Практическая работа № 14

LINQ to Objects

Цель. Получить практические навыки разработки программы, использующей язык запросов LINQ to Objects.

Теоретические сведения

Постановка задачи

1. Сформировать обобщенную коллекцию (лабораторная работа № 11), содержащую ссылки на другие коллекции.
2. Заполнить коллекции объектами иерархии классов (лабораторная работа №10).

Выполнить запросы функции (всего должно быть выполнено не менее 5 запросов):

a) На выборку данных.
b) Получение счетчика (количества объектов с заданным параметром).
c) Использование операций над множествами (пересечение, объединение, разность).
d) Агрегирование данных.

Запросы должны быть выполнены двумя способами:

a) С использованием LINQ запросов.
b) С использованием расширенных методов.

Каждый запрос выполняется в отдельной функции.

Примеры запросов (лабораторная работа №10).

Варианты

Коллекция_1	Коллекция_2	Иерархия классов
Вуз	Факультеты вуза	студент, преподаватель, **персона**, сотрудник;
Завод	Цеха завода	служащий, **персона**, рабочий, инженер;
Предприятие	Отделы предприятия	рабочий, **персона**, инженер, администрация;
Город	Район	**организация,** страховая компания, судостроительная компания, завод, библиотека;

Запросы на выборку

1. Имена всех лиц мужского (женского) пола.

2. Имена студентов указанного курса.
3. Имена и должность преподавателей указанной кафедры.
4. Имена служащих со стажем не менее заданного.
5. Имена служащих заданной профессии.
6. Имена рабочих заданного цеха.
7. Имена рабочих заданной профессии.
8. Имена студентов, сдавших все (заданный) экзамены на отлично (хорошо и отлично).
9. Имена всех монархов на заданном континенте.
10. Наименование всех деталей (узлов), входящих в заданный узел (механизм).
11. Наименование всех книг в библиотеке (магазине), вышедших не ранее указанного года.
12. Названия всех городов заданной области.
13. Наименование всех товаров в заданном отделе магазина.
14. Наименование всех цехов на данном заводе.
15. Наименование птиц в зоопарке.
16. Имена пароходов, приписанных к данному порту.
17. Наименование журналов, выписываемых библиотекой.

Получение счетчика

18. Количество инженеров на заводе.
19. Количество мужчин (женщин).
20. Количество студентов на указанном курсе.
21. Количество рабочих со стажем не менее заданного.
22. Количество рабочих заданной профессии.
23. Количество инженеров в заданном подразделении.
24. Количество товара заданного наименования.
25. Количество студентов, сдавших все экзамены на отлично.
26. Количество студентов, не сдавших хотя бы один экзамен.
27. Количество деталей (узлов), входящих в заданный узел (механизм).
28. Количество указанного транспортного средства в автопарке (на автостоянке).
29. Количество пассажиров во всех вагонах экспресса.
30. Количество библиотек в городе.
31. Количество рабочих в заданном цехе.
32. Количество жителей данного континента.
33. Количество различных типов ДВС, обслуживаемых автомастерской.
34. Количество книг во всех библиотеках города.
35. Количество чеков на сумму превышающую заданную.

Операции над множествами

Агрегирование данных

36. Суммарная стоимость товара заданного наименования.
37. Средний балл за сессию заданного студента.
38. Суммарное количество учебников в библиотеке (магазине).
39. Суммарное количество жителей всех городов в области.

40. Суммарная стоимость продукции заданного наименования по всем накладным.
41. Средняя мощность всех (заданного типа) транспортных средств в организации.
42. Средняя мощность всех дизелей, обслуживаемых заданной фирмой.
43. Средний вес животных заданного вида в зоопарке.
44. Среднее водоизмещение всех парусников на верфи (в порту).
45. Суммарный вес всех деталей в заданном узле.
46. Суммарная стоимость всех деталей в механизме.
47. Суммарный страховой фонд всех страховых компаний региона.
48. Самый мощный автомобиль в данной организации.
49. Общая сумма по всем чекам, выписанным в организации.
50. Самая дорогая и самая дешевая игрушка в магазине(наименование и стоимость).

Цель работы: закрепление и углубление знаний, полученных студентами при изучении дисциплины «Программирование», а также получение практических навыков разработки программы средней сложности с использованием современных технологий и инструментальных средств.

В ходе выполнения работы студент получает профессиональные навыки в постановке задачи, анализе требований, выборе представления исходных данных и результата, разработке спецификаций, проектирования программной системы, написании программы на выбранном языке программирования с использованием объектно-ориентированной технологии и библиотек классов, тестировании и отладке программы, оформлении документации.

Задачей является разработка программной системы от начала (анализ требований) до конца (тестирование и сопровождение-документация).

Для описания модели используется язык **UML.**

Процесс проектирования – **Rational Unified Process(RUP).**

В качестве языка программирования можно использовать C++ , C#, Java, Perl, Python, PHP, Ruby.

В качестве среды разработки программы можно использовать Microsoft Visual Studio.NET, Eclipse, Rational Software Architect,Turbo JBuilder, Visual Studio Express Editor, NetBeans, Code::Blocks.

По умолчанию предполагается язык программирования C# или Java, среда разработки - Microsoft Visual Studio для языка C# и Rational Software Architect для языка Java.

Другие языки программирования и среды разработки должны быть согласованы с преподавателем до начала этапа реализации.

В любом случае следует использовать лицензионный программный продукт, либо свободное ПО. Приложение разрабатывается на платформе **Windows**.

Приложение должно иметь интерфейс **GUI.**

Конечным результатом работы является **проект**, содержащий **техническое задание** на разработку программной системы, **модель** разработанной системы на языке UML и **программный продукт** в виде исполняемого (exe) файла и исходных файлов, необходимых для сборки программы в среде разработки.

Вариант задания студент формулирует самостоятельно и в обязательном порядке согласовывает с преподавателем. В варианте дается общее описание разрабатываемой системы. На основе этого описания следует сформировать требования к программной системе и формализовать их в виде спецификаций вариантов использования (прецедентов).

В проекте должны быть представлены все стадии разработки:

- анализ и определение требований (результатом является набор моделей анализа и ТЗ);

- проектирование (архитектурное и детальное) (результатом является набор моделей проектирования, в том числе, при необходимости, схема БД);

- реализация (кодирование на выбранном языке программирования) (результатом является набор файлов, реализующих программную систему и, при необходимости БД, а также описание программы);

- тестирование (результатом является набор тестов, как минимум, «черный ящик»);

- сопровождение (документирование) (инструкция пользователю, инструкция программисту).

В соответствии с этими стадиями модель создается как Rational Unified Process (RUP) и содержит соответствующие пакеты

В проекте должна быть представлена UML- модель в виде следующих диаграмм:

- диаграммы вариантов использования (сценариев)- **Use case diagram**

- диаграммы активности(видов деятельности) – **Activity diagram**

- диаграммы взаимодействия – **Interaction diagram**

- диаграммы последовательностей –**Sequence diagram**

- диаграммы кооперации – **Collaboration diagram**

- диаграммы состояний – **Statechart diagram**

- диаграммы классов – **Class diagram**

- диаграммы компонент – **Component diagram**

Диаграммы должны иметь прикрепленные файлы с их описанием. Все окна документации для всех элементов модели должны быть заполнены.

В проекте должна быть предусмотрена генерация программного кода на основе диаграммы классов.

Кроме UML–модели проект содержит разработанную программную систему в виде исходных и исполняемого файлов, а также пояснительную записку.

Пояснительная записка должна содержать следующие разделы :

1.Титульный лист

2.Аннотация (объемом 0.5 страницы)

3.Оглавление

4.Постановка задачи, включая спецификацию требований на проектируемую программную систему. Здесь также должны быть описаны все прецеденты, представленные на **Use case diagram**

5.Описание модели поведения системы, представленной на диаграммах активности.

6.Описание модели взаимодействия , представленной на диаграммах последовательностей и кооперации.

7.Описание модели поведения, представленной на диаграммах состояний.

8.Описание логической структуры системы, представленной на диаграммах классов.

9.Описание физической структуры системы, представленной на диаграммах компонентов.

При описании диаграмм использовать ссылки на соответствующие артефакты.

11.Описание программы:

- определения классов;

- определения структур хранения данных, включая коллекции;

- определения основных обработчиков событий.

- описания нетривиальных алгоритмов.

Обратите внимание, что описание программы - это не код на языке программирования, а некоторый текст, объясняющий для чего используется тот или иной класс, структура, метод, как реализовано в программе взаимодействие элементов системы и т.д.

12.Результаты тестирования.

13.Заключение - вывод о том, насколько разработанная система удовлетворяет требованиям (смотри раздел 4) и предложения по дальнейшему развитию системы.

14.Список использованной литературы.

15.Приложение

1. Техническое задание.
2. Use case diagram.
3. Activity diagram.
4. Sequence diagram.
5. Collaboration diagram.
6. Statechart diagram.
7. Class diagram.
8. Component diagram.
9. Схема БД (если нужна).
10. Руководство пользователя.
11. Руководство программиста (перечисляются все файлы проекта с указанием их назначения и описывается процесс создания исполняемого кода программы (компиляция и компоновка).
12. .Листинг программы с комментариями.

Преподавателю необходимо представить:

1.Отпечатанную пояснительную записку (оформление по ГОСТУ)

2.CD содержащий :

- электронную версию пояснительной записки:

- исполняемый (exe) файл;

- данные для тестирования программы;

- исходные файлы программы;

- файл(ы) моделей.

CD должен быть в подписанном конверте.

Методические указания к этапам выполнения работы

I этап.

В соответствии с заданием выявляются функциональные и нефункциональные требования к проектируемой программной системе. Составляется документ, описывающий пользовательские и системные требования- **спецификация требований** ([1], § 5.4).

При выполнении этого этапа можно использовать следующие источники:

[1] часть 2. Требования

[2] глава 3. Установление требований

[3] глава 9 Технологический процесс управления требованиями

Результатом выполнения этого этапа является документ **«Спецификация требований на программный продукт»**

II этап

На этом этапе строится модель вариантов использования. Модель включает диаграмму Use Case и файлы с описанием артефактов модели. Полезно построить RAD средствами начальный прототип системы.

Результатом выполнения этого этапа являются диаграмма Use Case в виде файла и печатного документа, описание прецедентов и потоков событий для каждого прецедента в виде текстовых файлов и печатных документов, а также техническое задание на разработку программной системы.

III этап

Строится модель анализа, которая включает диаграммы деятельности, диаграммы классов модели анализа, диаграмму последовательности модели анализа и файлы с описанием артефактов модели. Уточняется прототип системы.

Результатом выполнения этого этапа являются диаграммы классов, деятельности, последовательности, описание состояний деятельности и классов с их обязанностями в виде текстовых файлов и печатных документов, прототип системы в виде исполняемого файла.

Для выявления классов используйте именные группы, общие шаблоны, описания прецедентов, CRC (Class – Responsibility – Collaborators).

IV этап

Строится модель проектирования, которая включает диаграммы последовательности, состояний, классов модели проектирования и файлы с описанием артефактов модели.

Результатом выполнения этого этапа являются диаграммы классов, состояний, последовательности, описание состояний, взаимодействий, объектов и классов в виде текстовых файлов и печатных документов.

V этап

Единственная диаграмма, которая здесь строится - это диаграмма компонентов. Остальные диаграммы могут уточняться.

На этом этапе на основе диаграммы классов, отражающих логику системы, выполняется автоматическая генерация скелета проектируемой программной системы, если среда моделирования позволяет это сделать. Затем добавляется и отлаживается остальной код.

Интерфейсная часть программы не входит в UML-модель и разрабатывается с использованием любой библиотеки классов, поддерживающей GUI программирование.

Результатом выполнения этого этапа являются диаграмма компонентов, программа в виде исходных и исполняемого файлов, тестовые данные и результаты тестирования, описание программы.

Литература

1. Соммервилл Ион. Инженерия программного обеспечения, 6-е издание- М.: Издательский дом «Вильямс», 2002

2. Мацяшек Лешек А. Анализ требований и проектирование систем- М.: Издательский дом «Вильямс», 2002

3. Кратчен Филипп. Введение в Rational Unified Process, 2-е издание- М.: Издательский дом «Вильямс», 2002