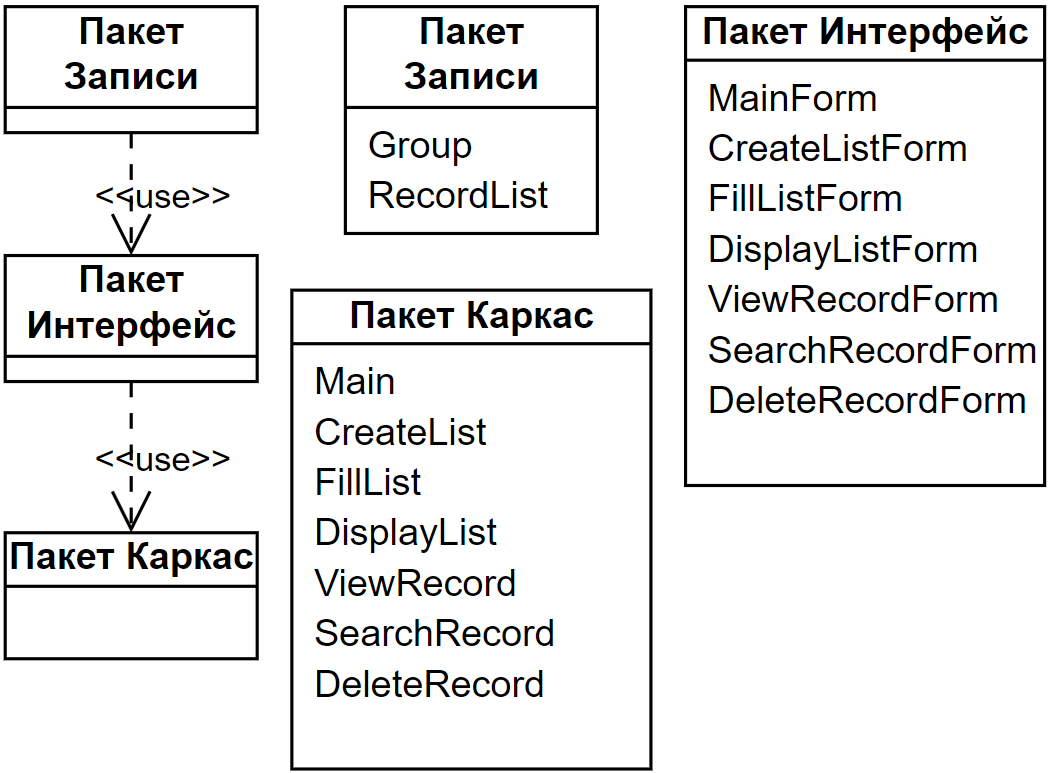
Уточненное описание классов "Group" (Группа), "RecordList" (Список записей) присутствует в пункте 4.



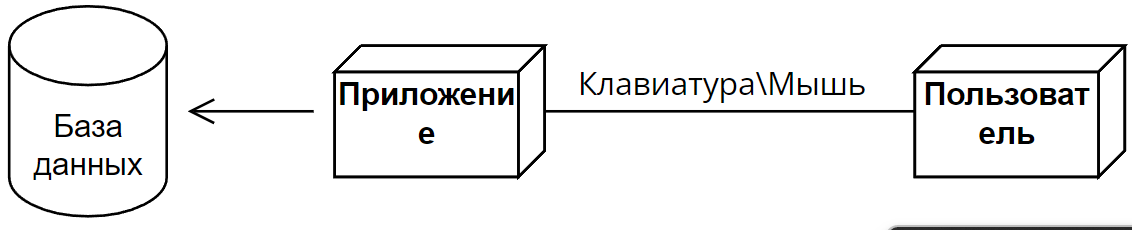
**8.**

****

**10.**

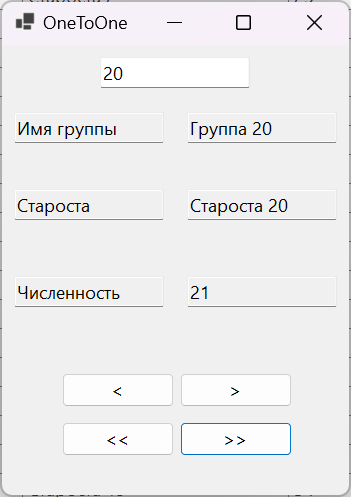
****

**11.**

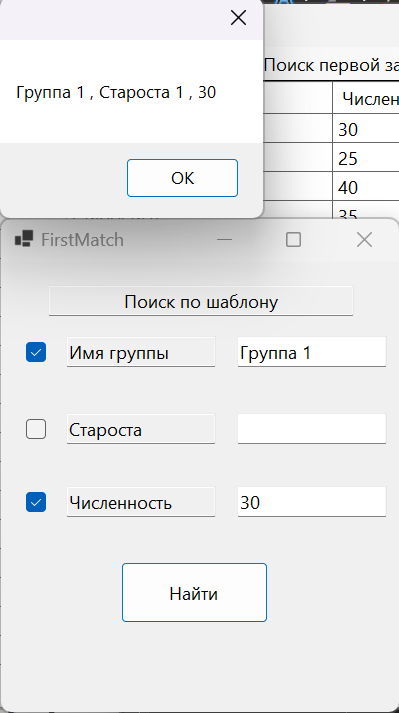
****

**12.**

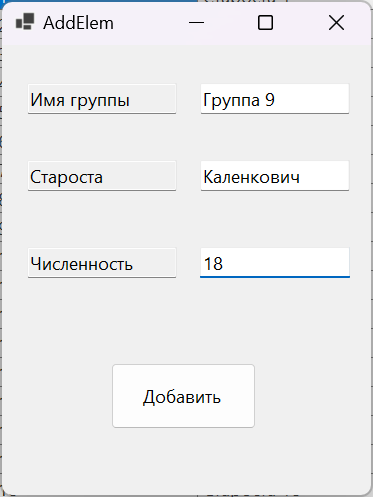
**Вывод записи за записью:**



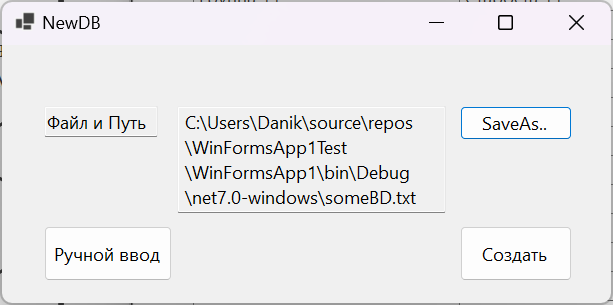
**Поиск записи:**

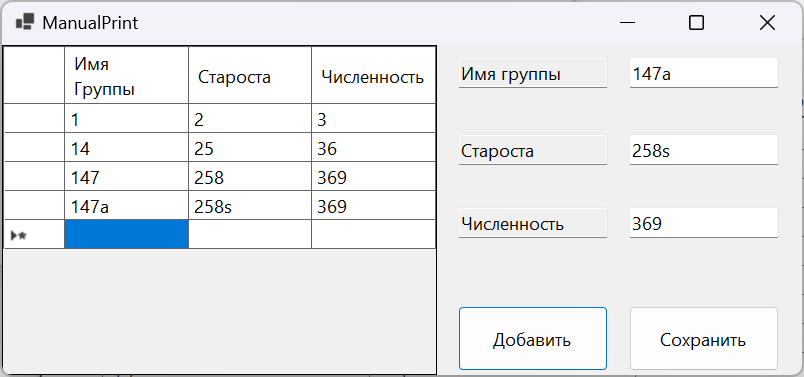
****

**Добавление записи:**



**Добавление списка:**





**Вывод:** изучили содержание типовых этапов объектно-ориентированной разработки программ. Изучили использование диаграмм UML при разработке программ. Выполнили объектно-ориентированный анализ с элементами проектирования и реализации приложения с документированием результатов на языке UML.