МЕЖГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»

Кафедра «Автоматизированные системы управления»

ТЕХНОЛОГИИ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Методические рекомендации к лабораторным работам
для студентов специальности
1-40 05 01 «Информационные системы и технологии
(по направлениям)»
очной и заочной форм обучения

Могилев 2023

УДК 004.45
ББК 32.973-018.2
Т38

Рекомендовано к изданию
учебно-методическим отделом
Белорусско-Российского университета

Одобрено кафедрой «Автоматизированные системы управления» «31» августа 2023 г., протокол № 1

Составитель ст. преподаватель Д. А. Денисевич

Рецензент канд. техн. наук, доц. А. Е. Науменко

Методические рекомендации к лабораторным работам предназначены для студентов специальности 1-40 05 01 «Информационные системы и технологии (по направлениям)» очной и заочной форм обучения.

Учебное издание

ТЕХНОЛОГИИ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Ответственный за выпуск	А. И. Якимов
Корректор	А. Т. Червинская
Компьютерная верстка	Е. В. Ковалевская

Подписано в печать . Формат 60×84/16. Бумага офсетная. Гарнитура Таймс.
Печать трафаретная. Усл. печ. л. . Уч.-изд. л. . Тираж 21 экз. Заказ №

Издатель и полиграфическое исполнение:
Межгосударственное образовательное учреждение высшего образования
«Белорусско-Российский университет».
Свидетельство о государственной регистрации издателя,
изготовителя, распространителя печатных изданий
№ 1/156 от 07.03.2019.
Пр-т Мира, 43, 212022, г. Могилев.

Содержание

Введение

Дисциплина «Технологии разработки программного обеспечения» формирует систематизированные знания о жизненном цикле разработки программного обеспечения и технологиях, применяемых на различных его этапах, включая моделирование предметной области, формализацию требований, алгоритмизацию проектных решений, программную реализацию и отладку приложений.

1 Лабораторная работа № 1. Анализ предметной области. Требования, виды требований, разработка прототипа технического задания в соответствии со стандартами

Цель работы: провести анализ предметной области и оформить результаты исследования деятельности предприятия.

Порядок выполнения работы

1 Изучить постановку задачи и порядок выполнения работы.
2 Выполнить практическое задание.
3 Сделать выводы по полученным результатам.
4 Ответить на контрольные вопросы.
5 Оформить отчет.

Постановка задачи

В процессе выполнения лабораторной работы необходимо провести анализ выбранной предметной области. В качестве примера рассмотрим предприятие «Мединтерпласт» и оформим результаты исследования деятельности. По итогам проведения исследования обычно формируется отчет.

Исследование начинается со сбора предварительной информации о компании. Итогом являются следующие данные:

– краткая информация о компании (профиль клиента);
– цели проекта;
– подразделения и пользователи системы.

На основе предварительной информации формируется и согласовывается с заказчиком общее представление о проекте.

Видение выполнения проекта и границы проекта – документ, который кратко описывает, в каких подразделениях и в какой функциональности будет внедряться информационные системы (ИС).

Затем выполняется детальное исследование предприятия, результаты которого оформляются в виде отдельного документа – отчета об исследовании.

Отчет об исследовании содержит следующие разделы:

– анализ существующего уровня автоматизации;
– список программного обеспечения, используемого в компании, и приводятся данные об использовании этих пакетов в каждом из подразделений организации;
– общие требования к ИС;
– общие требования к функциональности разрабатываемой системы.

Организационная диаграмма используется для отражения организационной структуры подразделений предприятия и их зон ответственности.

Все бизнес-процессы компании должны быть перечислены в общем списке и каждый должен иметь свой уникальный номер.

Для выделения автоматизируемых бизнес-процессов и их основных исполнителей используются диаграммы прецедентов.

Физическая диаграмма служит для того, чтобы описать взаимодействие организации на верхнем уровне с внешними контрагентами;

– описания бизнес-процессов (книга бизнес-процессов).

Далее в отчет об исследовании включается книга бизнес-процессов, содержащая подробное описание автоматизируемых бизнес-процессов. Модели бизнес-процессов позволяют выделить отдельные операции, выполнение которых должно поддерживаться разрабатываемой ИС.

На последнем этапе осуществляется отображение модели предметной области на функциональность типовой системы – выбираются модули системы для поддержки выделенных операций, определяются особенности их настройки, выявляется необходимость разработки дополнительных программных элементов.

Пример отчета исследования предметной области.

Краткая информация о компании «Мединтерпласт».

Предприятие производит препараты двух лекарственных форм:

1) лекарственные средства в твердых желатиновых капсулах;

2) порошки для приготовления растворов или суспензий для приема внутрь в пакетиках.

Производство осуществляется по полному циклу: от контроля поступающей субстанции до конечного продукта.

Все лекарственные препараты, выпускаемые предприятием, – препараты высокого европейского качества. Большинство из них являются первыми генериками импортных аналогов, что позволяет компании сделать эффективное и безопасное лечение финансово доступным для большего количества людей. Реализация лекарственных препаратов осуществляется через сеть аптек. Компания осуществляет доставку товаров как собственным транспортом, так и с помощью услуг сторонних организаций.

Основные бизнес-процессы компании – производство, закупки, складирование запасов, продажи, взаиморасчеты с поставщиками и клиентами.

Производство ИУПП «Мединтерпласт» находится в г. Несвиже Минской области.

По предварительным планам, компания намерена открыть также дочернее предприятие для организации производства в непосредственной близости к своим заказчикам.

Адрес: 222603, Республика Беларусь, Минская область, г. Несвиж, ул. Ленинская, 115, ком. 204, тел.: (+37517) 227-10-00, факс (+37517) 240-41-66, Директор: Тризонов Андрей Аркадьевич.

На момент проведения анализа штат компании состоит из 110 сотрудников.

Основными целями проекта автоматизация компании «Мединтерпласт» являются:

– разработка и внедрение комплексной автоматизированной системы поддержки логистических процессов компании;

– повышение эффективности работы всех подразделений компании и обеспечение ведения учета в единой информационной системе.

Не рассматривается в границах проекта автоматизация учета основных средств, расчета и начисления заработной платы, управления кадрами. Выходит за рамки проекта автоматизация процессов взаимоотношения с клиентами.

Количество рабочих мест пользователей – 50.

Отчет об исследовании.

Список программного обеспечения, используемого компанией на момент исследования:

1) 1С: Предприятие 8.3 («Бухгалтерия», «Торговля», «Зарплата», «Кадры», «Касса», «Банк») для работы бухгалтерии;

2) две собственные разработки на базе конфигуратора «1С» – «Закупки» и «Продажи»;

3) собственная разработка на базе NET для финансового отдела;

4) MS Excel 2019 для планирования продаж;

5) спутниковая система БЕЛТРАНС для группы логистики.

Общие требования к информационной системе.

Одно из основных требований компании «Мединтерпласт» к будущему решению состоит в том, чтобы оно было построено на фундаменте единой интегрированной системы, а работа всех сотрудников велась в одном информационном пространстве.

Ключевые функциональные требования к информационной системе:

– мощные средства защиты данных от несанкционированного доступа. Разграничения доступа к данным в соответствии с должностными обязанностями;

– возможность удаленного доступа;

– управление запасами. Оперативное получение информации об остатках на складе;

– управление закупками. Планирование закупок в разрезе поставщиков;

– управление продажами. Контроль лимита задолженности с возможностью блокировки формирования отгрузочных документов;

– полный контроль взаиморасчетов с поставщиками и клиентами;

– получение управленческих отчетов в необходимых аналитических срезах – как детальных для менеджеров, так и агрегированных для руководителей подразделений и директоров фирмы.

Организационная диаграмма.

Оргструктура предприятия оптовой торговли «Мединтерпласт» имеет вид, представленный на рисунке 1.

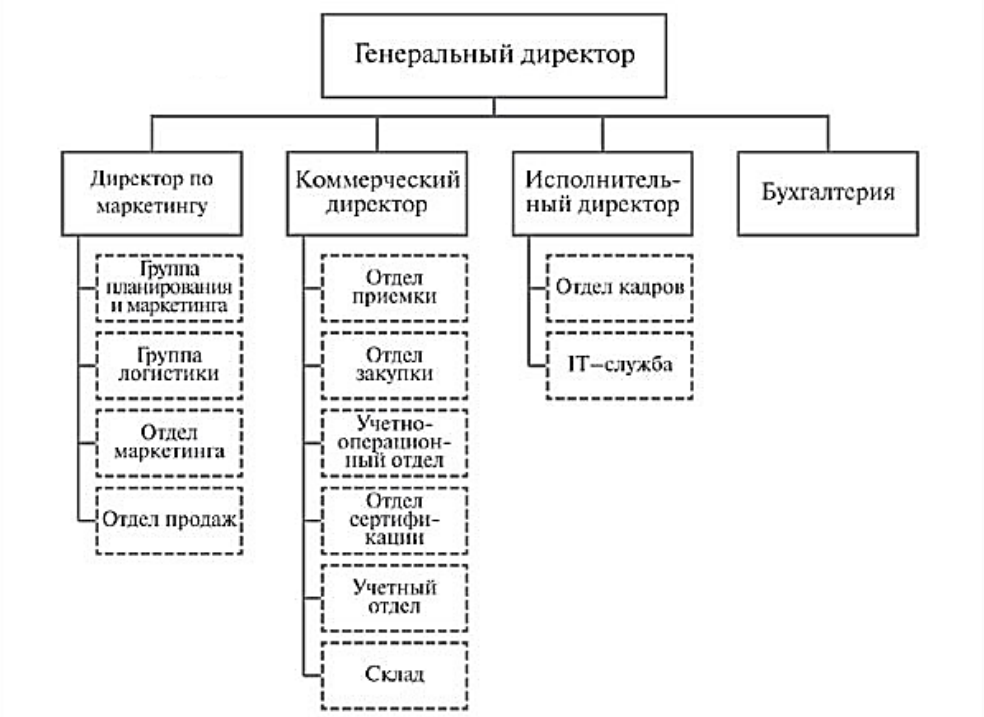

Рисунок 1 – Организационная диаграмма структуры предприятия «Мединтерпласт»

Задания для самостоятельной работы.

Провести анализ предметной области согласно варианту. Номер варианта соответствует номеру студента в журнале группы. Список предметных областей приведен ниже.

1 Компьютерный магазин.
2 Компания по производству и продаже нефтяных ископаемых.
3 Компания по производству обуви.
4 Секция игры в футбол.
5 Обучение иностранным языкам.
6 Автосервис.
7 Автомойка.
8 Автотранспортное предприятие.
9 Такси.
10 Склад готовой продукции.
11 Гостиница.
12 Бассейн.
13 Компания по транспортировке газа.
14 Транспортные происшествия.
15 Учет средств вычислительной и оргтехники.
16 Учет горюче-смазочных материалов на автобазе.
17 Банк данных туристических путевок сети турбюро.
18 Каталог изданий периодической печати.
19 Учет инвентаря на складах предприятия.

Результаты анализа оформить в виде отчета, в котором должны содержаться следующие разделы:

– краткая информация о компании;
– список программного обеспечения, используемого компанией на момент обследования;
– существующий уровень автоматизации;
– общие требования к информационной системе;
– организационная диаграмма структуры предприятия;
– описание состава автоматизируемых бизнес-процессов.

Содержание отчета.

1 Титульный лист.
2 Цель работы.
3 Исходные данные.
4 Ход выполнения работы.
5 Выводы.

Контрольные вопросы

1 Перечислите этапы анализа системы.
2 Перечислите разделы технического задания, согласно ГОСТу.
3 Опишите этап анализа предметной области.
4 Опишите этап анализа постановки задачи.

2 Лабораторная работа № 2. Разработка процессов или бизнес-процессов «as is – как есть» и «to be – как должно быть»

Цель работы: сформировать список бизнес-процессов предприятия, осуществить постановку задачи по обработке информации.

Порядок выполнения работы.

1 Изучить теоретические сведения.
2 Выполнить практическое задание.
3 Сделать выводы по полученным результатам.
4 Ответить на контрольные вопросы.
5 Оформить отчет.

Теоретические сведения.

Номер бизнес-процесса составляется из букв и цифр так, чтобы по номеру был интуитивно понятен смысл бизнес-процесса. Для того чтобы выделить бизнес-процессы, необходимо выделить действия, которые совершает компания. В рассматриваемом случае компания производит медицинские препараты,

планирует закупки, закупает медикаменты, доставляет медикаменты на склад, продает медикаменты.

Пример заполнения таблицы бизнес-процессов представлен в таблице 1.

Таблица 1 – Бизнес-процессы компании «Мединтерпласт»

Номер бизнес-процесса	Название бизнес-процесса
1Пл_Зак	Планирование закупок
2-Закпк	Закупки
3-Доствк	Доставка
4-Склад	Запасы-Склад
Примечание – В целях упрощения задачи в дальнейшем объединены описание бизнес-процессов «Закупки» и «Планирование закупок» в один бизнес-процесс под названием «Планирование закупок и размещение заказов»	

Бизнес-процесс «Планирование закупок и размещение заказов поставщикам». Общее описание бизнес-процесса.

Предприятие планирует закупки медикаментов. Планирование закупок осуществляется в Департаменте маркетинга, в группе маркетинга и планирования. Планирование закупок осуществляется следующим образом.

1 Менеджер группы планирования и маркетинга ежесуточно получает от контрагентов данные внешней и внутренней статистики продаж медикаментов в виде отчетов продаж.

2 Для планирования закупок медикаментов менеджер группы планирования и маркетинга еженедельно на основании статистики продаж производит расчет потребности в товаре. В результате расчета формируется таблица потребностей в товаре.

3 Определив количество и номенклатуру заказываемых товаров, менеджер отдела закупок приступает к анализу предложений поставщиков. *Данный* процесс осуществляется ежемесячно или по мере необходимости. Выбираются наиболее выгодные условия поставки. Для этого сравниваются цены поставщиков. Данные сведения берутся из прайс-листа для закупок. При выборе поставщика важно учесть предоставляемую отсрочку платежа. Эта информация берется из контрактов, отмеченных как приоритетные (действующие). В результате формируется список поставщиков, каждой позиции присваивается признак основного и запасных поставщиков в порядке убывания приоритета.

4 Менеджер отдела закупок ежемесячно на основании таблицы потребностей в товаре и списка выбранных поставщиков формирует графики поставок с указанием сроков и периодичности, но без количества поставки.

5 Ежемесячно после определения потребности в товаре менеджер группы логистики рассчитывает необходимое количество закупок. Необходимое количество закупок рассчитывается на основании фактических запасов на

складе, необходимого минимального и максимального уровней запасов. Нормы минимального и максимального количества запасов устанавливаются в днях. При расчете необходимого количества закупки учитывается также время товара в пути. Таким образом, данный расчет должен обеспечить возможность бесперебойного отпуска товара со склада. По результату расчетов формируется план заявок на месяц.

6 Затем в группе логистики ежедневно по плану заявок, графику поставок, прайс-листам поставщиков формируются заказы поставщикам.

7 Если предстоит сделать заказ импортному поставщику, то менеджер группы логистики рассчитывает затраты на сертификацию, создается отчет о затратах на сертификацию. Затраты на сертификацию проверяются на соответствие внутрифирменным нормам. Данная операция производится по мере необходимости.

8 Если затраты на сертификацию превышают внутрифирменные нормы, то менеджер группы логистики повторяет процесс формирования заказов поставщикам. Формируются новые заказы.

9 Ежедневно подготовленный заказ поставщику акцептуется, заказ должен подписать менеджер по логистике и директор Департамента маркетинга и управления товарными запасами.

Ежедневно менеджер группы логистики направляет заказ в отдел закупок. Менеджер отдела закупок направляет заказ поставщику.

Задания для самостоятельной работы.

На основании выбранной Вами предметной области в лабораторной работе № 1 определите бизнес-процессы (см. таблицу 1, примечание). Выделите действия к каждому бизнес-процессу, которые совершает компания и занесите их в таблицу. Каждому процессу дайте общее описание (более трех каждому).

Содержание отчета.

1 Титульный лист.
2 Цель работы.
3 Задание.
4 Ход выполнения работы.
5 Выводы.

Контрольные вопросы

1 Для чего нужна постановка задачи на разработку?
2 Назовите основные цели изучения предметной области.

3 Лабораторная работа № 3. Описание бизнес-процессов на языке BPMN

Цель работы: разработать диаграммы согласно методологии BPMN.

Порядок выполнения работы.
1 Изучить основные теоретические сведения.
2 Выполнить практическое задание.
3 Сделать выводы по полученным результатам.
4 Ответить на контрольные вопросы.
5 Оформить отчет.

Основные теоретические сведения.
Методология BPMN [7, с. 1–25].

Задания для самостоятельной работы.
1 Разработать контекстную диаграмму.
2 Разработать диаграмму декомпозиции первого уровня.

Содержание отчета.
1 Титульный лист.
2 Цель работы.
3 Задание.
4 Ход выполнения работы.
5 Выводы.

Контрольные вопросы

1 Что такое *ICOM*-коды?
2 Какие бывают типы стрелок?
3 Что такое словарь работ, стрелок?
4 Какие бывают типы связей работ?
5 Каким образом происходит слияние и расщепление стрелок?

4 Лабораторная работа № 4. Применение объектно- ориентированного подхода в анализе и проектировании программного обеспечения (UML)

Цель работы: познакомиться с языком графического описания для объектного моделирования UML, научиться на практике построению диаграммы вариантов использования.

Порядок выполнения работы.

1 Изучить основные теоретические сведения.
2 Выполнить практическое задание.
3 Сделать выводы по полученным результатам.
4 Ответить на контрольные вопросы.
5 Оформить отчет.

Основные теоретические сведения.

Разработка моделей с помощью Rational Rose [2, с. 5–10].

Rational Rose представляет собой case-средство проектирования и разработки информационных систем и программного обеспечения для управления предприятиями. Принципиальное отличие Rational Rose от других средств заключается в объектно-ориентированном подходе. Графические модели, создаваемые с помощью этого средства, основаны на объектно-ориентированных принципах и языке UML (Unified Modeling Language).

Моделирование бизнес-процессов в Rational Rose выполняется за счет применения различных аспектов. К таким аспектам относятся следующие:

– вариант использования (use-case). Этот аспект дает возможность понять, каким образом действуют участники процесса, и за счет этого определить их взаимодействие и влияние на процесс. Для построения моделей процесса в рамках данного аспекта применяются диаграммы use-case, диаграммы последовательностей, диаграммы совместной работы и диаграммы действий;

– логический аспект. С помощью этого аспекта можно определить функциональные требования процесса. Он задает логическую взаимосвязь между классами элементов процесса. Для построения моделей применяются диаграммы классов и диаграммы состояний;

– составляющие элементы. Этот аспект обращает внимание на состав элементов процесса и их распределение при создании информационной системы. Модели в этом аспекте строятся с помощью диаграммы компонентов. Она содержит информацию об элементах процесса и программном обеспечении;

– ввод в действие. Этот аспект показывает схему процесса в привязке к аппаратному обеспечению информационной системы. Для построения моделей применяется только одна диаграмма – диаграмма топологии.

Диаграмма вариантов использования отражает отношения между действующими лицами и прецедентами (вариантами использования).

Прецедент – возможность моделируемой системы (часть ее функциональности), благодаря которой пользователь может получить конкретный, измеримый и нужный ему результат.

Основное назначение диаграммы – описание функциональности и поведения, позволяющее заказчику, конечному пользователю и разработчику совместно обсуждать проектируемую или существующую систему.

Задания для самостоятельной работы.

Создать диаграмму вариантов использования на языке UML согласно выбранной предметной области.

Контрольные вопросы

1 Охарактеризуйте диаграмму вариантов использования.

2 Опишите назначение варианта использования и актера на диаграмме прецедентов.

3 Опишите отношения между вариантами использования на диаграмме прецедентов.

5 Лабораторная работа № 5. Диаграммы модели UML

Цель работы: научиться на практике построению диаграмм классов и диаграмм последовательности.

Порядок выполнения работы.

1 Изучить основные теоретические сведения.
2 Выполнить практическое задание.
3 Сделать выводы по полученным результатам.
4 Ответить на контрольные вопросы.
5 Оформить отчет.

Основные теоретические сведения.

Диаграмма классов UML является разновидностью статической структурной диаграммы, демонстрирующей классы системы, их атрибуты, операции (или методы) и взаимосвязи между объектами.

В верхней части диаграммы задается имя класса. Посередине располагаются поля (атрибуты) класса. Нижняя часть содержит методы класса.

Для задания видимости членов класса (любой атрибут или метод) эти обозначения должны быть размещены перед именем участника.

В UML представлены следующие виды отношений.

Зависимость обозначает такое отношение между классами, при котором изменение спецификации класса-поставщика может повлиять на работу зависимого класса, но не наоборот.

Ассоциация показывает, что объекты одной сущности (класса) связаны с объектами другой сущности таким образом, что можно перемещаться от объектов одного класса к другому. Ассоциация является общим случаем композиции и агрегации.

Агрегация – это разновидность ассоциации при отношении между целым и его частями. Одно отношение агрегации не может включать более двух классов (контейнер и содержимое).

Композиция – более строгий вариант агрегации по значению, при котором имеется жесткая зависимость между временем существования экземпляров класса и типом экземпляров классов.

Диаграмма последовательности наглядно отображает временной аспект взаимодействия. Она имеет два измерения. Одно измерение (слева направо) указывает на порядок вовлечения экземпляров сущностей во взаимодействие. Крайним слева на диаграмме отображается экземпляр действующего лица или объект, который является инициатором взаимодействия. Правее отображается другой экземпляр сущности, который непосредственно взаимодействует с первым, и т. д. Второе измерение (сверху вниз) указывает на порядок обмена сообщениями.

Задания для самостоятельной работы.

Создать диаграмму классов и диаграммы последовательности действий, представленных на диаграмме вариантов использования UML (см. лабораторную работу № 4) согласно выбранной предметной области.

Контрольные вопросы

1 Поясните назначение статических моделей объектно-ориентированных программных систем.

2 Что является основным средством для представления статических моделей?

3 Какие секции входят в графическое обозначение класса?

4 Поясните общий синтаксис представления свойства.

5 Какие уровни видимости вы знаете? Их смысл?

6 Лабораторная работа № 6. Разработка схем и программного кода. Отладка проекта программы

Цель работы: научиться созданию, компиляции, отладке и выполнению проектов в интегрированной среде разработки.

Порядок выполнения работы.

1 Изучить теоретические сведения, постановку задачи и порядок выполнения работы.

2 Выполнить практическое задание.

3 Сделать выводы по полученным результатам.

4 Ответить на контрольные вопросы.

5 Оформить отчет.

Теоретические сведения.

Платформа Microsoft.NET [6, с. 25–46].

Задание для самостоятельной работы.

Создание простой программы в текстовом редакторе.

В этом упражнении напишите программу на языке C#. В окне командной строки программа будет спрашивать, как Вас зовут и затем здороваться с Вами по имени.

Создайте новое консольное приложение C#:

− запустите Microsoft Visual Studio. NET. (Start→AllPrograms→VisualStudio. NET→ Microsoft Visual Studio.NET);

− выберите пункт меню File→New→Project;

− на панели Project Types выберите Visual C# Projects;

− на панели Templates выберите Console Application;

− в текстовое поле Name введите имя проекта Greetings;

− в поле Location укажите каталог для проекта и нажмите OK;

− измените имя класса на Greeter;

− сохраните проект, выбрав пункт меню File→Save All.

Напишите код, запрашивающий имя пользователя и приветствующий его по имени. В методе Main вставьте следующую строку кода:

```
string myName;
```

Напишите код, запрашивающий имя пользователя:

```
Console.WriteLine("Please enter your name");
```

Напишите код, считывающий введенное пользователем имя и присваивающий полученное значение строковой переменной myName:

```
myName = Console.ReadLine( );
```

Добавьте код, который будет выводить на экран строку “Hello myName”, где myName – имя, введенное пользователем.

```
Console.WriteLine("Hello, {0}", myName);
```

Итоговый текст метода Main должен выглядеть следующим образом:

```
static void Main(string[ ] args)
{
string myName;
Console.WriteLine("Please enter your name"); myName = Console.ReadLine( ); Console.WriteLine("Hello, {0}", myName);
}
```

Сохраните проект, откомпилируйте и запустите программу:

− выберите пункт меню Build→Build Solution (или Ctrl+Shift+B);

− при необходимости исправьте ошибки и откомпилируйте программу заново;

− выберите пункт меню Debug→Start Without Debugging (или Ctrl+F5);

− в появившемся окне введите свое имя и нажмите ENTER. Закройте приложение.

Использование отладчика Visual Studio .NET.

В этом задании Вы научитесь работать с интегрированным отладчиком.

Visual Studio .NET: проходить программу по шагам и просматривать значения переменных.

Поставьте точки остановки и запустите пошаговое выполнение:

− запустите Visual Studio .NET, если она не запущена;

− выберите пункт меню File→Open→Project;

− откройте проект Greetings.sln из папки install folder\Lab\Lab01\Greetings;

− в редакторе кода класса Greeter щелкните по крайнему левому полю на уровне строки кода, где впервые встречается команда Console.WriteLine;

− выберите пункт меню Debug→Start (или нажмите F5);

− программа запустится на выполнение, появится консольное окно и затем программа прервется в месте точки остановки.

Просмотрите значение переменной:

− выберите пункт меню Debug →Windows→Watch→Watch1;

− в окне Watch в список выражений для мониторинга добавьте переменную myName;

− в окне Watch появится переменная myName с текущим значением null.

Используйте команды пошагового выполнения:

− для выполнения первой команды Console.WriteLine выберите пункт меню Debug→Step Over (или нажмите F10);

− для выполнения следующей строчки кода, содержащей команду Console.ReadLine, снова нажмите F10;

− вернитесь в консольное окно, введите свое имя и нажмите ENTER;

− вернитесь в Visual Studio. Текущее значение переменной myName в окне Watch будет содержать Ваше имя;

− для выполнения следующей строчки кода, содержащей команду Console.WriteLine, снова нажмитеF10;

− разверните консольное окно. Там появилось приветствие;

− вернитесь в Visual Studio. Для завершения выполнения программы выберите пункт меню Debug→Continue (или нажмите F5).

Добавление в программу обработчика исключительных ситуаций.

В этом упражнении Вы напишите программу, в которой будет использоваться обработчик исключительных ситуаций, который будет отлавливать ошибки времени выполнения. Программа будет запрашивать у пользователя два целых числа, делить первое число на второе и выводить полученный результат.

Создайте новый проект.

Запустите Visual Studio .NET:

− выберите пункт меню File→New→Projec;

− на панели Project Types выберите Visual C# Projects;

− на панели Templates выберите Console Application;

− в текстовое поле Name введите имя проекта Divider;

− в поле Location укажите каталог для проекта install folder\Lab\Lab01 и нажмите OK;

− измените имя класса на DivideIt;

− сохраните проект, выбрав пункт меню File→Save All.

Напишите код, запрашивающий у пользователя два целых числа.

В методе Main напишите код, запрашивающий у пользователя первое целое число:

```
Console.WriteLine("Please enter the first integer");
```

Напишите код, считывающий введенное пользователем число и присваивающий полученное значение переменной temp типastring:

```
string temp = Console.ReadLine( );
```

Добавьте код, который переведет значение переменной temp из типа данныхstring вint и сохранит полученный результат в переменной i:

```
int i = Int32.Parse(temp);
```

Аналогичным образом создайте следующий код:

− запросите у пользователя второе целое число;

− считайте введенное пользователем число и присвойте полученное значение переменной temp;

− переведите значение переменной temp в тип данных int и сохраните полученный результат в переменной j.

Итоговый текст программы должен выглядеть следующим образом:

```
Console.WriteLine("Please enter the first integer"); string temp = Console.ReadLine( ); int i = Int32.Parse(temp);
Console.WriteLine("Please enter the second integer"); temp = Console.ReadLine( ); int j = Int32.Parse(temp);
```

Сохраните проект.

Разделите первое число на второе и выведите результат на экран:

− напишите код, создающий новую переменную k типа int, в которую будет заноситься результат деления числа i на j, и поместите его после кода, созданного в предыдущем пункте.int k = i / j;

− добавьте код, выводящий значение k на экран;

− сохраните проект.

Протестируйте программу.

Выберите пункт меню Debug→Start Without Debugging (или Ctrl+F5). Введите первое число 10 и нажмите ENTER.

Введите второе число 5 и нажмите ENTER.

Проверьте, что выводимое значение k будет равным 2.

Обратите внимание на типы данных. Подумайте, почему при делении, например, 10 на 3 получается 3.

Снова запустите программу на выполнение. Введите первое число 10 и нажмите ENTER. Введите второе число 0 и нажмите ENTER.

В программе возникнет исключительная ситуация (деление на ноль). Для очистки окна диалога Just-In-TimeDebugging выберите No.

Добавьте в программу обработчик исключительных ситуаций.

Поместите код метода Main внутрь блока try следующим образом:

```
try
{
Console.WriteLine (...);
...
int k = i / j; Console.WriteLine(...);
}
```

В методе Main после блока try добавьте блок catch, внутри которого должно выводиться краткое сообщение об ошибке:

```
catch(Exception e)
{
 Console.WriteLine("An exception was thrown: {0}", e);
}
...
```

Сохраните проект.

Итоговый текст метода Main должен выглядеть следующим образом:

```
public static void Main(string[ ] args)
{
try {
Console.WriteLine ("Please enter the first integer"); string temp = Console.ReadLine( );
int i = Int32.Parse(temp);
Console.WriteLine ("Please enter the second integer"); temp = Console.ReadLine( );
int j = Int32.Parse(temp); int k = i / j;
Console.WriteLine("The result of dividing {0} by {1} is {2}", i, j, k);
}
catch(Exception e) {
Console.WriteLine("An exception was thrown: {0}", e);
}
}
```

Протестируйте код обработчика исключительных ситуаций

Снова запустите программу на выполнение, нажав Ctrl+F5. Введите первое число 10 и нажмите ENTER.

Введите второе число 0 и нажмите ENTER.

В программе вновь возникнет исключительная ситуация (деление на ноль).

Содержание отчета.

1 Титульный лист.

2 Цель работы.

3 Задание.

4 Ход выполнения задания.

5 Выводы.

Контрольные вопросы

1 Перечислите и охарактеризуйте варианты выпуска среды программирования Microsoft Visual Sludio.

2 Какая информация находится в обозревателе решений?

3 Объясните назначение директивы using System?

4 Как обозначаются в программе комментарии?

5 Объясните использование в программе ключевого слова namespace?

6 Объясните использование в программе ключевого слова class?

7 Лабораторная работа № 7. Работа с системой контроля версий

Цель работы: изучить систему контроля версий Git и установить на компьютер.

Порядок выполнения работы.

1 Изучить краткие теоретические сведения.

2 Выполнить задание.

3 Сделать выводы по полученным результатам.

4 Ответить на контрольные вопросы.

5 Оформить отчет.

Краткие теоретические сведения.

В настоящее время владение Git стало обязательным требованием при приеме на работу не только для профессионалов, но даже для стажеров.

Git – это система контроля версий (СКВ). Существует несколько подобных систем, однако Git – на настоящий момент наиболее используемая СКВ.

Git создан для решения нескольких проблем любого программиста.

1 История изменений. Работа программиста – это всегда история изменений в коде программы:

– внося изменения в файлы, хочется знать ответ на вопросы «Кто сделал? Что сделал? Когда сделал?». Таким образом легко отслеживать, когда появились ошибки и кто их сделал. Любая система, которая позволит нам видеть такую историю изменений, и является системой контроля версий;

– иметь возможность отмены изменений или отката по истории назад (и вперед).

2 Легкость внесения изменений:

– если нужно попробовать какой-то вариант – это должно быть просто;

– вернуться на основной вариант – также просто;

– возможность легко принять или отвергнуть альтернативу.

3 Совместная работа:

– хорошо тому живется, у кого одна нога… Если ног несколько, возникает проблема синхронизации;

– изменения разных пользователей нужно изолировать друг от друга…;

– а по мере готовности – сливать вместе.

Программа Git решает все эти проблемы. Эта программа – набор скриптов, который умеет управлять изменениями. Он следит за файлами, ведет их историю, умеет ими манипулировать, откатывать, сливать и т. д.

Git используется в разработке ядра Linux и создан Линусом Торвальдсом.

Задача Git – вести полную историю изменений в некоей папке на сервере или локальном компьютере (репозиторий). К изменениям относятся добавление и удаление файлов, модификация их содержимого (нужно также учитывать, что Git не следит за пустыми папками, в папке нужно создать хотя бы один файл).

В историю также записывается, кто и когда сделал изменения.

Git – это распределенная система контроля версий, в ней нет центрального репозитория. Репозиторий – это просто папка с вашими файлами, в которой есть еще некая служебная информация для Gita. Их может быть очень много. Репозитории могут обмениваться изменениями между собой. Однако, это не DropBox, он не занимается хранением файлов, его задача следующая. Он может от одного репозитория к другому передать изменения!

Задание для самостоятельной работы.

1 Установка Git.

Для установки на Windows нужно перейти по ссылке.

git-for-windows.github.io

Данная версия программы представляет собой не только Git, но и нужное для его работы окружение, которое установится одним пакетом.

Нажимаем Download (вторая кнопка позволяет принять участие в разработке и нам пока не нужна). Устанавливаем все с настройками по умолчанию.

Содержание отчета.

1 Титульный лист.
2 Цель работы.
3 Задание.
4 Ход выполнения задания.
5 Выводы.

Контрольные вопросы

1 Что такое СКВ?
2 Какие проблемы решает Git?
3 Для разработки какой операционной системы используется Git?
4 Что такое репозиторий?
5 В чем разница между GoogleDrive/DropBox/YandexDisk и репозиторием Git?

8 Лабораторная работа № 8. Создание эргономичных пользовательских интерфейсов

Цель работы: познакомиться с основными элементами управления и приобрести навыки проектирования графического интерфейса пользователя.

Порядок выполнения работы.
1 Изучить теоретические сведения.
2 Выполнить задание.
3 Сделать выводы по полученным результатам.
4 Ответить на контрольные вопросы.
5 Оформить отчет.

Теоретические сведения.

Графический интерфейс пользователя (GUI) – разновидность пользовательского интерфейса, в котором элементы интерфейса (меню, кнопки, значки, списки и т. п.), представленные пользователю на дисплее, выполнены в виде графических изображений.

В GUI пользователь имеет произвольный доступ (с помощью устройств ввода – клавиатуры, мыши, джойстика и т. п.) ко всем видимым экранным объектам (элементам интерфейса) и осуществляет непосредственное манипулирование ими.

Графический интерфейс пользователя является частью пользовательского интерфейса и определяет взаимодействие с пользователем на уровне визуализированной информации.

Можно выделить следующие виды графического интерфейса пользователя:

– простой: типовые экранные формы и стандартные элементы интерфейса, обеспечиваемые самой подсистемой GUI;

– истинно-графический, двухмерный: нестандартные элементы интерфейса и оригинальные метафоры, реализованные собственными средствами приложения или сторонней библиотекой;

– трехмерный.

Проектирование графического интерфейса пользователя представляет собой междисциплинарную деятельность. Оно требует усилий многофункциональной бригады – один человек, как правило, не обладает знаниями, необходимыми для реализации многоаспектного подхода к проектированию GUI-интерфейса. Надлежащее проектирование GUI-интерфейса требует объединения навыков художника-графика, специалиста по анализу требований, системного проектировщика, программиста, эксперта по технологии, специалиста в области социальной психологии, а также, возможно, некоторых других специалистов, в зависимости от характера системы.

В современном мире миллиарды вычислительных устройств. Еще больше программ для них. И у каждой свой интерфейс, являющийся «рычагами» взаимодействия между пользователем и машинным кодом. Не удивительно, что чем лучше интерфейс, тем эффективнее взаимодействие.

Однако далеко не все разработчики и даже дизайнеры, задумываются о создании удобного и понятного графического интерфейса пользователя.

Какие элементы интерфейса (ЭИ) создавать?

1 Разработка интерфейса обычно начинается с определения задачи или набора задач, для которых продукт предназначен.

2 Простое должно оставаться простым. Не стоит усложнять интерфейсы. Нужно постоянно думать о том, как сделать интерфейс проще и понятнее.

3 Пользователи не задумываются над тем, как устроена программа. Все, что они видят – это интерфейс. Поэтому, с точки зрения потребителя именно интерфейс является конечным продуктом.

4 Интерфейс должен быть ориентированным на человека, т. е. отвечать нуждам человека и учитывать его слабости. Нужно постоянно думать о том, с какими трудностями может столкнуться пользователь.

5 Необходимо думать о поведении и привычках пользователей. Не менять хорошо известные всем ЭИ на неожиданные, а новые делать интуитивно понятными.

6 Разрабатывать интерфейс необходимо исходя из принципа наименьшего возможного количества действий со стороны пользователя.

Какой должен быть дизайн элементов интерфейса?

В дизайне ЭИ нужно учитывать все: начиная от цвета, формы, пропорций, заканчивая когнитивной психологией. Однако несколько принципов все же стоит отметить:

– цвет. Цвета делятся на теплые (желтый, оранжевый, красный), холодные (синий, зеленый), нейтральные (серый). Обычно для ЭИ используют теплые цвета. Это как раз связано с психологией восприятия. Стоит отметить,

что мнение о цвете – очень субъективно и может меняться даже от настроения пользователя;

– форма. В большинстве случаев – прямоугольник со скругленными углами. Или круг. Опять же, форма как и цвет достаточно субъективна;

– основные ЭИ (часто используемые) должны быть выделены. Например, размером или цветом;

– иконки в программе должны быть очевидными. Или подписанными. Ведь, по сути дела, вместо того чтобы объяснять, пиктограммы зачастую сами требуют для себя объяснений.

Как правильно расположить элементы интерфейса на экране?

1 Есть утверждение, что визуальная привлекательность основана на пропорциях. Помните известное число 1.62? Это так называемый принцип Золотого сечения. Суть в том, что весь отрезок относится к большей его части так, как большая часть, относится к меньшей. Например, общая ширина сайта 900px, делим 900 на 1.62, получаем ~555px, это ширина блока с контентом. Теперь от 900 отнимаем 555 и получаем 345px. Это ширина меньшей части.

2 Перед расположением ЭИ следует упорядочить (сгруппировать) по значимости. То есть определить, какие наиболее важны, а какие – менее.

3 Обычно (но не обязательно) элементы размещаются в следующей градации: слева направо, сверху вниз. Слева вверху самые значимые элементы, справа внизу – менее. Это связано с порядком чтения текста. В случае с сенсорными экранами, самые важные элементы располагаются в области действия больших пальцев рук.

4 Необходимо учитывать привычки пользователя. Например, если в Windows кнопка закрыть находится в правом верхнем углу, то программе аналогичную кнопку необходимо расположить там же. То есть интерфейс должен иметь как можно больше аналогий, с известными пользователю вещами.

5 Размещать ЭИ стоит поближе там, где большую часть времени находится курсор пользователя, чтобы ему не пришлось перемещать курсор, например, от одного конца экрана к другому.

6 Элемент интерфейса можно считать видимым, если он либо в данный момент доступен для органов восприятия человека, либо он был настолько недавно воспринят, что еще не успел выйти из кратковременной памяти. Для нормальной работы интерфейса должны быть видимы только необходимые вещи – те, что идентифицируют части работающих систем, и те, что отображают способ, которым пользователь может взаимодействовать с устройством.

7 Отступы между ЭИ лучше делать равными или кратными друг другу.

Как элементы интерфейса должны себя вести?

1 Пользователи привыкают. Например, при удалении файла, появляется окно с подтверждением: «Да» или «Нет». Со временем пользователь перестает читать предупреждение и по привычке нажимает «Да». Поэтому диалоговое окно, которое было призвано обеспечить безопасность, абсолютно не выполняет

своей роли. Следовательно, необходимо дать пользователю возможность отменять сделанные им действия.

2 Если пользователю дают информацию, которую он должен куда-то ввести или как-то обработать, то информация должна оставаться на экране до того момента, пока человек ее не обработает. Иначе он может просто забыть.

3 Нужно избегать двусмысленности. Например, на фонарике есть одна кнопка. При нажатии фонарик включается, нажали еще раз – выключается. Если в фонарике перегорела лампочка, то при нажатии на кнопку не понятно, включаем мы его или нет. Поэтому, вместо одной кнопки выключателя лучше использовать переключатель (например, checkbox с двумя позициями: «вкл.» и «выкл.»). За исключением случаев, когда состояние задачи очевидно.

4 Имеет смысл делать монотонные интерфейсы. Монотонный интерфейс – это интерфейс, в котором какое-то действие можно сделать только одним способом. Такой подход обеспечит быструю привыкаемость к программе и автоматизацию действий.

5 Не стоит делать адаптивные интерфейсы, которые изменяются со временем, т. к. для выполнения какой-то задачи, лучше изучать только один интерфейс, а не несколько. Пример – стартовая страница браузера Chrome.

6 Если задержки в процессе выполнения программы неизбежны или действие, производимое пользователем, очень значимо, важно, чтобы в интерфейсе была предусмотрена сообщающая о них обратная связь. Например, можно использовать индикатор хода выполнения задачи (status bar).

7 ЭИ должны отвечать. Если пользователь произвел клик, то ЭИ должен как-то отозваться, чтобы человек понял, что клик произошел.

Задания для самостоятельной работы.

1 Создайте карту навигации для выбранной системы. На карте в зависимости от специфики системы выделите разделы, доступные различным пользователям в зависимости от роли, опишите условия перехода из различных разделов (при необходимости).

2 Используя графический редактор на выбор, создайте макеты графического интерфейса пользователя (от каждого члена бригады – не менее трех макетов). Предлагаемые системы:

– Microsoft Visio 2010;

– Axure;

– Adobe Photoshop;

– Balsamiq;

– Cacoo.

3 Для разработанных макетов подготовьте их текстовое описание в следующем виде (таблица 2).

Таблица 2 – Текстовое описание разработанных макетов

Название поля	Тип	Условие видимости	Условие доступности	Описание
Логотип	Ссылка	Виден всем	Доступен всем	Ссылка на сайт

Контрольные вопросы

1 Что такое графический интерфейс пользователя?
2 Какие бывают виды графического интерфейса?
3 Что такое карта навигации?

Список литературы

1 **Арлоу, Д.** UML и унифицированный процесс. Практический объектно-ориентированный анализ и проектирование / Д. Арлоу, А. Нейштадт. – Москва: Символ, 2016. – 624 с.

2 **Боггс, М.** UML и Rational Rose / М. Боггс. – Москва : РГГУ, 2010. – 9 с.

3 **Гуриков, С. Р.** Информатика: учебник / С. Р. Гуриков. – Москва: ФОРУМ; ИНФРА-М, 2018. – 463 с.

4 **Дадян, Э. Г.** Методы, модели, средства хранения и обработки данных: учебник / Э. Г Дадян, Ю. А. Зеленков. – Москва: Вузовский учебник; ИНФРА-М, 2017. – 168 с.

5 **Отвагин, А. В.** Платформа Microsoft.NET: лабораторный практикум по дисциплине «Объектно-ориентированное проектирование и программирование» для студентов специальности 1-40 02 01 «Вычислительные машины, системы и сети» всех форм обучения / А. В. Отвагин, Н. А. Павлёнок. – Минск: БГУИР, 2011. – 46 с.

6 **Репин, В.** Моделирование бизнес-процессов в нотации BPMN в Business Studio 5. Практическое руководство / В. Репин. – Москва : Ridero, 2022. – 5 с.