

Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Саратовский государственный технический университет  
имени Гагарина Ю.А.»

Филиал федерального государственного бюджетного образовательного  
учреждения высшего образования  
«Саратовский государственный технический университет  
имени Гагарина Ю.А.» в г. Петровске



УТВЕРЖДАЮ  
Директор филиала СГТУ  
имени Гагарина Ю.А. в г.Петровске  
Е.А.Бесшапошникова  
«30» июня 2025 г.

## МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ ПРАКТИЧЕСКИХ РАБОТ

по дисциплине  
МДК.02.02 «Инструментальные средства разработки программного  
обеспечения»

направление подготовки  
09.02.07 «Информационные системы и программирование»

Методические указания рассмотрены  
на заседании предметной (цикловой) комиссии  
общепрофессиональных дисциплин и  
профессиональных модулей  
«16» июня 2025 года, протокол №13

Председатель ПЦК  /Ю.А.Табарова/

Петровск 2025

### **Пояснительная записка**

Методические указания по выполнению практических работ подготовлены на основе рабочей программы учебной дисциплины МДК.02.02 «Инструментальные средства разработки программного обеспечения», разработанной на основе ФГОС СПО по специальности 09.02.07 «Информационные системы и программирование» и соответствующих общих (ОК) и профессиональных (ПК) компетенций:

ОК 01. Выбирать способы решения задач профессиональной деятельности применительно к различным контекстам.

ОК 02. Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности

ОК 04. Эффективно взаимодействовать и работать в коллективе и команде.

ОК 05. Осуществлять устную и письменную коммуникацию на государственном языке Российской Федерации с учетом особенностей социального и культурного контекста

ОК 09. Пользоваться профессиональной документацией на государственном и иностранном языках.

ПК 2.1. Разрабатывать требования к программным модулям на основе анализа проектной и технической документации на предмет взаимодействия компонент.

ПК 2.2. Выполнять интеграцию модулей в программное обеспечение

ПК 2.3. Выполнять отладку программного модуля с использованием специализированных программных средств

ПК 2.4. Осуществлять разработку тестовых наборов и тестовых сценариев для программного обеспечения

ПК 2.5. Производить инспектирование компонент программного обеспечения на предмет соответствия стандартам кодирования

При выполнении практических работ студент должен *знать*:

- модели процесса разработки программного обеспечения;
- основные принципы процесса разработки программного обеспечения;
- основные подходы к интегрированию программных модулей;
- виды и варианты интеграционных решений;
- современные технологии и инструменты интеграции;
- основные протоколы доступа к данным;
- методы и способы идентификации сбоев и ошибок при интеграции приложений;
- методы отладочных классов;
- стандарты качества программной документации;
- основы организации инспектирования и верификации;
- встроенные и основные специализированные инструменты анализа качества программных продуктов;

- графические средства проектирования архитектуры программных продуктов;
- методы организации работы в команде разработчиков;
- основы верификации и аттестации программного обеспечения;
- основные методы отладки;
- методы и схемы обработки исключительных ситуаций;
- основные методы и виды тестирования программных продуктов;
- приемы работы с инструментальными средствами тестирования и отладки;

- методы организации работы в команде разработчиков

При выполнении практических работ студент должен *уметь*:

- анализировать проектную и техническую документацию;
- использовать специализированные графические средства построения и анализа архитектуры программных продуктов;
- организовывать заданную интеграцию модулей в программные средства на базе имеющейся архитектуры и автоматизации бизнес-процессов;
- определять источники и приемники данных;
- проводить сравнительный анализ; выполнять отладку, используя методы и инструменты условной компиляции;
- оценивать размер минимального набора тестов;
- разрабатывать тестовые пакеты и тестовые сценарии;
- выявлять ошибки в системных компонентах на основе спецификаций;
- использовать выбранную систему контроля версий;
- использовать методы для получения кода с заданной функциональностью и степенью качества;
- организовывать заданную интеграцию модулей в программные средства на базе имеющейся архитектуры и автоматизации бизнес-процессов;
- использовать различные транспортные протоколы и стандарты форматирования сообщений;
- выполнять тестирование интеграции;
- организовывать постобработку данных;
- создавать классы-исключения на основе базовых классов;
- выполнять ручное и автоматизированное тестирование программного модуля;
- использовать приемы работы в системах контроля версий;
- анализировать проектную и техническую документацию;
- использовать инструментальные средства отладки программных продуктов;
- выполнять отладку, используя методы и инструменты условной компиляции.

Содержание практических занятий определено рабочей программой и тематическим планированием, соответствует теоретическому материалу изучаемых разделов учебной дисциплины.

Объем практических занятий по дисциплине определяется учебным планом по данной специальности.

Продолжительность практического занятия – 2 академических часа. Перед проведением практического занятия преподавателем организуется инструктаж, а по ее окончании – обсуждение итогов.

Комплект методических указаний по выполнению практических работ по дисциплине МДК.02.02 «Инструментальные средства разработки программного обеспечения» содержит 42 практических занятия.

**Перечень практических работ  
по дисциплине МДК.02.02 «Инструментальные средства разработки  
программного обеспечения»**

**ПРАКТИЧЕСКАЯ РАБОТА № 1**

Тема: Разработка структуры проекта

**ПРАКТИЧЕСКАЯ РАБОТА № 2**

Тема: Разработка структуры проекта

**ПРАКТИЧЕСКАЯ РАБОТА № 3**

Тема: Разработка структуры проекта

**ПРАКТИЧЕСКАЯ РАБОТА № 4**

Тема: Разработка структуры проекта

**ПРАКТИЧЕСКАЯ РАБОТА № 5**

Тема: Разработка модульной структуры проекта (диаграммы модулей)

**ПРАКТИЧЕСКАЯ РАБОТА № 6**

Тема: Разработка модульной структуры проекта (диаграммы модулей)

**ПРАКТИЧЕСКАЯ РАБОТА № 7**

Тема: Разработка модульной структуры проекта (диаграммы модулей)

**ПРАКТИЧЕСКАЯ РАБОТА № 8**

Тема: Разработка модульной структуры проекта (диаграммы модулей)

**ПРАКТИЧЕСКАЯ РАБОТА № 9**

Тема: Разработка перечня артефактов и протоколов проекта

**ПРАКТИЧЕСКАЯ РАБОТА № 10**

Тема: Разработка перечня артефактов и протоколов проекта

**ПРАКТИЧЕСКАЯ РАБОТА № 11**

Тема: Разработка перечня артефактов и протоколов проекта

**ПРАКТИЧЕСКАЯ РАБОТА № 12**

Тема: Разработка перечня артефактов и протоколов проекта

**ПРАКТИЧЕСКАЯ РАБОТА № 13**

Тема: Настройка работы системы контроля версий (типов импортируемых файлов, путей, фильтров и других параметров импорта в репозиторий)

**ПРАКТИЧЕСКАЯ РАБОТА № 14**

Тема: Разработка и интеграция модулей проекта (командная работа)

**ПРАКТИЧЕСКАЯ РАБОТА № 15**

Тема: Разработка и интеграция модулей проекта (командная работа)

**ПРАКТИЧЕСКАЯ РАБОТА № 16**

Тема: Разработка и интеграция модулей проекта (командная работа)

**ПРАКТИЧЕСКАЯ РАБОТА № 17**

Тема: Разработка и интеграция модулей проекта (командная работа)

**ПРАКТИЧЕСКАЯ РАБОТА № 18**

Тема: Разработка и интеграция модулей проекта (командная работа)

**ПРАКТИЧЕСКАЯ РАБОТА № 19**

Тема: Отладка отдельных модулей программного проекта. Организация обработки исключений.

**ПРАКТИЧЕСКАЯ РАБОТА № 20**

Тема: Отладка отдельных модулей программного проекта. Организация обработки исключений.

#### **ПРАКТИЧЕСКАЯ РАБОТА № 21**

Тема: Отладка отдельных модулей программного проекта. Организация обработки исключений.

#### **ПРАКТИЧЕСКАЯ РАБОТА № 22**

Тема: Отладка проекта

#### **ПРАКТИЧЕСКАЯ РАБОТА № 23**

Тема: Отладка проекта

#### **ПРАКТИЧЕСКАЯ РАБОТА № 24**

Тема: Отладка проекта

#### **ПРАКТИЧЕСКАЯ РАБОТА № 25**

Тема: Инспекция кода модулей проекта

#### **ПРАКТИЧЕСКАЯ РАБОТА № 26**

Тема: Инспекция кода модулей проекта

#### **ПРАКТИЧЕСКАЯ РАБОТА № 27**

Тема: Инспекция кода модулей проекта

#### **ПРАКТИЧЕСКАЯ РАБОТА № 28**

Тема: Тестирование интерфейса пользователя средствами инструментальной среды разработки

#### **ПРАКТИЧЕСКАЯ РАБОТА № 29**

Тема: Тестирование интерфейса пользователя средствами инструментальной среды разработки

#### **ПРАКТИЧЕСКАЯ РАБОТА № 30**

Тема: Тестирование интерфейса пользователя средствами инструментальной среды разработки

#### **ПРАКТИЧЕСКАЯ РАБОТА № 31**

Тема: Разработка тестовых модулей проекта для тестирования отдельных модулей

#### **ПРАКТИЧЕСКАЯ РАБОТА № 32**

Тема: Разработка тестовых модулей проекта для тестирования отдельных модулей

#### **ПРАКТИЧЕСКАЯ РАБОТА № 33**

Тема: Разработка тестовых модулей проекта для тестирования отдельных модулей

#### **ПРАКТИЧЕСКАЯ РАБОТА № 34**

Тема: Выполнение функционального тестирования

#### **ПРАКТИЧЕСКАЯ РАБОТА № 35**

Тема: Выполнение функционального тестирования

#### **ПРАКТИЧЕСКАЯ РАБОТА № 36**

Тема: Выполнение функционального тестирования

#### **ПРАКТИЧЕСКАЯ РАБОТА № 37**

Тема: Тестирование интеграции

#### **ПРАКТИЧЕСКАЯ РАБОТА № 38**

Тема: Тестирование интеграции

**ПРАКТИЧЕСКАЯ РАБОТА № 39**

Тема: Тестирование интеграции

**ПРАКТИЧЕСКАЯ РАБОТА № 40**

Тема: Документирование результатов тестирования

**ПРАКТИЧЕСКАЯ РАБОТА № 41**

Тема: Документирование результатов тестирования

**ПРАКТИЧЕСКАЯ РАБОТА № 42**

Тема: Документирование результатов тестирования

## **ИНСТРУКЦИИ ДЛЯ ОБУЧАЮЩИХСЯ ПО ВЫПОЛНЕНИЮ ПРАКТИЧЕСКИХ РАБОТ**

Прежде чем приступить к выполнению заданий, внимательно прочитайте данные рекомендации. Практические работы включают в себя задания следующих видов.

В ходе выполнения практических работ студент должен:

- выполнять требования по охране труда
- соблюдать инструкцию по правилам и мерам безопасности в кабинете информационных технологий
- строго выполнять весь объем работы, указанный в задании
- соблюдать требования эксплуатации компьютерной техники (правила включения и выключения)
- предоставить отчет о проделанной работе по окончании выполненной работы, который должен содержать:

1. Название работы.
2. Цель работы.
3. Задание и его решение.
4. Вывод о проделанной работе.

Текст отчета по практической работе должен быть набран на компьютере шрифтом Times New Roman размером 14 пт. (при оформлении текста используется текстовый редактор MS Word). Шрифт, используемый в иллюстративном материале (таблицы и рисунки), рекомендуется уменьшить до 12 пт. Межстрочный интервал в основном тексте - полуторный. В иллюстративном материале межстрочный интервал рекомендуется сделать одинарным. Поля страницы должны быть: левое поле - 30 мм; правое поле – 15 мм; верхнее и нижнее поле - 20 мм.

Каждый абзац должен начинаться с красной строки. Отступ абзаца – 1,25 см от левой границы текста.

Студент должен выполнить практическую работу самостоятельно (или в группе, если это предусмотрено заданием). Практическая работа выполняется согласно заданию и методическим рекомендациям. После выполнения практической работы обучающийся самостоятельно себя контролирует путем ответов на вопросы. Результат работы представляется преподавателю в виде файла (файлов) в личном каталоге, защищается обучающимися.

По ходу выполнения работы при возникновении вопросов обучающийся может получить консультацию у преподавателя или самостоятельно воспользоваться лекционным материалом, рекомендуемой литературой.

### **1. Оформление текстовых документов**

Текст работы печатается на одной стороне листа формата А4, должен быть только чёрного цвета и иметь поля (верхнее, нижнее – 2 см, левое – 3 см, правое – 1,5 см). Шрифт Times New Roman размером 14, межстрочный интервал 1,5, абзацный отступ 1,25.



## Правила оформления таблиц, рисунков, графиков

Все таблицы и рисунки должны иметь названия и порядковую нумерацию (например, Таблица 1, Рисунок 3). Нумерация таблиц и рисунков должна быть сквозной для всего текста до приложений. Таблицы, рисунки каждого приложения обозначают отдельной нумерацией арабскими цифрами с добавлением перед цифрой обозначения приложения (напр., Таблица В.1).

### Оформление таблицы.

Название таблицы помещается слева над таблицей без абзацного отступа, в одной строке с ее номером через тире (14 шрифтом).

В каждой таблице следует указывать единицы измерения показателей. Если единица измерения в таблице является общей для всех числовых табличных данных, то ее приводят в заголовке таблицы после ее названия.

При переносе: слово “Таблица” указывают один раз слева над первой частью таблицы, над другими частями пишут слова “Продолжение таблицы” или “Окончание таблицы” справа, с указанием номера (обозначения) таблицы. Если в конце страницы таблица прерывается и ее продолжение будет на следующей странице, то в первой части таблицы нижнюю горизонтальную черту, ограничивающую таблицу, не проводят.

Таблица 1 – Распределение ответов респондентов на вопросы анкеты по возрастным группам (в процентах)

Варианты ответов	Возрастные группы				Всего по выборке
	18-24 года	25-29 лет	30-45 лет	старше 45 лет	

Не допускается прямое копирование в текст Диплома выходных таблиц отчета компьютерной программы STATISTICA. Таблицы должны быть построены заново.

### Оформление рисунка.

Все иллюстративные материалы (рисунки, диаграммы, графики) в Дипломе имеют название «Рисунок». На графический материал должна быть дана ссылка в тексте документа.

Иллюстрации могут быть в компьютерном исполнении, в том числе и цветные.

Порядковый номер рисунка и – через тире – его название проставляются под рисунком по центру строки (см. Рисунок 1).



Рисунок 1 – Пятиконечная звезда

## 2. Составление программы на языке программирования

Правила оформления кода:

### 1. Используйте разумные имена для переменных и функций

Программа должна быть хорошо понятна человеку при чтении. Если при чтении программы приходится понимать назначение переменных и функций по тому, как они используются, то читать код становится гораздо

сложнее. Неудачно выбранные имена могут привести к тому, что смысл программы может быть неправильно понят. Выбирайте такие имена, которые бы объясняли смысл переменных и функций, тогда код станет гораздо понятнее, и не потребуется писать множество комментариев.

При написании составных слов, например в именах переменных, пишите их слитно без пробелов, при этом каждое новое слово пишется с большой буквы.

## 2. Не дублируйте код

Если в программе есть одинаковые выражения или фрагмента кода, вынесите этот код в отдельную функцию. Верным признаком необходимости создания новой функции является желание скопировать фрагмент кода из одного места программы в другое. В таком случае сразу перенесите этот фрагмент в отдельную функцию.

Если фрагменты кода похожи, но не идентичны, подумайте, не получится ли и их вынести в одну функцию, возможно добавив параметры или условия.

## 3. Не используйте «магические константы»

Использование неименованных «магических» констант в коде нежелательно:

- при чтении кода может быть не понятно, что это за число, и почему оно именно такое;
- чаще всего одно и то же число потребуется написать в нескольких местах кода. Если его придётся изменять, можно пропустить одно из использований, что приведёт к ошибке.

Если в коде нужно использовать константу, дайте ей имя, используя `const`.

## 4. Расставляйте пробелы вокруг бинарных операторов. Это улучшает читаемость формул.

5. Всегда выделяйте блоки условных операторов и циклов скобками. В любой блок условия или цикла может захотеться добавить новое выражение. При этом можно забыть добавить скобки. Лучше сразу добавить скобки, чтобы потом не было с этим проблем.

6. Расставляйте скобки одинаково. Выберите и используйте для себя один из стилей расстановки скобок. Это улучшает читаемость структуры программы.

7. Не делайте строки слишком длинными. Строка программы должна помещаться на экране. Обычно рекомендуют ограничить максимальную ширину строки в 80 символов. Если определение или вызов функции получается слишком широким поместите по одному параметру на каждой строке. Длинные математические выражения разбивайте на несколько строк, разбивая по границам логических блоков выражения.

8. Объявляйте переменные непосредственно перед использованием. Обязательно указывайте начальное значение для переменных. Значение неинициализированной переменной может быть любым. Использование (чтение) такого значения приведёт к недетерминированной работе программы, а в некоторых случаях является неопределённым поведением.

Чтобы избежать проблем всегда инициализируйте переменные прямо в момент их создания.

9.Единый стиль оформления кода во всем проекте;

10. Визуальное выделение наиболее значимых частей — используя *вертикальное форматирование*, мы выделяем объявление переменных, цикл заполнения массива случайными числами и цикл обработки по формуле. Если ваша функция выполняет несколько действий — то разумно разделить соответствующие блоки кода пустыми строками.

### **3.Ответ на поставленные вопросы (с аргументацией)**

Прочитайте вопрос и вникните в него.

Для удобства подчеркните ту, фразу, которая, по вашему мнению, является главной. Это поможет вам быстрее сориентироваться при ответе на вопрос.

Если вы считаете, что можете ответить на вопрос без помощи лекции и дополнительной литературы — приступайте. Если же вопрос заставляет вас сомневаться, откройте лекционную тетрадь (учебник или дополнительную литературу), прочитайте необходимый пункт, вникните в содержание и после этого приступайте за работу.

**ГЛАВНОЕ!** Не переписывайте отрывки лекции в рабочую тетрадь! Четко отвечайте на ПОСТАВЛЕННЫЙ вопрос!

Не забудьте привести аргументацию (обоснование) вашей позиции, если вопрос предполагает личностное отношение к проблеме.

## **ПРАКТИЧЕСКАЯ РАБОТА № 1**

### **Тема: Разработка структуры проекта**

**Цель работы:** формирование навыков постановки задачи и разработки технического задания на программный продукт.

**Оборудование:** ПК, программное обеспечение – Visual Studio, MS Word, инструкции по выполнению работы.

#### **Содержание работы:**

##### **Задание:**

1. Выбрать вариант задания на проектирование и разработку учебной программы.
2. В соответствии с вариантом выполнить разработку технического задания, которое должно включать:
  - введение;
  - основание для разработки;
  - назначение;
  - требования к программе и программному продукту;
  - требования к программной документации.
3. Оформить отчет. Содержание отчета:
  - тема практической работы
  - цель практической работы
  - ответы на контрольные вопросы
  - задание на практическую работу
  - разработанное техническое задание
  - выводы по проделанной работе.

##### Варианты задания:

1. Ввести вещественную матрицу размерности  $n * m$  построчно, а вывести по столбцам.
2. Выяснить сколько положительных элементов содержит матрица размерности  $n * m$ , если  $a_{ij} = \sin(i + j/2)$ .
3. Дана квадратная вещественная матрица размерности  $n$ . Является ли матрица симметричной относительно главной диагонали.
4. Дана квадратная вещественная матрица размерности  $n$ . Транспонировать матрицу.
5. Дана квадратная вещественная матрица размерности  $n$ . Сравнить сумму элементов матрицы на главной и побочной диагоналях.
6. Дана квадратная вещественная матрица размерности  $n$ . Найти количество нулевых элементов, стоящих:
  - выше главной диагонали;
  - ниже главной диагонали;
  - выше и ниже побочной.
7. Дана вещественная матрица размерности  $n * m$ . По матрице получить логический вектор, присвоив его  $k$ -ому элементу значение True, если выполнено указанное условие и значение False иначе:
  - все элементы  $k$  столбца нулевые;

элементы  $k$  строки матрицы упорядочены по убыванию;  
 $k$  строка массива симметрична.

8. Дана вещественная матрица размерности  $n * m$ . Сформировать вектор  $b$ , в котором элементы вычисляются как:

произведение элементов соответствующих строк;  
среднее арифметическое соответствующих столбцов;  
разность наибольших и наименьших элементов соответствующих строк;  
значения первых отрицательных элементов в столбце.

9. Дана вещественная матрица размерности  $n * m$ . Вывести номера столбцов, содержащих только отрицательные элементы.

10. Дана вещественная матрица размерности  $n * m$ . Вывести номера строк, содержащих больше положительных элементов, чем отрицательных.

### **Контрольные вопросы**

1. Перечислите этапы разработки программных продуктов.
2. Для чего необходимо техническое задание?
3. Кто занимается разработкой технического задания?
4. Какие пункты включает техническое задание?

## **ПРАКТИЧЕСКАЯ РАБОТА № 2**

### **Тема: Разработка структуры проекта**

**Цель работы:** формирование навыков постановки задачи и разработки технического задания на программный продукт.

**Оборудование:** ПК, программное обеспечение – Visual Studio, MS Word, инструкции по выполнению работы.

#### **Содержание работы:**

##### **Задание:**

1. Выбрать вариант задания на проектирование и разработку учебной программы.
2. В соответствии с вариантом выполнить разработку технического задания, которое должно включать:
  - введение;
  - основание для разработки;
  - назначение;
  - требования к программе и программному продукту;
  - требования к программной документации.
3. Оформить отчет. Содержание отчета:
  - тема практической работы
  - цель практической работы
  - задание на практическую работу
  - разработанное техническое задание
  - выводы по проделанной работе.

##### **Варианты задания:**

1. Дана вещественная матрица размерности  $n * m$ . Найти общую сумму элементов только тех столбцов, которые имеют хотя бы один нулевой элемент.
2. Дана вещественная матрица размерности  $n * m$ . Поменять местами строки с максимальным и минимальным элементами.
3. Дана вещественная матрица размерности  $n * m$ . Удалить  $k$  столбец матрицы.
4. Дана вещественная квадратная матрица размерности  $n$ . Поменять местами элементы главной и побочной диагоналей матрицы:
  - по строкам;
  - по столбцам.
5. Дана вещественная матрица размерности  $m * n$ . Упорядочить элементы каждой четной строки по возрастанию.
6. Дана вещественная матрица размерности  $m * n$ . Расположить все элементы матрицы по убыванию. Обход матрицы осуществлять по строкам.
7. Дана вещественная матрица размерности  $m * n$ . Определить индексы первого нулевого элемента матрицы. Обход матрицы осуществлять по столбцам.
8. Известно положение двух ферзей на шахматной доске. Бьют ли они друг друга?

9. Напишите программу, отображающую окно, в котором внутренняя часть закрашена желтым цветом. При наведении курсора на область окна цвет фона меняется на зеленый. При щелчке мышью размеры окна должны увеличиваться на 10%

10. Напишите программу, в которой отображается окно с текстовой меткой и тремя кнопками. В текстовой метке содержится число (начальное значение — нулевое). Щелчок по одной из меток приводит к увеличению значения числа на единицу. Щелчок по другой кнопке приводит к уменьшению значения числа на единицу. Щелчок по третьей кнопке приводит к закрытию окна.

## **ПРАКТИЧЕСКАЯ РАБОТА № 3**

### **Тема: Разработка структуры проекта**

**Цель работы:** формирование навыков постановки задачи и разработки технического задания на программный продукт.

**Оборудование:** ПК, программное обеспечение – Visual Studio, MS Word, инструкции по выполнению работы.

#### **Содержание работы:**

##### **Задание:**

1. Выбрать вариант задания на проектирование и разработку учебной программы.
2. В соответствии с вариантом выполнить разработку технического задания, которое должно включать:
  - введение;
  - основание для разработки;
  - назначение;
  - требования к программе и программному продукту;
  - требования к программной документации.
3. Оформить отчет. Содержание отчета:
  - тема практической работы
  - цель практической работы
  - задание на практическую работу
  - разработанное техническое задание
  - выводы по проделанной работе.

##### **Варианты задания:**

1. Напишите программу, в которой открывается окно с раскрывающимся списком. Список содержит названия цветов (красный, желтый, зеленый и так далее). Также окно содержит область, закрашенную тем цветом, который выбран в списке. При выборе в списке нового цвета область закрашивается этим цветом автоматически.
2. Напишите программу, в которой открывается окно с полем ввода. При вводе текста в окно этот текст автоматически дублируется в текстовой метке. В окне должны быть две опции, которые позволяют применять к тексту в метке жирный и курсивный стили.
3. Напишите программу, в которой отображается окно с двумя текстовыми полями. Предполагается, что в эти текстовые поля вводятся целочисленные значения. Кроме полей, в окне размещена метка, в которой содержится информация о том, какое из двух чисел больше/меньше или что числа равны друг другу. Информация в метке обновляется автоматически при изменении содержимого полей. Если хотя бы в одном из полей указано не число, метка должна содержать информацию об этом.
4. Напишите программу, в которой отображается окно с изображением и двумя кнопками. Имеется несколько изображений, которые последовательно циклически отображаются в окне при щелчке по кнопкам. При щелчке по одной кнопке появляется следующее изображение в последовательности, а



при щелчке по другой кнопке отображается предыдущее изображение в последовательности.

5. Напишите программу, в которой отображается окно со списком выбора. В списке выбора представлены названия шрифтов. Также окно содержит раскрывающийся список с названиями цветов (красный, зеленый, синий и так далее). Окно содержит область с текстом. При выборе цвета или названия шрифта этот цвет или шрифт применяются для отображения текста.

6. Напишите программу, в которой отображается окно с закрашенной областью. Для этой области есть контекстное меню с названиями цветов (красный, желтый, зеленый и так далее). При выборе команды из контекстного меню область закрашивается соответствующим цветом.

7. Напишите программу, в которой отображается окно с главным меню и областью с текстом. Текст содержит информацию о названии, стиле и размере шрифта, которым отображается текст. В меню есть пункты для выбора названия шрифта, стиля шрифта и размера шрифта. При выборе команды из меню соответствующая характеристика применяется для отображения текста, а также с учетом новых параметров шрифта меняется сам текст.

## **ПРАКТИЧЕСКАЯ РАБОТА № 4**

### **Тема: Разработка структуры проекта**

**Цель работы:** формирование навыков постановки задачи и разработки технического задания на программный продукт.

**Оборудование:** ПК, программное обеспечение – Visual Studio, MS Word, инструкции по выполнению работы.

#### **Содержание работы:**

##### **Задание:**

1. Выбрать вариант задания на проектирование и разработку учебной программы.
2. В соответствии с вариантом выполнить разработку технического задания, которое должно включать:
  - введение;
  - основание для разработки;
  - назначение;
  - требования к программе и программному продукту;
  - требования к программной документации.
3. Оформить отчет. Содержание отчета:
  - тема практической работы
  - цель практической работы
  - задание на практическую работу
  - разработанное техническое задание
  - выводы по проделанной работе.

##### **Варианты задания:**

1. Создать структуру данных, которая хранит информацию о банковском счете – его номер, тип и баланс. Создать переменную такого типа, заполнить структуру значениями и напечатать результат
2. Создать перечислимый тип ВУЗ{КГУ, КАИ, КХТИ}. Создать структуру работник с двумя полями: имя, ВУЗ. Заполнить структуру данными и распечатать.
3. Написать программу, которая вычисляет число гласных и согласных букв в файле. Имя файла передавать как аргумент в функцию Main. Содержимое текстового файла заносится в массив символов. Количество гласных и согласных букв определяется проходом по массиву. Предусмотреть метод, входным параметром которого является массив символов. Метод вычисляет количество гласных и согласных букв.
4. Написать программу, вычисляющую среднюю температуру за год. Создать двумерный случайный массив temperature[12,30], в котором будет храниться температура для каждого дня месяца (предполагается, что в каждом месяце 30 дней). Сгенерировать значения температур случайным образом. Для каждого месяца распечатать среднюю температуру. Для этого написать метод, который по массиву temperature [12,30] для каждого месяца вычисляет среднюю температуру в нем, и в качестве результата возвращает массив

средних температур. Полученный массив средних температур отсортировать по возрастанию.

5. Реализовать класс для описания здания (уникальный номер здания, высота, этажность, количество квартир, подъездов). Поля сделать закрытыми, предусмотреть методы для заполнения полей и получения значений полей для печати. Добавить методы вычисления высоты этажа, количества квартир в подъезде, количества квартир на этаже и т.д. Предусмотреть возможность, чтобы уникальный номер здания генерировался программно. Для этого в классе предусмотреть статическое поле, которое бы хранило последний использованный номер здания, и предусмотреть метод, который увеличивал бы значение этого поля

6. Работа со строками. Дан текстовый файл, содержащий ФИО и e-mail адрес. Разделителем между ФИО и адресом электронной почты является символ #: Иванов Иван Иванович # iviviv@mail.ru Петров Петр Петрович # petr@mail.ru Сформировать новый файл, содержащий список адресов электронной почты. Предусмотреть метод, выделяющий из строки адрес почты. Методу в качестве параметра передается символьная строка s, e-mail возвращается в той же строке s: public void SearchMail (ref string s).

## ПРАКТИЧЕСКАЯ РАБОТА № 5

**Тема:** Разработка модульной структуры проекта (диаграммы модулей)

**Цель работы:** изучить процесс разработки модульной структуры программного обеспечения, осуществляемого с помощью структурных карт Консантайна, научиться разрабатывать модульную структуру проекта.

**Оборудование:** ПК, программное обеспечение – Visual Paradigm Online, MS Word, инструкции по выполнению работы.

### Справочный материал:

*Структурные карты Консантайна* — это модель отношений между модулями программы. Они используются при проектировании ПО и показывают, как продукт будет выполнять системные требования.

### Структурная карта состоит из разных элементов:

*Модули* — ключевой элемент карты. Бывают разных типов: детализированный модуль, совокупность подпрограмм в модуле (библиотека), модули с областями глобальных или распределённых данных.

*Вызов модуля.* Есть разные виды вызовов: последовательный, параллельный, вызов сопрограммы.

*Связь по данным.*

*Связь по управлению*

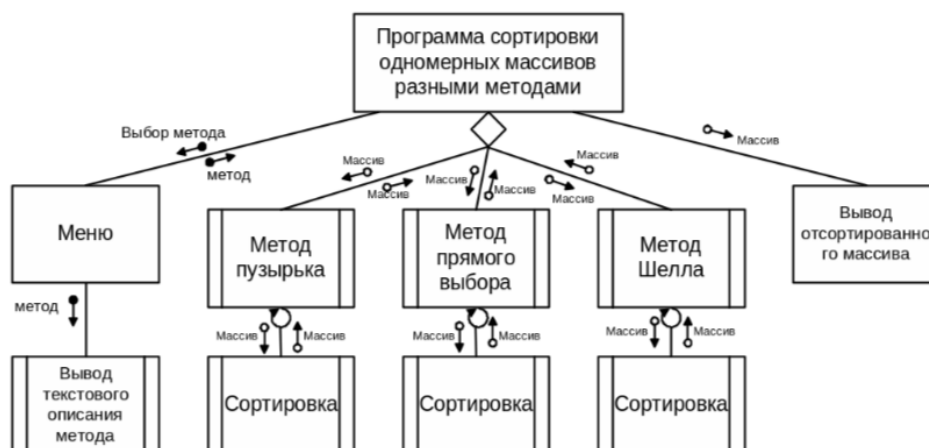
### Некоторые объекты структурной карты Консантайна:

1. Структурный блок — блок кодов с одним входом и выходом.
2. Процедурный блок — вызов процедуры, определённой ранее.
3. Библиотечный блок — вызов библиотечного модуля.

Чтобы увязывать блоки между собой, используют разные виды связей:

1. Последовательная связь — последовательное выполнение слева направо.
2. Параллельная связь — блоки выполняются одновременно.
3. Условная связь — можно выбрать одну из альтернатив.
4. Итерационная связь — блоки выполняются в цикле.

Пример структурной карты Консантайна



### Содержание работы:

**Задание 1.** Разработать модульную структуру программного модуля (по вариантам).

1. Запустите программу Visual Paradigm Online (<https://online.visual-paradigm.com/drive/#diagramlist:proj=0&dashboard> )

2. Выберите шаблоны группы Диаграммы: SDL Diagram, ArchiMate Tutorial Starter, Functional Decomposition Diagram
3. Добавьте необходимые формы



4. Составьте структурную карту Константайна
5. Составьте спецификацию программного модуля.

#### **Варианты заданий:**

1. Разработать программный модуль «Учет успеваемости студентов». Программный модуль предназначен для оперативного учета успеваемости студентов в сессию деканом, заместителями декана и сотрудниками деканата. Сведения об успеваемости студентов должны храниться в течение всего срока их обучения и использоваться при составлении справок о прослушанных курсах и приложений к диплому.
2. Разработать программный модуль «Личные дела студентов». Программный модуль предназначен для получения сведений о студентах сотрудниками деканата, профкома и отдела кадров. Сведения должны храниться в течение всего срока обучения студентов и использоваться при составлении справок и отчетов.
3. Разработать программный модуль «Решение комбинаторно-оптимизационных задач». Модуль должен содержать алгоритмы поиска цикла минимальной длины (задача коммивояжера), поиска кратчайшего пути и поиска минимального связывающего дерева.
4. Разработать приложение Windows «Органайзер». Приложение предназначено для записи, хранения и поиска адресов и телефонов физических лиц, и организаций, а также расписания, встреч и др. Приложение предназначено для любых пользователей компьютера.

Примечание. При разработке программы не ограничиваться функциями, приведенными в варианте, добавить несколько своих функций

## ПРАКТИЧЕСКАЯ РАБОТА № 6

**Тема:** Разработка модульной структуры проекта (диаграммы модулей)

**Цель работы:** изучить процесс разработки модульной структуры программного обеспечения, осуществляемого с помощью структурных карт Константайна, научиться разрабатывать модульную структуру проекта.

**Оборудование:** ПК, программное обеспечение – Visual Paradigm Online, MS Word, инструкции по выполнению работы.

**Содержание работы:**

**Задание 1.** Разработать модульную структуру программного модуля (по вариантам).

1. Запустите программу Visual Paradigm Online (<https://online.visual-paradigm.com/drive/#diagramlist:proj=0&dashboard> )
2. Выберите шаблоны группы Диаграммы: SDL Diagram, ArchiMate Tutorial Starter, Functional Decomposition Diagram
3. Добавьте необходимые формы



4. Составьте структурную карту Константайна
5. Составьте спецификацию программного модуля.

**Варианты заданий:**

1. Разработать приложение Windows «Калькулятор». Приложение предназначено для любых пользователей и должно содержать все арифметические операции (с соблюдением приоритетов) и желательно (но не обязательно) несколько математических функций.
2. Разработать программный модуль «Кафедра», содержащий сведения о сотрудниках кафедры (ФИО, должность, ученая степень, дисциплины, нагрузка, общественная работа, совместительство и др.). Модуль предназначен для использования сотрудниками отдела кадров и деканата.
3. Разработать программный модуль «Лаборатория», содержащий сведения о сотрудниках лаборатории (ФИО, пол, возраст, семейное положение, наличие детей, должность, ученая степень). Модуль предназначен для использования сотрудниками профкома и отдела кадров.
4. Разработать программный модуль «Автосервис». При записи на обслуживание заполняется заявка, в которой указываются ФИО владельца, марка автомобиля, вид работы, дата приема заказа и стоимость ремонта. После выполнения работ распечатывается квитанция.

Примечание. При разработке программы не ограничиваться функциями, приведенными в варианте, добавить несколько своих функций

## ПРАКТИЧЕСКАЯ РАБОТА № 7

**Тема:** Разработка модульной структуры проекта (диаграммы модулей)

**Цель работы:** изучить процесс разработки модульной структуры программного обеспечения, осуществляемого с помощью структурных карт Константайна, научиться разрабатывать модульную структуру проекта.

**Оборудование:** ПК, программное обеспечение – Visual Paradigm Online, MS Word, инструкции по выполнению работы.

**Содержание работы:**

**Задание 1.** Разработать модульную структуру программного модуля (по вариантам).

1. Запустите программу Visual Paradigm Online (<https://online.visual-paradigm.com/drive/#diagramlist:proj=0&dashboard> )
2. Выберите шаблоны группы Диаграммы: SDL Diagram, ArchiMate Tutorial Starter, Functional Decomposition Diagram
3. Добавьте необходимые формы



4. Составьте структурную карту Константайна
5. Составьте спецификацию программного модуля.

**Варианты заданий:**

1. Разработать программный модуль «Картотека абонентов АТС». Картотека содержит сведения о телефонах и их владельцах. Фиксирует задолженности по оплате (абонентской и повременной). Считается, что повременная оплата местных телефонных разговоров уже введена.
2. Разработать программный модуль «Авиакасса», содержащий сведения о наличии свободных мест на авиамаршруты. В базе должны содержаться сведения о номере рейса, экипаже, типе самолета, дате и времени вылета, а также стоимости авиабилетов (разного класса). При поступлении заявки на билеты программа производит поиск подходящего рейса.
3. Разработать программный модуль «Книжный магазин», содержащий сведения о книгах (автор, название, издательство, год издания, цена). Покупатель оформляет заявку на нужные ему книги, если таковых нет, он заносится в базу и оповещается, когда нужные книги поступают в магазин.
4. Разработать программный модуль «Автостоянка». В программе содержится информация о марке автомобиля, его владельце, дате и времени въезда, стоимости стоянки, скидках, задолженности по оплате и др.

**Примечание.** При разработке программы не ограничиваться функциями, приведенными в варианте, добавить несколько своих функций

## ПРАКТИЧЕСКАЯ РАБОТА № 8

**Тема:** Разработка модульной структуры проекта (диаграммы модулей)

**Цель работы:** изучить процесс разработки модульной структуры программного обеспечения, осуществляемого с помощью структурных карт Константайна, научиться разрабатывать модульную структуру проекта.

**Оборудование:** ПК, программное обеспечение – Visual Paradigm Online, MS Word, инструкции по выполнению работы.

**Содержание работы:**

**Задание 1.** Разработать модульную структуру программного модуля (по вариантам).

1. Запустите программу Visual Paradigm Online (<https://online.visual-paradigm.com/drive/#diagramlist:proj=0&dashboard> )
2. Выберите шаблоны группы Диаграммы: SDL Diagram, ArchiMate Tutorial Starter, Functional Decomposition Diagram
3. Добавьте необходимые формы



4. Составьте структурную карту Константайна
5. Составьте спецификацию программного модуля.

**Варианты заданий:**

1. Разработать программный модуль «Учет нарушений правил дорожного движения». Для каждой автомашины (и ее владельца) в базе хранится список нарушений. Для каждого нарушения фиксируется дата, время, вид нарушения и размер штрафа. При оплате всех штрафов машина удаляется из базы.
2. Разработать программный модуль «Кадровое агентство», содержащий сведения о вакансиях и резюме. Программный модуль предназначен как для поиска сотрудника, отвечающего требованиям руководителей фирмы, так и для поиска подходящей работы.
3. Разработать программный модуль «Картотека агентства недвижимости», предназначенный для использования работниками агентства. В базе содержатся сведения о квартирах (количество комнат, этаж, метраж и др.). При поступлении заявки на обмен (куплю, продажу) производится поиск подходящего варианта. Если такого нет, клиент заносится в клиентскую базу и оповещается, когда вариант появляется.

**Примечание.** При разработке программы не ограничиваться функциями, приведенными в варианте, добавить несколько своих функций



**Контрольные вопросы:**

1. Из чего состоит структурная схема?
2. Что такое спецификация программного модуля?
3. В чем заключается методика Константайна?

## ПРАКТИЧЕСКАЯ РАБОТА № 9

**Тема:** Разработка перечня артефактов и протоколов проекта

**Цель работы:** изучение процесса разработки необходимого перечня артефактов и протоколов создаваемого проекта.

**Оборудование:** ПК, программное обеспечение – Visual Paradigm Online, MS Word, инструкции по выполнению работы.

### **Справочный материал:**

Артефакт – это любой созданный искусственно элемент программной системы.

К элементам программной системы, а, следовательно, и к артефактам, могут относиться исполняемые файлы, исходные тексты, веб-страницы, справочные файлы, сопроводительные документы, файлы с данными, модели и многое другое, являющееся физическим носителем информации. Другими словами, артефактами являются те информационные элементы, которые тем или иным способом используются при работе программной системы и входят в ее состав.

*Артефакты* – это некоторые продукты проекта, порождаемые или используемые в нем при работе над окончательным продуктом.

Весь процесс разработки программной системы рассматривается в RUP как процесс создания артефактов. Причем то, что попадает в руки конечного пользователя, будь то программный модуль или программная документация, – это один из подклассов всех артефактов проекта.

Каждый член проектной группы создает свои артефакты и несет за них ответственность. Программист разрабатывает программу, руководитель – проектный план, а аналитик – модели системы. RUP позволяет определить, когда, кому и какой артефакт необходимо создать.

Система гиперссылок построена таким образом, что можно легко переходить от работ к артефактам, создаваемым в процессе конкретной деятельности, а через них к ролям исполнителей и обратно. К артефактам можно добраться различными путями – например, через список процессов, примеры итераций, роли исполнителей, а можно просто найти нужный артефакт в дереве ссылок, что позволяет рассматривать один и тот же процесс разработки с различных точек зрения – руководителя и исполнителя, пользователя и программиста.

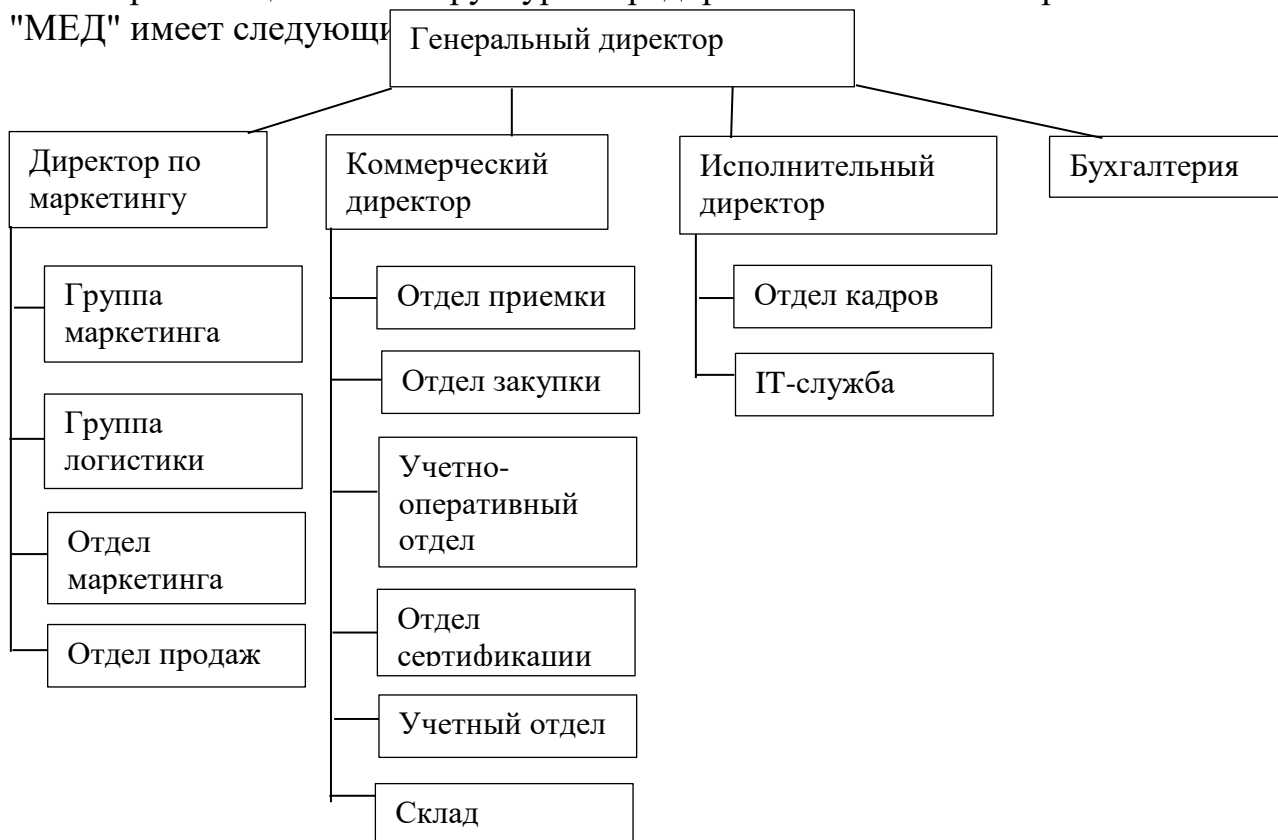
### **Содержание работы:**

**Задание 1.** Определите внешних исполнителей (контрагентов компании). Создайте «физическую» диаграмму. Постройте диаграмму прецедентов.

#### *1. Краткое описание предметной области*

Компания - дистрибьютор ЗАО "МЕД" закупает медицинские препараты отечественных и зарубежных производителей и реализует их через собственную дистрибьюторскую сеть и сеть аптек. Компания осуществляет доставку товаров, как собственным транспортом, так и с помощью услуг сторонних организаций.

Организационная структура предприятия оптовой торговли ЗАО "МЕД" имеет следующую



Основные цели автоматизации компании "МЕД":

- Разработка и внедрение комплексной автоматизированной системы поддержки логистических процессов компании.
- Повышение эффективности работы всех подразделений компании и обеспечение ведения учета в единой информационной системе.

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

Ключевые функциональные требования к информационной системе:

1. Управление запасами. Оперативное получение информации об остатках на складе.
2. Управление закупками. Планирование закупок в разрезе поставщиков.
3. Управление продажами. Контроль лимита задолженности с возможностью блокировки формирования отгрузочных документов.
4. Полный контроль взаиморасчетов с поставщиками и клиентами.
5. Получение управленческих отчетов в необходимых аналитических срезах – как детальных для менеджеров, так и агрегированных для руководителей подразделений, и директоров фирмы.

*Ограничения предметной области*

В рамках проекта не рассматривается автоматизация учета основных средств, расчета и начисления заработной платы, управления кадрами. Развертывание новой системы предполагается осуществить только в следующих подразделениях ЗАО "МЕД":

- Отдел закупок;
- Отдел приемки;

- Отдел продаж;
- Отдел маркетинга;
- Группа планирования и маркетинга;
- Группа логистики;
- Учетно-операционный отдел;
- Учетный отдел;
- Отдел сертификации (в части учета сертификатов на медикаменты);
- Бухгалтерия (только в части учета закупок, продаж, поступлений и платежей).

#### *Описание состава автоматизируемых бизнес-процессов*

1. Компания "МЕД» осуществляет закупки у отечественных и зарубежных производителей, следовательно, контрагентами компании являются отечественные и зарубежные поставщики медикаментов.
2. Компания пользуется услугами транспортных компаний для доставки медикаментов. Следовательно, внешними контрагентами также являются транспортные компании.
3. Компания реализует медикаменты через дистрибьюторскую сеть и сеть аптек. Следовательно, контрагентами компании являются покупатели (дистрибьюторы, аптеки).

**Внешними контрагентами** компании "МЕД" являются поставщики (отечественные, зарубежные), покупатели (дистрибьюторы, аптеки) и транспортные компании.

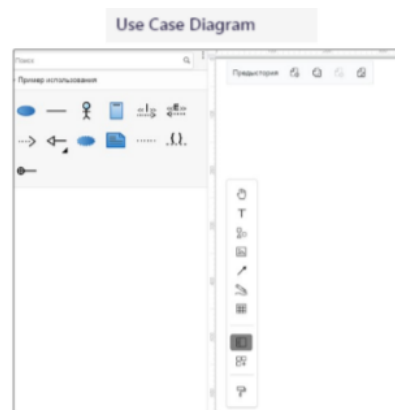
На физической диаграмме компанию изобразим прямоугольником. Для отображения контрагентов используются графический символ Actor (фигурка человечка). Для изображения взаимодействия между компанией и внешними контрагентами используются соединительные линии, поименованные для того, чтобы были понятны функции контрагентов по отношению к компании.

#### *2. Создание физической диаграммы*

2.1. Запустите Visual Paradigm Online (<https://online.visual-paradigm.com/drive/#diagramlist:proj=0&dashboard> )

2.2. Появится окно, в котором выберите (катеорию шаблонов) Вкладку Diagram

2.3. В открывшемся списке выберите диаграмму Use Case, активизируйте команду Create Blank.



2.4. Для изображения границ компании «МЕД» выберите из набора графических элементов, представленных в левой части окна, пиктограмму прямоугольника с надписью «Система» и переносите ее на рабочее поле мышкой при нажатой правой клавише, Отрегулируйте размеры прямоугольника.

2.5. Задайте цвет прямоугольника «Система» и введите подпись «ЗАО «МЕД»».

2.6. Для изображения на диаграмме контрагентов следует воспользоваться графическим символом с изображением человечка с надписью «Актер» и перенести его на рабочее поле при нажатой правой клавише мышки.

2.7. Соедините линиями изображение каждого контрагента с прямоугольником используя линии «Ассоциация/Сообщение» и при нажатой левой клавише мышки осуществите соединение фигур.

2.8. Внесите наименования контрагентов "Покупатели (аптеки)", "Покупатели (дистрибьюторы)", "Поставщики (Россия)", "Поставщики (импорт)", "Транспортные компании". Для того чтобы внести надписи на диаграмме, необходимо на панели инструментов "Форматирование" зафиксировать пиктограмму «Текст» (символ буквы "А"). Щелкните мышкой на изображении человечка, курсор установится на поле с надписью Актер. Введите в это поле наименование контрагента.

2.9. Введите наименование компании "МЕД" в нарисованный прямоугольник, щелкнув мышкой по прямоугольнику. Обратите внимание на то, что при этом должна быть активна пиктограмма «Текст» (символ буквы "А").

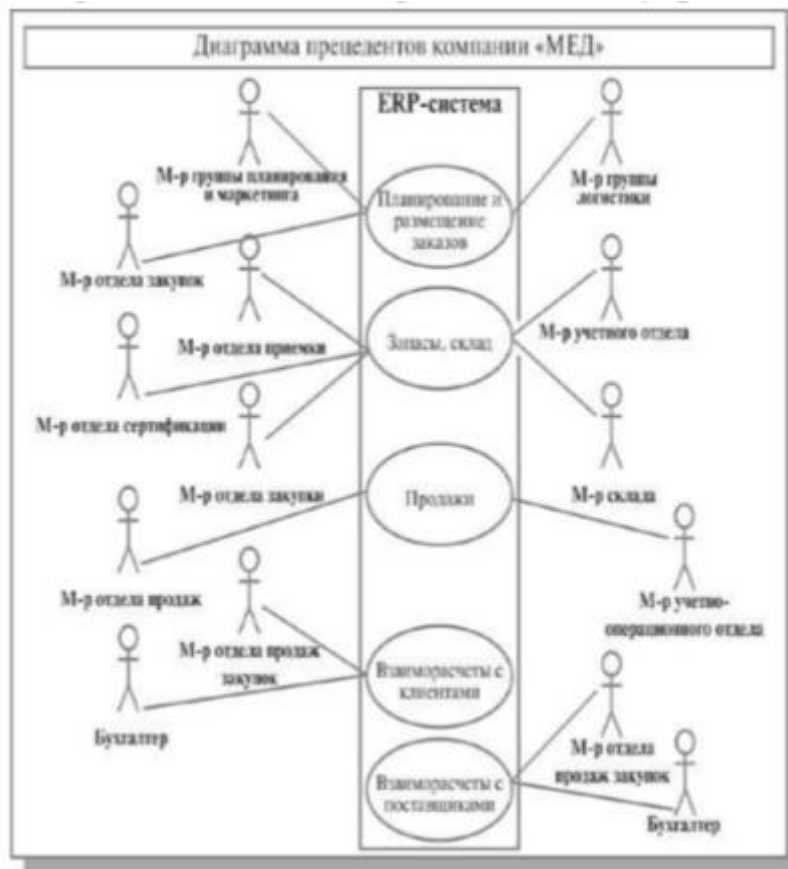
2.10. Аналогичным образом внесите надписи к линиям соединения фирмы и контрагентов.

Физическая диаграмма ЗАО "МЕД" представлена на рисунке ниже.



### 3. Построение диаграммы прецедентов

Используя навыки, полученные при выполнении задания 1, постройте диаграмму прецедентов, отображающую прецеденты (варианты использования) компании «Мед» и внутренних исполнителей, обеспечивающих реализацию этих прецедентов внутри системы.



**Задание 2.** Определите внешних исполнителей (контрагентов компании). Создайте «физическую» диаграмму. Постройте диаграмму прецедентов.

Предметная область: Страховая медицинская компания

Сущность задачи: Страховая медицинская компания (СМК) заключает договоры добровольного медицинского страхования с населением и договоры с лечебными учреждениями на лечение застрахованных клиентов. При возникновении страхового случая клиент подает заявку на оказание медицинских услуг по условиям договора инспектору, который работает с данным клиентом. Инспектор направляет данного клиента в лечебное учреждение. Отчеты о своей деятельности инспектор предоставляет в бухгалтерию. Бухгалтерия проверяет оплату договоров, перечисляет денежные средства за оказанные услуги лечебным учреждениям, производит отчисления в налоговые органы и предоставляет отчетность в органы государственной статистики. СМК не только оплачивает лечение застрахованного лица при возникновении с ним страхового случая, но и, при возникновении каких-либо осложнений после лечения, оплачивает лечение этих осложнений.

**Задание 3.** Определите внешних исполнителей (контрагентов компании). Создайте «физическую» диаграмму. Постройте диаграмму прецедентов.

Предметная область: Банк

Сущность задачи: Банк – это предприятие, осуществляющее регулирование платежного оборота в наличной и безналичной формах. Банк привлекает денежные средства физических и юридических лиц во вклады; размещает привлеченные средства от своего имени и за свой счет; открывает и ведет банковские счета физических и юридических лиц; инкассирует денежные средства, векселя, платежные и расчетные документы; производит кассовое обслуживание физических и юридических лиц; производит куплю-продажу иностранной валюты в наличной и безналичной формах; предоставляет услугу хранения ценных бумаг и драгоценных металлов.

## ПРАКТИЧЕСКАЯ РАБОТА № 10

**Тема:** Разработка перечня артефактов и протоколов проекта

**Цель работы:** изучение процесса разработки необходимого перечня артефактов и протоколов создаваемого проекта.

**Оборудование:** ПК, программное обеспечение – Visual Paradigm Online, MS Word, инструкции по выполнению работы.

**Справочный материал:**

**Системный анализ и пути решения задачи**

При разработке ПС человек имеет дело с системами. Под *системой* будем понимать совокупность взаимодействующих (находящихся в отношениях) друг с другом элементов. ПС можно рассматривать как пример системы. Логически связанный набор программ является другим примером системы. Любая отдельная программа также является системой. Понять систему – значит осмысленно перебрать все пути взаимодействия между ее элементами.

Целью системного анализа в наиболее общем виде является описание и исследование систем, определение путей и методов разработки ПО. Система характеризуется структурой и поведением. Применительно к разработке ПО системный анализ представляет собой анализ существующей структуры отношений в рамках конкретной предметной области, выявление роли и места будущей программной системы, ее основных функций и свойств. В этой связи системный анализ также можно назвать *внешним проектированием*.

Этап системного анализа состоит из следующих трех стадий:

1. обоснование необходимости разработки программы;
2. научно-исследовательские работы (НИР);
3. разработка и утверждение технического задания.

На первой стадии выполняются постановка задачи, сбор исходных материалов, выбор и обоснование критериев эффективности и качества разрабатываемой программы, обоснование необходимости проведения научно-исследовательских работ.

На стадии научно-исследовательских работ решаются следующие задачи: определяется структура входных и выходных данных, осуществляется предварительный выбор методов решения задач, обосновывается целесообразность применения ранее разработанных программ, определяются требования к техническим средствам, обосновывается принципиальная возможность решения поставленной задачи.

На стадии разработки и утверждения технического задания определяются требования к программе, разрабатываются технико-экономического обоснования разработки программ, определяются стадии, этапы и сроки разработки программы и документации на нее, согласовывается и утверждается *техническое задание*.

Результат системного анализа — **спецификация** (техническое задание) как самостоятельный документ имеет очень важное значение. Этот документ



является формальным соглашением между заказчиком продукта и его разработчиками.

### **Содержание работы:**

#### **Задание.**

1. В соответствии с подготовленным техническим заданием выполнить разработку спецификаций на программный продукт, которые должны включать:
  - спецификации процессов;
  - словарь терминов;
  - диаграммы переходов состояний;
  - диаграммы потоков с детализацией.
2. Оформить отчет. Содержание отчета:
  - тема практической работы;
  - цель практической работы;
  - разработанные спецификации процессов;
  - словарь терминов;
  - диаграммы переходов состояний;
  - диаграммы потоков с детализацией;
  - выводы по проделанной работе.

### **Варианты заданий:**

1. Кадровое агентство способствует трудоустройству безработных граждан. Агентство ведет учет и классификацию данных о безработных на основании резюме от них. От предприятий города поступают данные о свободных вакансиях, на основании которых агентство предлагает различные варианты трудоустройства соискателям. В случае положительного исхода поиска вакансия считается заполненной, а безработный становится трудоустроенным. По результатам своей деятельности кадровое агентство производит отчисления в налоговые органы и предоставляет отчетность в органы государственной статистики.
2. Компания по разработке программных продуктов заключает договор с клиентом на разработку программного продукта согласно техническому заданию. После утверждения технического задания определяется состав и объем работ, составляется предварительная смета. На каждый проект назначается ответственный за его выполнение – куратор проекта, который распределяет нагрузку между программистами и следит за выполнением технического задания. Когда программный продукт готов, то его внедряют, производят обучение клиента и осуществляют дальнейшее сопровождение. По результатам своей деятельности компания производит отчисления в налоговые органы и предоставляет отчетность в органы государственной статистики.
3. Туроператор предоставляет возможность своим клиентам осуществить туристическую или деловую поездку в различные города России и мира. При разработке нового тура сначала анализируется текущая ситуация на рынке

туризма и выбирается направление тура. После этого определяется статус тура, бронируются места в гостиницах и билеты на переезд к месту тура, разрабатывается культурная/деловая/развлекательная программа, подтверждаются сроки тура. На каждый тур назначается ответственное лицо от туроператора, которое будет вести данный тур для улаживания проблем в случае возникновения каких-нибудь чрезвычайных или форс-мажорных ситуаций. Клиент приходит в офис туроператора, где вместе с менеджером выбирает уже разработанный тур и оформляет путевку. После возвращения из тура клиент может высказать свои замечания или пожелания, которые будут учтены при доработке существующих туров или при разработке новых. Также, для дальнейшего улучшения тура, туроператор проводит анализ отчетов от посредников (гостиница, гиды и т.д.). По результатам своей деятельности туроператор производит отчисления в налоговые органы и предоставляет отчетность в органы государственной статистики.

## **Тема: Разработка перечня артефактов и протоколов проекта**

**Цель работы:** изучение процесса разработки необходимого перечня артефактов и протоколов создаваемого проекта.

**Оборудование:** ПК, программное обеспечение – Visual Paradigm Online, MS Word, инструкции по выполнению работы.

### **Содержание работы:**

#### **Задание.**

1. В соответствии с подготовленным техническим заданием выполнить разработку спецификаций на программный продукт, которые должны включать:

- спецификации процессов;
- словарь терминов;
- диаграммы переходов состояний;
- диаграммы потоков с детализацией.

2. Оформить отчет. Содержание отчета:

- тема практической работы;
- цель практической работы;
- разработанные спецификации процессов;
- словарь терминов;
- диаграммы переходов состояний;
- диаграммы потоков с детализацией;
- выводы по проделанной работе.

#### **Варианты заданий:**

1. Строительная организация занимается строительством объектов по заказам клиентов. Сначала заказ проходит предварительную стадию: сбор различных разрешений на строительство, составление эскиза объекта, расчет объема и закупка строительных материалов. Сами строительные материалы доставляются на объект партиями. По мере поступления очередной партии стройматериалов закладывается фундамент объекта, строится каркас здания. По результатам данной работы происходит согласование с заказчиком, после чего утепляется контур, вставляются окна, устанавливается крыша. Далее идет обсуждение с клиентом внутренней отделки здания, закупаются отделочные материалы. После того, как объект проходит технический контроль, он передается заказчику. В дополнительные услуги строительной организации входят: услуги дизайнера по интерьеру, закупка и доставка мебели, сотрудничество с охранным предприятием по установке сигнализации. По результатам своей деятельности строительная организация производит отчисления в налоговые органы и предоставляет отчетность в органы государственной статистики.

2. Компьютерная компания занимается продажей, ремонтом, сборкой, тестированием компьютерной техники. Также, специалисты компании предоставляют услуги по разработке и монтажу локальных вычислительных сетей. Вся техника и комплектующие закупаются оптом у дилеров и хранятся

на складе. Клиент, который хочет приобрести товар, оформляет заказ в торговом зале, а забирает технику со склада или оставляет заявку на ее доставку. Клиент, который хочет отремонтировать технику, приносит ее в сервисный отдел, откуда, по прошествии некоторого времени, забирает как отремонтированную или как технику, не подлежащую ремонту. По желанию клиента, специалисты компании могут выехать к клиенту для общей диагностики возникшей проблемы с техникой. По результатам своей деятельности компьютерная компания производит отчисления в налоговые органы и предоставляет отчетность в органы государственной статистики.

3. Компания предоставлению телекоммуникационных услуг занимается оказанием телекоммуникационных услуг абонентам. Клиент делает заявку на подключение к телекоммуникационным услугам и ему, по необходимости, устанавливают соответствующее оборудование. Оплата за услуги вносится путем авансовых платежей. Каждый факт предоставления услуги фиксируется соответствующим оборудованием и является основанием для списания соответствующей суммы с личного счета абонента. Клиент в любое время суток может получить отчет об оказанных ему услугах, их стоимости и остатку на личном счете абонента. По результатам своей деятельности компания производит отчисления в налоговые органы и предоставляет отчетность в органы государственной статистики.

## **ПРАКТИЧЕСКАЯ РАБОТА № 12**

**Тема: Разработка перечня артефактов и протоколов проекта**

**Цель работы:** изучение процесса разработки необходимого перечня артефактов и протоколов создаваемого проекта.

**Оборудование:** ПК, программное обеспечение – Visual Paradigm Online, MS Word, инструкции по выполнению работы.

**Содержание работы:**

**Задание.**

1. В соответствии с подготовленным техническим заданием выполнить разработку спецификаций на программный продукт, которые должны включать:

- спецификации процессов;
- словарь терминов;
- диаграммы переходов состояний;
- диаграммы потоков с детализацией.

2. Оформить отчет. Содержание отчета:

- тема практической работы;
- цель практической работы;
- разработанные спецификации процессов;
- словарь терминов;
- диаграммы переходов состояний;
- диаграммы потоков с детализацией;
- выводы по проделанной работе.

**Варианты заданий:**

1. Управляющая компания ЖКХ занимается обслуживанием жилого фонда города. УК получает финансовые средства от населения и бюджета города в виде компенсаций и субсидий на коммунальные услуги. На основании поступивших средств УК осуществляет текущий ремонт жилого фонда, а также капитальный ремонт согласно плану. Для непосредственного выполнения работ УК нанимает соответствующую рабочую силу (сантехников, дворников, электриков и т.д.). По результатам своей деятельности УКЖКХ производит отчисления в налоговые органы и предоставляет отчетность в органы государственной статистики.

2. Автобаза предоставляет услуги по перевозке пассажиров, различных грузов как в черте города, так и между соседними городами. Для регулярных рейсов оплата клиентами услуги происходит в моментах оказания. В остальных случаях клиент должен сделать заявку, которая может быть отклонена. Для междугородных перевозок в диспетчерские автобазы фиксируется маршрут следования рейса. По результатам своей деятельности автобаза производит отчисления в налоговые органы и предоставляет отчетность в органы государственной статистики.

3. Спортивный комплекс предоставляет услуги по проведению спортивных тренировок. Тренировки, относящиеся к одному виду спорта, объединяются

в спортивные секции. Клиент обращается в спортивный комплекс, где получает абонемент на посещение спортивной секции. На основе купленных абонементов составляется расписание тренировок на следующий месяц. Также, в зависимости от загруженности спортивного комплекса, распределяются тренеры спортивных секций. По результатам своей деятельности спортивный комплекс производит отчисления в налоговые органы и предоставляет отчетность в органы государственной статистики.

## ПРАКТИЧЕСКАЯ РАБОТА № 13

**Тема: Настройка работы системы контроля версий (типов импортируемых файлов, путей, фильтров и других параметров импорта в репозиторий)**

**Цель работы:** изучить систему контроля версий, приобрести практические навыки по работе с системой git, научиться производить настройку работы системы контроля версий.

**Оборудование:** ПК, программное обеспечение – GitHub, MS Word, инструкции по выполнению работы.

### **Справочный материал:**

Система управления/контроля версиями (от англ. Version Control System или Revision Control System) — программное обеспечение для облегчения работы с изменяющейся информацией.

Система управления версиями позволяет хранить несколько версий одного и того же документа, при необходимости, возвращаться к более ранним версиям, определять, кто и когда сделал то или иное изменение и многое другое. Такие системы наиболее широко применяются при разработке программного обеспечения, для хранения исходных кодов разрабатываемой программы.

Распространенные системы управления версиями: Subversion, Darcs, Microsoft Visual SourceSafe, Bazaar, Rational ClearCase, Perforce, BitKeeper, Mercurial, Git, GNU Arch, CVS — устаревшая, потомок: Subversion, RCS — устаревшая, потомок: CVS.

### Основные понятия

Репозиторий (repository) – центральное хранилище, которое содержит версии файлов. Очень часто репозиторий организуется средствами какой-нибудь СУБД.

Версия файла (revision) – состояние файла в определенный момент времени. Репозиторий предоставляет возможность хранить неограниченное число версий одного и того же файла.

Актуальная версия файла – обычно это самая последняя версия файла, размещенного в репозитории.

Рабочая версия файла (working copy) – версия файла, с которой в текущий момент ведется работа, и которая не загружена в репозиторий.

Загрузка (Upload) – размещение файла в репозитории. В процессе загрузки в репозиторий помещается рабочая версия файла.

Выгрузка (Checkout) – получение файла из репозитория. В процессе выгрузки осуществляется получение из репозитория необходимой версии файла.

Синхронизация (update, sync) – приведение в соответствие рабочих версий файлов с актуальными версиями в репозитории. В процессе синхронизации в репозиторий загружаются те файлы, рабочие копии которых являются более "свежими" (т.е. имеют более поздние версии), по сравнению с файлами в репозитории, и выгружаются те файлы, рабочие копии которых устарели по сравнению с копиями в репозитории.

## **Содержание работы:**

### **Задание.**

1. Установка Git. Для начала работы с Git необходимо установить его на свой компьютер. Git можно скачать с официального сайта (<https://git-scm.com/downloads>) и установить, следуя инструкциям установщика.

2. Создание репозитория. После установки Git можно создать новый репозиторий. Для этого необходимо открыть командную строку (терминал) и перейти в папку, где будет располагаться репозиторий. Затем выполните следующую команду: `git init`

Эта команда создаст новый пустой репозиторий в текущей папке.

3. Добавление файлов в репозиторий. Чтобы добавить файлы в репозиторий, необходимо выполнить следующую команду: `git add <имя файла>`

Здесь <имя файла> - это имя файла, который вы хотите добавить. Вы также можете использовать символ \*, чтобы добавить все файлы в текущей папке.

4. Создание коммита. После добавления файлов в репозиторий необходимо создать коммит. Коммит фиксирует изменения и добавляет их в историю репозитория. Для создания коммита выполните следующую команду:

`git commit -m "Описание изменений"`

Здесь "Описание изменений" - это описание внесенных изменений. Оно должно быть информативным и описывать, что было сделано в данном коммите.

5. Создание ветки. Для создания новой ветки выполните следующую команду: `git branch <имя ветки>`

Здесь <имя ветки> - это имя новой ветки. Вы также можете использовать команду `git checkout -b <имя ветки>`, чтобы сразу переключиться на новую ветку.

6. Слияние веток. Для слияния изменений из одной ветки в другую выполните следующую команду: `git merge <имя ветки>`

Здесь <имя ветки> - это имя ветки, из которой вы хотите взять изменения. Обратите внимание, что для слияния веток необходимо находиться в той ветке, в которую вы хотите внести изменения.

7. Разрешение конфликтов слияния. Если при слиянии веток возникает конфликт, Git покажет сообщение об ошибке и указывает на файлы, в которых произошел конфликт. Разрешить конфликт можно вручную, открыв файл в текстовом редакторе и выбрав нужные изменения.

### **Контрольные вопросы:**

1. Система управления версиями это?
2. На чем основана CVS?
3. Какие системы распределения контроля версий Вы знаете?
4. Охарактеризуйте систему Git.
5. Что такое клонирование репозитория?
6. Что такое синхронизация репозитория?



## **ПРАКТИЧЕСКАЯ РАБОТА № 14**

**Тема: Разработка и интеграция модулей проекта (командная работа)**

**Цель работы:** изучить принципы разработки и интеграции модулей проекта и принципы формирования и управления командной работы.

**Оборудование:** ПК, программное обеспечение – Visual Studio, Visual Paradigm Online, MS Word, инструкции по выполнению работы.

### **Справочный материал:**

Термин «интеграция» относится к такой операции в процессе разработки ПО, при которой вы объединяете отдельные программные компоненты в единую систему. В небольших проектах интеграция может занять одно утро и заключаться в объединении горстки классов. В больших — могут потребоваться недели или месяцы, чтобы связать воедино весь набор программ. Независимо от размера задач в них применяются одни и те же принципы.

Тема интеграции тесно переплетается с вопросом последовательности конструирования. Порядок, в котором вы создаете классы или компоненты, влияет на порядок их интеграции: вы не можете интегрировать то, что еще не было создано. Последовательности интеграции и конструирования имеют большое значение.

Поскольку интеграция выполняется после того, как разработчик завершил модульное тестирование, и одновременно с системным тестированием, ее иногда считают операцией, относящейся к тестированию. Однако она достаточно сложна, и поэтому ее следует рассматривать как независимый вид деятельности.

Аккуратная интеграция обеспечивает: упрощенную диагностику дефектов; меньшее число ошибок; меньшее количество «лесов»; раннее создание первой работающей версии продукта; уменьшение общего времени разработки; лучшие отношения с заказчиком; улучшение морального климата; увеличение шансов завершения проекта; более надежные оценки графика проекта; более аккуратные отчеты о состоянии; лучшее качество кода; меньшее количество документации.

### **Содержание работы:**

#### **Задание.**

1. Разработать модули будущей информационной системы автоматизации библиотечного каталога. Оформить внешнюю спецификацию модулей. В спецификацию включить внешнее описание модуля, как подключается модуль, какие данные на входе/выходе модуля, структура модуля и средства защиты информации.
2. Составить в виде функциональной и (или) структурной схемы общий алгоритм работы ПО.
3. Спроектировать и разработать модули программы для решения задачи на любом языке программирования.
4. Выполнить инкрементную интеграцию модулей с использованием одного из подходов. Дать определение инкрементной интеграции модуля и описать подходы.

5. Оформить отчет по практической работе.

Отчет по практической работе должен включать:

1. Внешнюю спецификацию.
2. Схемы работы ПО.
3. Текст программы на языке программирования.
4. Описание процесса интеграции модулей.

## **ПРАКТИЧЕСКАЯ РАБОТА № 15**

**Тема: Разработка и интеграция модулей проекта (командная работа)**

**Цель работы:** изучить принципы разработки и интеграции модулей проекта и принципы формирования и управления командной работы.

**Оборудование:** ПК, программное обеспечение – Visual Studio, Visual Paradigm Online, MS Word, инструкции по выполнению работы.

**Содержание работы:**

**Задание.**

1. Разработать модули будущей информационной системы автоматизации деканата ВУЗа. Оформить внешнюю спецификацию модулей. В спецификацию включить внешнее описание модуля, как подключается модуль, какие данные на входе/выходе модуля, структура модуля и средства защиты информации.

2. Составить в виде функциональной и (или) структурной схемы общий алгоритм работы ПО.

3. Спроектировать и разработать модули программы для решения задачи на любом языке программирования.

4. Выполнить инкрементную интеграцию модулей с использованием одного из подходов. Дать определение инкрементной интеграции модуля и описать подходы.

5. Оформить отчет по практической работе.

Отчет по практической работе должен включать:

1. Внешнюю спецификацию.
2. Схемы работы ПО.
3. Текст программы на языке программирования.
4. Описание процесса интеграции модулей.

## **ПРАКТИЧЕСКАЯ РАБОТА № 16**

**Тема: Разработка и интеграция модулей проекта (командная работа)**

**Цель работы:** изучить принципы разработки и интеграции модулей проекта и принципы формирования и управления командной работы.

**Оборудование:** ПК, программное обеспечение – Visual Studio, Visual Paradigm Online, MS Word, инструкции по выполнению работы.

**Содержание работы:**

**Задание.**

1. Разработать модули будущей информационной системы автоматизации аптечного пункта. Оформить внешнюю спецификацию модулей. В спецификацию включить внешнее описание модуля, как подключается модуль, какие данные на входе/выходе модуля, структура модуля и средства защиты информации.

2. Составить в виде функциональной и (или) структурной схемы общий алгоритм работы ПО.

3. Спроектировать и разработать модули программы для решения задачи на любом языке программирования.

4. Выполнить инкрементную интеграцию модулей с использованием одного из подходов. Дать определение инкрементной интеграции модуля и описать подходы.

5. Оформить отчет по практической работе.

Отчет по практической работе должен включать:

1. Внешнюю спецификацию.
2. Схемы работы ПО.
3. Текст программы на языке программирования.
4. Описание процесса интеграции модулей.

## **ПРАКТИЧЕСКАЯ РАБОТА № 17**

**Тема: Разработка и интеграция модулей проекта (командная работа)**

**Цель работы:** изучить принципы разработки и интеграции модулей проекта и принципы формирования и управления командной работы.

**Оборудование:** ПК, программное обеспечение – Visual Studio, Visual Paradigm Online, MS Word, инструкции по выполнению работы.

**Содержание работы:**

**Задание.**

1. Разработать модули будущей информационной системы автоматизации автовокзала. Оформить внешнюю спецификацию модулей. В спецификацию включить внешнее описание модуля, как подключается модуль, какие данные на входе/выходе модуля, структура модуля и средства защиты информации.

2. Составить в виде функциональной и (или) структурной схемы общий алгоритм работы ПО.

3. Спроектировать и разработать модули программы для решения задачи на любом языке программирования.

4. Выполнить инкрементную интеграцию модулей с использованием одного из подходов. Дать определение инкрементной интеграции модуля и описать подходы.

5. Оформить отчет по практической работе.

Отчет по практической работе должен включать:

1. Внешнюю спецификацию.
2. Схемы работы ПО.
3. Текст программы на языке программирования.
4. Описание процесса интеграции модулей.

## **ПРАКТИЧЕСКАЯ РАБОТА № 18**

**Тема: Разработка и интеграция модулей проекта (командная работа)**

**Цель работы:** изучить принципы разработки и интеграции модулей проекта и принципы формирования и управления командной работы.

**Оборудование:** ПК, программное обеспечение – Visual Studio, Visual Paradigm Online, MS Word, инструкции по выполнению работы.

**Содержание работы:**

**Задание.**

1. Разработать модули будущей информационной системы автоматизации кинотеатра. Оформить внешнюю спецификацию модулей. В спецификацию включить внешнее описание модуля, как подключается модуль, какие данные на входе/выходе модуля, структура модуля и средства защиты информации.

2. Составить в виде функциональной и (или) структурной схемы общий алгоритм работы ПО.

3. Спроектировать и разработать модули программы для решения задачи на любом языке программирования.

4. Выполнить инкрементную интеграцию модулей с использованием одного из подходов. Дать определение инкрементной интеграции модуля и описать подходы.

5. Оформить отчет по практической работе.

Отчет по практической работе должен включать:

1. Внешнюю спецификацию.
2. Схемы работы ПО.
3. Текст программы на языке программирования.
4. Описание процесса интеграции модулей.

## ПРАКТИЧЕСКАЯ РАБОТА № 19

**Тема: Отладка отдельных модулей программного проекта. Организация обработки исключений.**

**Цель работы:** усвоить знание основ модульного программирования; освоить способы создания и применения модулей.

**Оборудование:** ПК, программное обеспечение – Visual Studio, MS Word, инструкции по выполнению работы.

### **Справочный материал:**

*Отладка* — это процесс определения и устранения причин ошибок. Этим она отличается от тестирования, направленного на обнаружение ошибок. В некоторых проектах отладка занимает до 50% общего времени разработки. Многие программисты считают отладку самым трудным аспектом программирования.

*Отладка* — это локализация и устранение ошибок. Отладка является следствием успешного тестирования. Это значит, что если тестовый вариант обнаруживает ошибку, то процесс отладки уничтожает ее.

Итак, процессу отладки предшествует выполнение тестового варианта. Его результаты оцениваются, регистрируется несоответствие между ожидаемым и реальным результатами. Несоответствие является симптомом скрытой причины. Процесс отладки пытается сопоставить симптом с причиной, вследствие чего приводит к исправлению ошибки. Возможны два исхода процесса отладки:

- 1) причина найдена, исправлена, уничтожена;
- 2) причина не найдена.

Во втором случае отладчик может предполагать причину. Для проверки этой причины он просит разработать дополнительный тестовый вариант, который поможет проверить предположение. Таким образом, запускается итерационный процесс коррекции ошибки.

Возможные разные способы проявления ошибок:

- 1) программа завершается нормально, но выдает неверные результаты;
- 2) программа зависает;
- 3) программа завершается по прерыванию;
- 4) программа завершается, выдает ожидаемые результаты, но хранимые данные испорчены (это самый неприятный вариант).

*Цель отладки* — найти оператор программы, при исполнении которого правильные аргументы приводят к неправильным результатам. Если место проявления симптома ошибки не является искомой аномалией, то один из аргументов оператора должен быть неверным. Поэтому надо перейти к исследованию предыдущего оператора, выработавшего этот неверный аргумент. В итоге пошаговое обратное прослеживание приводит к искомому ошибочному месту.

### *Конфигурация компиляции*

Для того, чтобы отладка программы была доступна, программа должна быть еще запущена в режиме отладки и из среды разработки. Простое нажатие клавиши F5 как раз запускает программу в отладочном режиме.

Чтобы запустить программу без возможности отладки, нужно нажать Ctrl+F5.

Для отладки программа должна быть скомпилирована в конфигурации Debug, которая включает в исполняемый файл дополнительную информацию, необходимую при отладке программы. Конфигурацию можно выбрать на панели инструментов чуть правее кнопки запуска программы.

**Содержание работы:**

**Задание 1.** Отладить один из модулей вашей ИС Библиотечного каталога с использованием встроенных в среду разработки инструментальных средств. Составить отчет по практической работе.

**Отчет** по практической работе должен включать:

1. Текст модуля на языке программирования до отладки и после.
2. Выявленные ошибки и как вы их исправили.

**Задание 2.** Отладить один из модулей вашей ИС Автовокзал с использованием встроенных в среду разработки инструментальных средств. Составить отчет по практической работе.

**Отчет** по практической работе должен включать:

1. Текст модуля на языке программирования до отладки и после.
2. Выявленные ошибки и как вы их исправили.



## ПРАКТИЧЕСКАЯ РАБОТА № 20

**Тема: Отладка отдельных модулей программного проекта. Организация обработки исключений.**

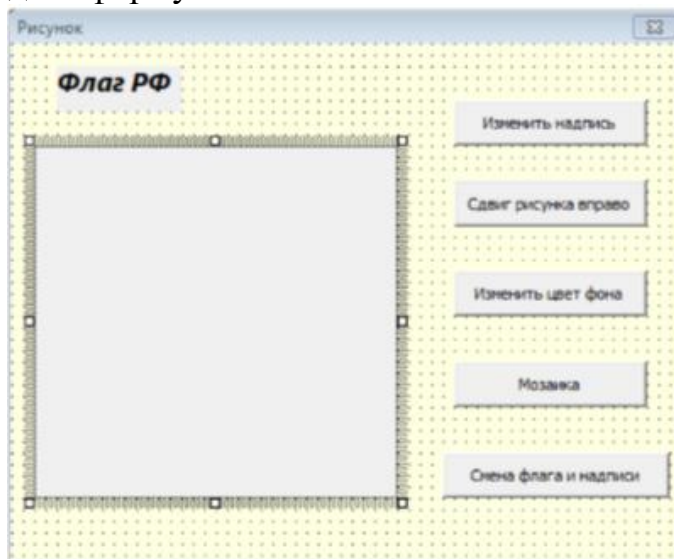
**Цель работы:** усвоить знание основ модульного программирования; освоить способы создания и применения модулей.

**Оборудование:** ПК, программное обеспечение – Visual Studio, MS Word, инструкции по выполнению работы.

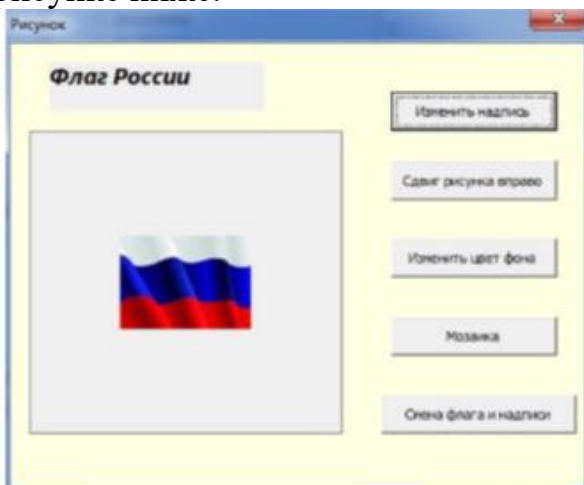
**Содержание работы:**

**Задание 1.** Произвести отладку модулей программного продукта

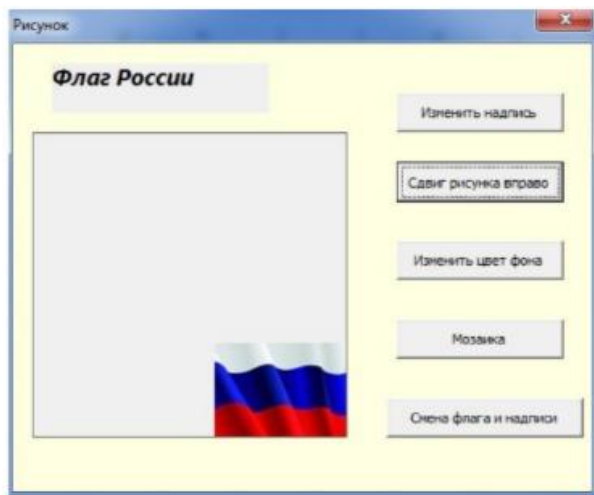
1. Создать форму пользователя.



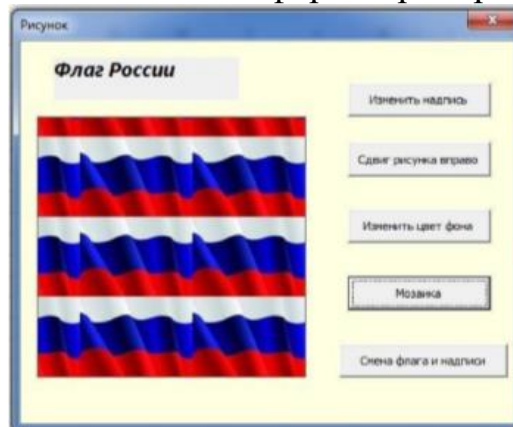
2. Создайте новую процедуру для кнопки «Измени надпись».
3. Введите текст процедуры. Правильно укажите адрес графического файла.
4. Выполните компиляцию и выведите форму для работы.
5. После появления формы на экране нажмем на кнопку «Измени надпись».
6. После щелчка по кнопке «Измени надпись» форма приобретет вид, представленный на рисунке ниже.



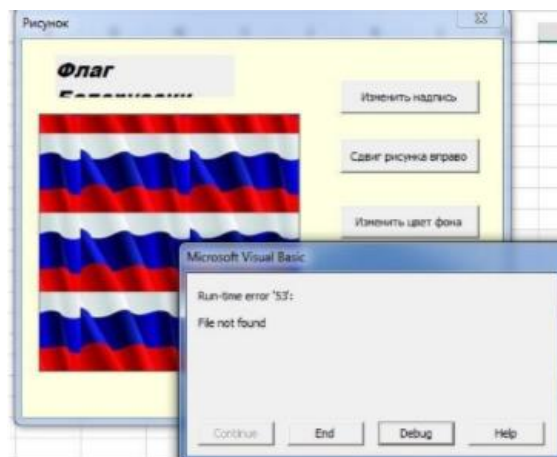
7. После щелчка по кнопке «Сдвинь рисунок вправо» форма приобретет вид, представленный на рисунке.



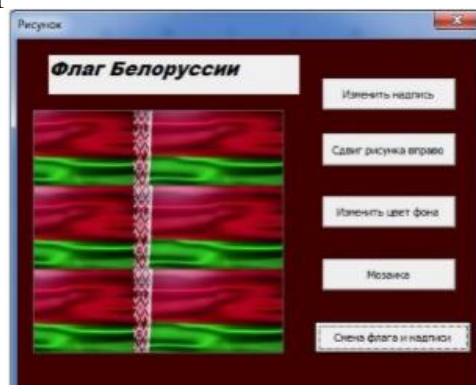
8. После щелчка по кнопке «Мозаика» форма приобретет вид



9. После щелчка по кнопке «Смена флага и надписи» появится сообщение об ошибке.



10. Добавить еще один флаг



Составить отчет по практической работе:

1. Текст модуля на языке программирования до отладки и после.
2. Выявленные ошибки и как вы их исправили.

## ПРАКТИЧЕСКАЯ РАБОТА № 21

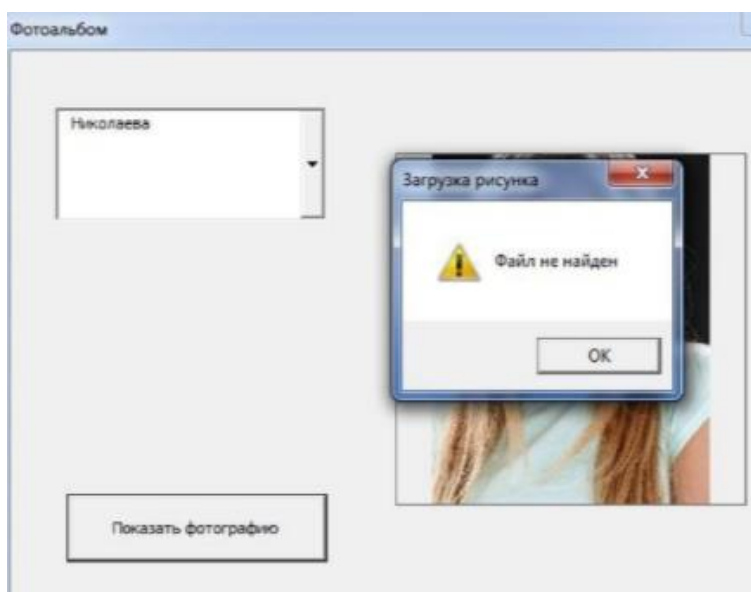
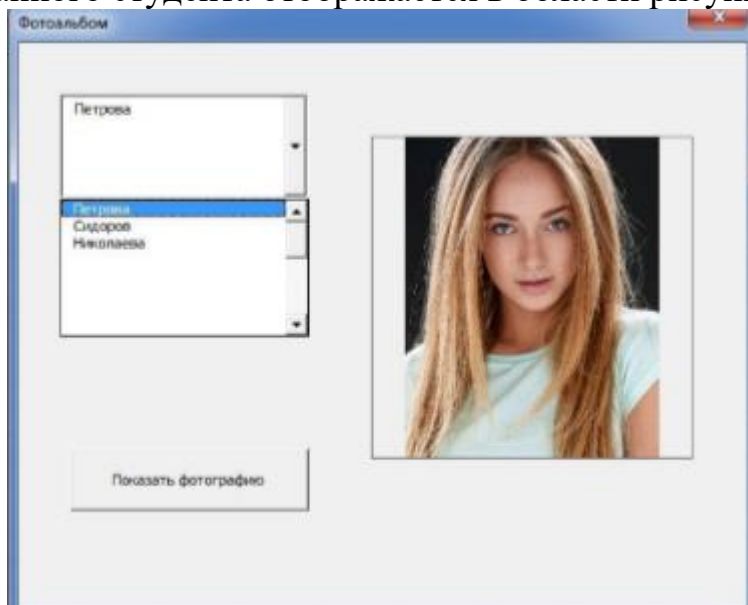
**Тема:** Отладка отдельных модулей программного проекта. Организация обработки исключений.

**Цель работы:** усвоить знание основ модульного программирования; освоить способы создания и применения модулей.

**Оборудование:** ПК, программное обеспечение – Visual Studio, MS Word, инструкции по выполнению работы.

**Содержание работы:**

**Задание 1.** Создать программу, которая показывает фотографии однокурсников. Фамилии студентов выбираются из раскрывающегося списка ComboBox и при нажатии кнопки «Показать фотографию» фотография выбранного студента отображается в области рисунка.



Модуль формы состоит из трех процедур:

- процедура обработки ошибочной ситуации;
- процедура обработки события Initialize объекта Userform;
- процедура обработки события CommandButton1\_Click.

Процедура `Обработка_ошибки` с помощью оператора выбора анализирует код ошибки. Если имя файла задано неверно, при загрузке рисунка возникает ошибка с кодом 53, если путь к файлу задан неверно - ошибка с кодом 76. Обработчик ошибок выводит соответствующее сообщение. При появлении других выполнение программы прерывается, пользователь информируется об ошибке. Осуществляется выход из процедуры.

Процедура обработки события `Initialize` объекта заполняет список `ComboBox` с фамилиями студентов, список `ListBox` – с полными именами файлов. В раскрывающемся списке отображается фамилия первого в списке студента, а в области рисунка - его фотография. При возникновении ошибки при загрузке рисунка вызывается процедура `Обработка_ошибки`.

Процедура обработки события `CommandButton1` фотографию выбранного из раскрывающегося списка отображает в области рисунка. Если при загрузке рисунка возникает ошибка, вызывается процедура `Обработка_ошибки`.

2. Проведите отладку программного продукта.
3. Составить отчет по практической работе:
  1. Текст модуля на языке программирования до отладки и после.
  2. Выявленные ошибки и как вы их исправили.

## **ПРАКТИЧЕСКАЯ РАБОТА № 22**

### **Тема: Отладка проекта**

**Цель работы:** научиться применять правила и последовательности отладки программного продукта.

**Оборудование:** ПК, программное обеспечение – Visual Studio, MS Word, инструкции по выполнению работы.

#### **Справочный материал:**

Тестирование – процесс выполнения программы на наборе тестов с целью выявления ошибок.

Локализацией называют процесс определения оператора программы, выполнение которого вызвало нарушение нормального вычислительного процесса. Для исправления ошибки необходимо определить ее причину, т.е. определить оператор или фрагмент, содержащие ошибку. В целом сложность отладки обусловлена следующими причинами:

- требует от программиста глубоких знаний специфики управления используемыми техническими средствами, операционной системы, среды и языка программирования, реализуемых процессов, природы и специфики различных ошибок, методик отладки и соответствующих программных средств;
- психологически дискомфортна, так как необходимо искать собственные ошибки и, как правило, в условиях ограниченного времени;
- возможно взаимовлияние ошибок в разных частях программы, например, за счет затирания области памяти одного модуля другим из-за ошибок адресации;
- отсутствуют четко сформулированные методики отладки.

Большинство ошибок можно обнаружить по косвенным признакам посредством тщательного анализа текстов программ и результатов тестирования без получения дополнительной информации. При этом используют различные методы: ручного тестирования; индукции; дедукции; обратного прослеживания.

#### **Метод ручного тестирования**

Это – самый простой и естественный способ данной группы. При обнаружении ошибки необходимо выполнить тестируемую программу вручную, используя тестовый набор, при работе с которыми была обнаружена ошибка. Метод очень эффективен, но не применим для больших программ, программ со сложными вычислениями и в тех случаях, когда ошибка связана с неверным представлением программиста о выполнении некоторых операций. Данный метод часто используют как составную часть других методов отладки.

#### **Метод индукции**

Метод основан на тщательном анализе симптомов ошибки, которые могут проявляться как неверные результаты вычислений или как сообщение об ошибке. Если компьютер просто "зависает", то фрагмент проявления ошибки вычисляют, исходя из последних полученных результатов и действий пользователя. Полученную таким образом информацию

организуют и тщательно изучают, просматривая соответствующий фрагмент программы. В результате этих действий выдвигают гипотезы об ошибках, каждую из которых проверяют. Если гипотеза верна, то детализируют информацию об ошибке, иначе – выдвигают другую гипотезу.

Самый ответственный этап – выявление симптомов ошибки. Организуя данные об ошибке, целесообразно записать все, что известно о ее проявлениях, причем фиксируют, как ситуации, в которых фрагмент с ошибкой выполняется нормально, так и ситуации, в которых ошибка проявляется. Если в результате изучения данных никаких гипотез не появляется, то необходима дополнительная информация об ошибке. В процессе доказательства пытаются выяснить, все ли проявления ошибки объясняет данная гипотеза, если не все, то либо гипотеза не верна, либо ошибок несколько.

### **Метод дедукции**

По методу дедукции вначале формируют множество причин, которые могли бы вызвать данное проявление ошибки. Затем анализируя причины, исключают те, которые противоречат имеющимся данным. Если все причины исключены, то следует выполнить дополнительное тестирование исследуемого фрагмента. В противном случае наиболее вероятную гипотезу пытаются доказать. Если гипотеза объясняет полученные признаки ошибки, то ошибка найдена, иначе – проверяют следующую причину.

### **Метод обратного прослеживания**

Для небольших программ эффективно применение метода обратного прослеживания. Начинают с точки вывода неправильного результата. Для этой точки строится гипотеза о значениях основных переменных, которые могли бы привести к получению имеющегося результата. Далее, исходя из этой гипотезы, делают предложения о значениях переменных в предыдущей точке. Процесс продолжают, пока не обнаружат причину ошибки.

### **Содержание работы:**

#### **Задание**

Задача: составить список учебной группы, включающей 25 человек. Для каждого учащегося указать дату рождения, год поступления в колледж, курс, группу, оценки каждого года обучения.

Назначение задачи: получить значение определенного критерия и упорядочить список студентов по нему.

Достигаемая цель: упорядочить список студентов по среднему баллу и получить его.

1. Составить в виде блок-схемы алгоритм решения задачи.
2. Создать программу решения задачи на любом языке программирования.
3. Отладить программу.
4. Составить отчет по практической работе.

#### **Отчет по практической работе должен включать:**

1. Алгоритм решения задачи.
2. Текст программы на языке программирования.
3. Набор тестов для отладки программы.

## ПРАКТИЧЕСКАЯ РАБОТА № 23

### Тема: Отладка проекта

**Цель работы:** научиться применять правила и последовательности отладки программного продукта.

**Оборудование:** ПК, программное обеспечение – Visual Studio, MS Word, инструкции по выполнению работы.

#### Справочный материал:

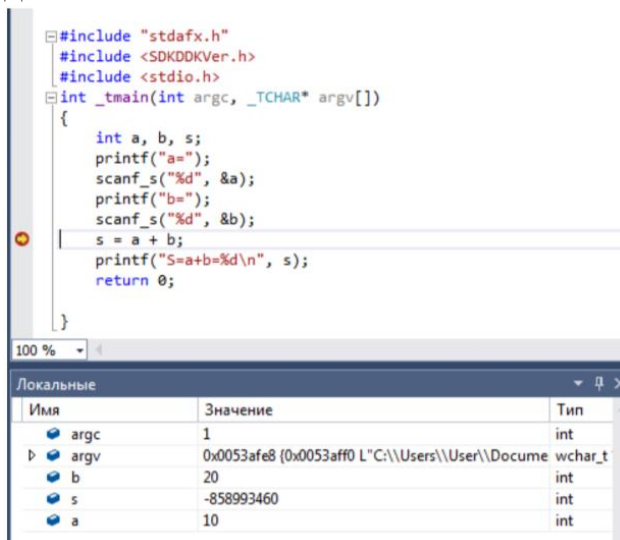
Рассмотрим действия для отладки проекта.

1. Установка точки останова. В файле с исходным кодом программы нужно установить курсор на строчку, где требуется остановка. После этого выполнить одно из действий:

- Отладка - Точка останова;
- нажать горячую клавишу F9;
- ПКМ – Вставить точку останова.

```
#include "stdafx.h"
#include <SDKDDKVer.h>
#include <stdio.h>
int _tmain(int argc, _TCHAR* argv[])
{
    int a, b, s;
    printf("a=");
    scanf_s("%d", &a);
    printf("b=");
    scanf_s("%d", &b);
    s = a + b;
    printf("S=a+b=%d\n", s);
    return 0;
}
```

2. Запуск программы до ближайшей точки останова. После указания точки останова можно запустить программу: команда Отладка▶Начать отладку или нажать горячую клавишу F5. Программа начнет выполняться до тех пор, пока не будет достигнута точка останова. О достижении этой точки будет свидетельствовать стрелочка, отображаемая поверх точки останова. В этой точке выполнение программы будет остановлено до указания дальнейших действий.

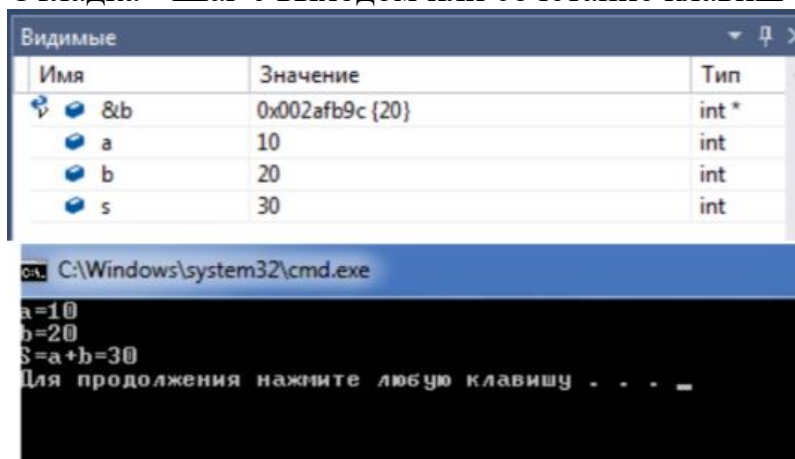


```
#include "stdafx.h"
#include <SDKDDKVer.h>
#include <stdio.h>
int _tmain(int argc, _TCHAR* argv[])
{
    int a, b, s;
    printf("a=");
    scanf_s("%d", &a);
    printf("b=");
    scanf_s("%d", &b);
    s = a + b;
    printf("S=a+b=%d\n", s);
    return 0;
}
```

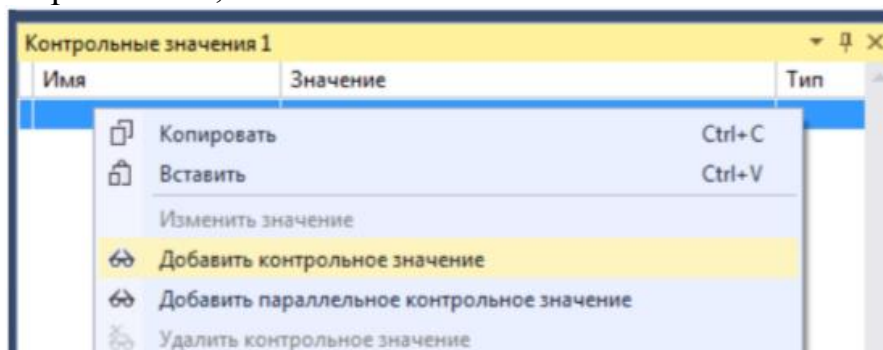
Имя	Значение	Тип
argc	1	int
argv	0x0053afe8 {0x0053aff0 L"C:\Users\User\Documents\...}	wchar_t
b	20	int
s	-858993460	int
a	10	int



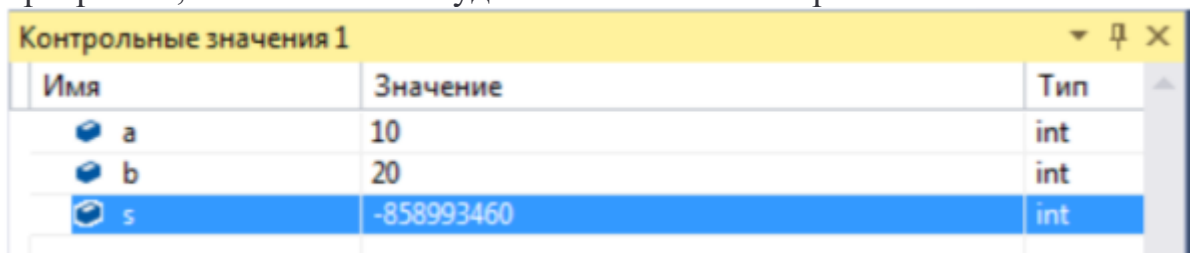
3.Выполнение программы по шагам. Выполнение программы по шагам можно осуществить следующим образом. Команда Отладка⇨Шаг с обходом выполняет одну строчку программы. Если очередным шагом программы является вызов функции, то программист может проанализировать работу этой функции. Для этого в режиме отладки используется команда Отладка - Шаг с заходом или горячая клавиша F11. После этой команды отладчик заходит «внутри» функции и программист может продолжить пошаговое выполнение команд. Если отладка функции закончена и необходимо вернуться в вызывающую программу, то можно использовать команду Отладка - Шаг с выходом или сочетание клавиш Shift+F11.



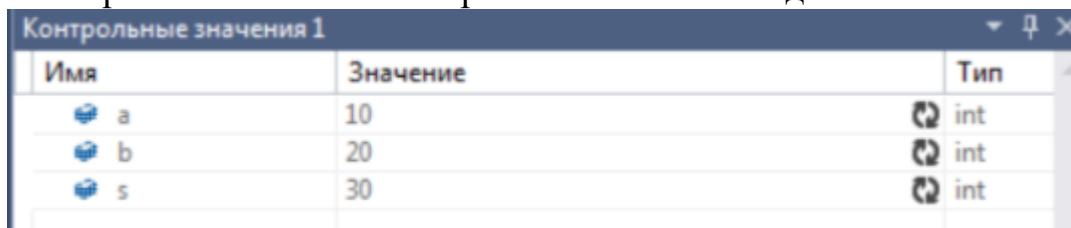
4. Просмотр значения переменных в процессе выполнения программы. В любой момент отладки программист может посмотреть значения переменных, используемых в программе. Для этого используется нижняя часть окна среды Microsoft Visual Studio. Откройте окно Контрольные значения, выбрав Отладка - Окна – Контрольные значения – Контрольные значения 1. Вы можете открыть дополнительные окна контрольных значений, выбрав окна 2, 3 или 4.



Для добавления переменной необходимо сделать ее активной и ввести с клавиатуры имя переменной. Если указанная переменная имеется в программе, то ее значение будет автоматически отражено в поле Значение.



Продолжите отладку, выбрав Отладка - Шаг с заходом или нажав клавишу F11 по мере необходимости для перехода. В процессе выполнения значения переменных в окне Контрольные значения 1 должны меняться.



Имя	Значение	Тип
a	10	int
b	20	int
s	30	int

Таким образом, используя описанные приемы, программист может тщательно проанализировать выполнение написанной программы, проследить изменения значений используемых переменных, устранить некорректную работу реализованных алгоритмов и добиться правильного выполнения программы.

### **Содержание работы:**

**Задание 1.** Для следующих задач написать код программы на любом языке программирования и провести отладку кода, используя точки останова.

1. Дана квадратная вещественная матрица размерности  $n$ . Является ли матрица симметричной относительно главной диагонали.
2. Дана квадратная вещественная матрица размерности  $n$ . Транспонировать матрицу.
3. Дана квадратная вещественная матрица размерности  $n$ . Сравнить сумму элементов матрицы на главной и побочной диагоналях.

Составить отчет по практической работе:

1. Алгоритм решения задачи.
2. Текст программы на языке программирования.
3. Набор тестов для отладки программы.

## **ПРАКТИЧЕСКАЯ РАБОТА № 24**

### **Тема: Отладка проекта**

**Цель работы:** научиться применять правила и последовательности отладки программного продукта.

**Оборудование:** ПК, программное обеспечение – Visual Studio, MS Word, инструкции по выполнению работы.

#### **Содержание работы:**

**Задание 1.** Для следующих задач написать код программы на любом языке программирования и провести отладку кода, используя точки останова.

1. Напишите программу со статическим методом для сравнения текстовых строк. Строки на предмет совпадения сравниваются посимвольно. Правило сравнения символов такое: два символа считаются одинаковыми, если их коды отличаются не больше, чем на единицу. Текстовые строки совпадают, если у них совпадают символы.

2. Написать программу, которая определяет, состоят ли два массива из одинаковых элементов (без учета порядка следования).

3. Написать программу, определяющую, сколько четных и нечетных элементов содержится в массиве

Составить отчет по практической работе:

1. Алгоритм решения задачи.
2. Текст программы на языке программирования.
3. Набор тестов для отладки программы.

## **ПРАКТИЧЕСКАЯ РАБОТА № 25**

### **Тема: Инспекция кода модулей проекта**

**Цель работы:** изучение формальных инспекций программного кода, научиться проводить инспекцию кода модулей проекта.

**Оборудование:** ПК, программное обеспечение – Visual Studio, MS Word, инструкции по выполнению работы.

#### **Справочный материал:**

Не во всех случаях возможна разработка автоматических или хотя бы четко формализованных ручных тестов для проверки функциональности программной системы. В некоторых случаях выполнение программного кода, подвергаемого тестированию, невозможно в условиях, создаваемых тестовым окружением (например, во встроенных системах, если программный код предназначен для обработки исключительных ситуаций, создаваемых только при установке системы на реальное оборудование). В других случаях верифицируется не программный код, а проектная документация на систему, которую нельзя "выполнить" или создать для нее отдельные тестовые примеры.

И в тех и в других случаях обычно прибегают к методу экспертных исследований программного кода или документации на корректность или непротиворечивость. Такие экспертные исследования обычно называют инспекциями или просмотрами. Существует два типа инспекций – неформальные и формальные. Формальная инспекция является четко управляемым процессом, структура которого обычно четко определяется соответствующим стандартом проекта. Таким образом, все формальные инспекции имеют одинаковую структуру и одинаковые выходные документы, которые затем используются при разработке.

Факт начала формальной инспекции четко фиксируется в общей базе данных проекта. Также фиксируются документы, подвергаемые инспекции, списки замечаний, отслеживаются внесенные по замечаниям изменения. Этим формальная инспекция похожа на автоматизированное тестирование – списки замечаний имеют много общего с отчетами о выполнении тестовых примеров. В ходе формальной инспекции группой специалистов осуществляется независимая проверка соответствия инспектируемых документов исходным документам. Независимость проверки обеспечивается тем, что она осуществляется инспекторами, не участвовавшими в разработке инспектируемого документа. Входами процесса формальной инспекции являются инспектируемые документы и исходные документы, а выходами – материалы инспекции, включающие список обнаруженных несоответствий и решение об изменении статуса инспектируемых документов.

#### **Содержание работы:**

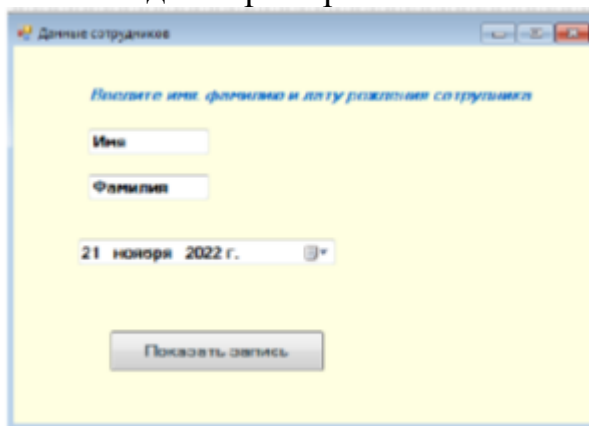
**Задание 1.** Написать программу, которая запрашивает у нового сотрудника имя, фамилию и дату рождения. Требуется создать новый класс с именем Person для хранения этой информации в свойствах, и создать метод класса, который будет вычислять текущий возраст нового сотрудника.

Добавление в ваш проект нового класса. Класс, определенный пользователем, позволяет определить в программе ваши собственные объекты, которые имеют свойства, методы и события, точно так же, как объекты, создаваемые на формах Windows с помощью элементов управления из Области элементов. Чтобы добавить в ваш проект новый класс, щелкните в меню Проект на команде Добавить класс, а затем определите этот класс с помощью кода программы.

1. Запустите Visual Studio, затем создайте в своей папке новый проект с именем Сотрудники.
2. Используйте элемент управления Label и добавьте в верхней части формы Form1 метку.
3. Используйте элемент управления TextBox и нарисуйте под меткой три текстового поля.
4. Используйте элемент управления DateTimePicker и создайте под текстовыми полями объект выбора даты и времени.
5. Используйте элемент управления Button и вставьте под объектом выбора даты и времени кнопку.
6. Установите для объектов формы следующие свойства:

Объект	Свойство	Установка
Label1	Text	Введите имя, фамилию и дату рождения сотрудника.
TextBox1	Text	Имя
TextBox2	Text	Фамилия
Button1	Text	Показать запись
Form1	Text	Данные сотрудников

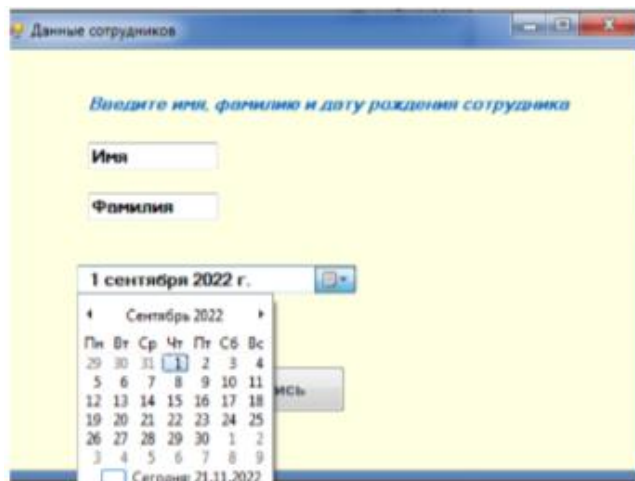
7. Ваша форма должна выглядеть примерно так.



Это базовый интерфейс пользователя для формы, которая определяет запись нового сотрудника фирмы. (Эта форма не подключена к базе данных, так что храниться может только одна запись.) Теперь вы должны добавить в проект класс для хранения информации из этой записи.

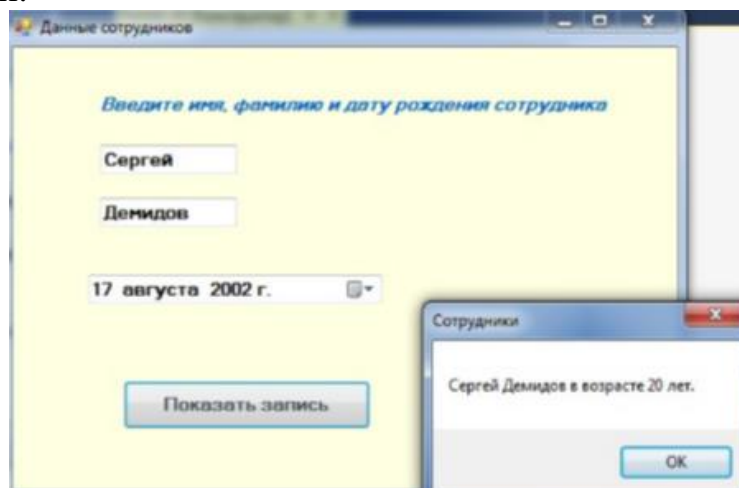
8. Щелкните на команде Добавить класс в меню Проект. Диалоговое окно Добавление нового элемента дает возможность задать имя вашего класса. Когда вы присвоите имя, обратите внимание, что вы можете сохранить в новом модуле класса несколько классов и указать имя, которое будет для них общим.

9. Введите в текстовом поле Имя имя Person.cs, а затем щелкните Добавить. Visual Studio откроет в Редакторе кода пустой модуль класса и добавит имя файла Person.cs в ваш проект в Обозревателе решений.
10. В классе объявите две переменные строкового типа Name, Name1. Создайте свойство вашего класса с именем FirstName, которое имеет тип String. Когда вы нажмете (Enter), VisualStudio немедленно создаст структуру кода для остальных элементов объявления свойства. Заполните структуру процедуры свойства FirstName.
11. Аналогично напишите процедуру свойства LastName. Эта процедура свойства аналогична первой, за исключением того, что она использует вторую строковую переменную (Name2), которую вы объявили в верхней части кода класса.
12. Теперь перейдем к методу с именем Age, который будет определять текущий возраст нового сотрудника на основе даты рождения.
13. Вернитесь к форме Form1 и используйте новый класс в процедуре события. Щелкните в Обозревателе решений на значке Form1.cs. Появится интерфейс пользователя Form1.
14. Чтобы открыть в Редакторе кода процедуру события Button1\_Click, сделайте двойной щелчок мышью на кнопке Показать запись.
15. Напишите процедуру, которая сохраняет в объекте с именем Employee, который имеет тип Person, значения, введенные пользователем. Ключевое слово New указывает, что вы хотите немедленно создать новый экземпляр объекта Employee. Теперь нужно объявить переменную с помощью класса, созданного вами самими. Затем процедура объявляет переменную с именем DOB типа Date. Она будет хранить дату, введенную пользователем, и устанавливает свойства FirstName и LastName объекта Employee равными имени и фамилии, введенным в два объекта текстовых полей формы. Значение, возвращаемое объектом выбора даты и времени, сохраняется в переменной DOB, а последний оператор программы отображает окно сообщения, содержащее свойства FirstName и LastName, а также возраст нового сотрудника, определенный методом Age, который при передаче в него переменной DOB возвращает целое значение. Как только вы определили класс в модуле класса, его легко можно использовать в процедуре события.
16. Чтобы запустить программу, щелкните на кнопке Начать отладку (F5). В среде разработки появится интерфейс пользователя, готовый к приему ваших данных.
17. Введите в текстовое поле FirstName ваше имя, а в текстовое поле LastName - фамилию.
18. Щелкните на раскрывающемся списке объекта выбора даты и времени, и прокрутите его до вашей даты рождения.



19. Щелкните на кнопке Показать запись. Ваша программа сохраняет значения имени и фамилии в свойствах и использует метод Age для вычисления текущего возраста нового сотрудника. Появится диалоговое окно с результатом.

20. Чтобы закрыть это окно сообщения, щелкните на ОК, а затем поэкспериментируйте с несколькими различными значениями дат, щелкая на Показать запись.



21. Проинспектируйте написанный код программы.

## **ПРАКТИЧЕСКАЯ РАБОТА № 26**

### **Тема: Инспекция кода модулей проекта**

**Цель работы:** научиться выполнять проверку кода модулей проекта на языке C# с целью выявления возможных ошибок, улучшения читаемости и соблюдения стандартов кодирования.

**Оборудование:** ПК, программное обеспечение – Visual Studio, MS Word, инструкции по выполнению работы.

#### **Содержание работы:**

##### **Задание 1.** Анализ структуры и читабельности кода модуля

1. Изучите модуль проекта, написанный в Практической работе № 1, определите его структуру, наличие комментариев, соблюдение стандартов оформления кода.

1.1. Откройте исходный код модуля в редакторе Visual Studio

1.2. Проанализируйте структуру класса/методов: наличие разделения на логические части, использование областей (regions).

1.3. Проверьте наличие комментариев к классам, методам и важным участкам кода.

1.4. Оцените читаемость кода: правильность именования переменных и методов, использование отступов.

1.5. Составьте краткий отчёт о выявленных достоинствах и недостатках.

##### **Задание 2.** Проверка соблюдения стандартов кодирования и исправление ошибок

1. Проверьте исходный код программы из Практической работы № 1 на соответствие стандартам кодирования, например, на использование PascalCase для методов, camelCase для переменных, отсутствие магических чисел.

1.1. Используйте встроенные или сторонние инструменты для анализа кода, например Visual Studio Code Analysis, Resharper, StyleCop.

1.2. Обратить внимание на: названия переменных, методов, классов; использование констант вместо магических чисел; стиль оформления кода.

1.3. Исправить обнаруженные несоответствия.

1.4. Зафиксировать изменённый код и подготовить отчёт о внесённых исправлениях.



## **ПРАКТИЧЕСКАЯ РАБОТА № 27**

### **Тема: Инспекция кода модулей проекта**

**Цель работы:** научиться выполнять проверку кода модулей проекта на языке C# с целью выявления возможных ошибок, улучшения читаемости и соблюдения стандартов кодирования.

**Оборудование:** ПК, программное обеспечение – Visual Studio, MS Word, инструкции по выполнению работы.

#### **Содержание работы:**

##### **Задание 1.** Анализ структуры и читабельности кода модуля

1. Изучите модуль проекта, написанный в Практической работе № 3, определите его структуру, наличие комментариев, соблюдение стандартов оформления кода.

1.1. Откройте исходный код модуля в редакторе Visual Studio

1.2. Проанализируйте структуру класса/методов: наличие разделения на логические части, использование областей (regions).

1.3. Проверьте наличие комментариев к классам, методам и важным участкам кода.

1.4. Оцените читаемость кода: правильность именования переменных и методов, использование отступов.

1.5. Составьте краткий отчёт о выявленных достоинствах и недостатках.

##### **Задание 2.** Проверка соблюдения стандартов кодирования и исправление ошибок

1. Проверьте исходный код программы из Практической работы № 3 на соответствие стандартам кодирования, например, на использование PascalCase для методов, camelCase для переменных, отсутствие магических чисел.

1.1. Используйте встроенные или сторонние инструменты для анализа кода, например Visual Studio Code Analysis, Resharper, StyleCop.

1.2. Обратите внимание на: названия переменных, методов, классов; использование констант вместо магических чисел; стиль оформления кода.

1.3. Исправить обнаруженные несоответствия.

1.4. Зафиксировать изменённый код и подготовить отчёт о внесённых исправлениях.

## **ПРАКТИЧЕСКАЯ РАБОТА № 28**

### **Тема: Тестирование интерфейса пользователя средствами инструментальной среды разработки**

**Цель работы:** изучения процесса тестирования интерфейса пользователя различными методами.

**Оборудование:** ПК, программное обеспечение –MS Visual Studio, MS Word, инструкции по выполнению работы.

#### **Справочный материал:**

Графический интерфейс пользователя (Graphical user interface, GUI) – разновидность интерфейса обеспечивающее взаимодействие через графические элементы (меню, кнопки, значки, списки и т. п.).

Варианты реализации GUI: интерфейс настольного приложения; мобильный интерфейс; веб-интерфейс.

#### **Цели тестирования:**

- обеспечение качественного взаимодействия с пользователем;
- выявление ошибок функциональности;
- выявление необработанных исключений при взаимодействии с интерфейсом;
- выявление потери или искажения данных, передаваемых через элементы интерфейса;
- выявление ошибки в интерфейсе (несоответствие проектной документации, отсутствие элементов интерфейса).

#### **Особенности тестирования:**

- тест планы в виде сценариев работы пользователя;
- сценарии на естественном языке или в виде скриптов;
- выполнение тестов в ручном режиме или с помощью эмулятора;
- анализ экранных форм и видимых элементов, а не внутренних переменных.

Покрытие – участие интерфейсных элементов в тестах.

Найденный дефект: несоответствие реального поведения требованиям или проблемы в требованиях.

Проблема отчета об ошибке: расплывчатость формулировок.

#### **Типы требований к UI:**

- требования к внешнему виду пользовательского интерфейса и формам взаимодействия с пользователем;
- требования к размещению элементов управления на экранных формах;
- требования к содержанию и оформлению выводимых сообщений;
- требования к форматам ввода;
- требования по доступу к внутренней функциональности системы при помощи пользовательского интерфейса;
- требования к реакции системы на ввод пользователя;
- требования к времени отклика на команды пользователя.

#### **Функциональное тестирование UI:**

- анализ требований к пользовательскому интерфейсу;
- разработка тест-требований и тест-планов для проверки пользовательского интерфейса;

- выполнение тестовых примеров и сбор информации о выполнении тестов;
- определение полноты покрытия пользовательского интерфейса требованиями;
- составление отчетов о проблемах в случае не совпадения поведения системы и требований либо в случае отсутствия требований на отдельные интерфейсные элементы.

Оценка покрытия UI. Функциональное покрытие – покрытие требований к пользовательскому интерфейсу. Структурное покрытие – для обеспечения полного структурного покрытия каждый интерфейсный элемент должен быть использован в тестовых примерах хотя бы один раз. Структурное покрытие с учетом состояния элементов интерфейса и внутреннего состояния системы – поведение некоторых интерфейсных элементов может изменяться в зависимости от внутреннего состояния системы. Каждое такое различимое поведение интерфейсного элемента должно быть проверено.

Ручное тестирование. Плюсы:

- контроль корректности проводится человеком;
- поиск «косметических» дефектов;
- анализ успешности прохождения теста будет выполняться не по формальным признакам, а согласно человеческому восприятию.

Минусы:

- требуются значительные человеческие и временные ресурсы;
- при проведении регрессионного тестирования и вообще любого повторного тестирования – на каждой итерации повторного тестирования пользовательского интерфейса требуется участие тестировщика-оператора.

Автоматическое тестирование. Плюсы:

- снижение стоимости тестирования;
- высокая скорость выполнения;
- большой объем покрытия;
- не требуется участие оператора-тестировщика при проведении регрессионного тестирования или любого другого повторного тестирования продукта.

Минусы:

- анализ успешности прохождения теста будет выполняться по формальным признакам;
- невозможность поиска «косметических» дефектов;
- высокая стоимость поддержки по сравнению с «обычными» функциональными тестами.

**Содержание работы:**

**Задание 1.** Протестируйте написанное приложение Аптечный пункт из Практической работы № 16. Составьте отчет по тестированию.

Отчет включает в себя:

1. Описание этапов тестирования
2. Результаты тестов

3.Отчёты о проблемах в случае несовпадения поведения системы и требований либо в случае отсутствия требований на отдельные интерфейсные элементы

**Оформление отчёта** может включать графический материал в виде рисунков, схем, таблиц.

## **ПРАКТИЧЕСКАЯ РАБОТА № 29**

**Тема:** Тестирование интерфейса пользователя средствами  
инструментальной среды разработки

**Цель работы:** изучения процесса тестирования интерфейса пользователя различными методами.

**Оборудование:** ПК, программное обеспечение –MS Visual Studio, MS Word, инструкции по выполнению работы.

**Содержание работы:**

**Задание 1.** Протестируйте написанное приложение Автовокзал из Практической работы № 17. Составьте отчет по тестированию.

Отчет включает в себя:

- 1.Описание этапов тестирования
- 2.Результаты тестов
- 3.Отчёты о проблемах в случае несовпадения поведения системы и требований либо в случае отсутствия требований на отдельные интерфейсные элементы

**Оформление отчёта** может включать графический материал в виде рисунков, схем, таблиц.

## ПРАКТИЧЕСКАЯ РАБОТА № 30

### Тема: Тестирование интерфейса пользователя средствами инструментальной среды разработки

**Цель работы:** изучения процесса тестирования интерфейса пользователя различными методами.

**Оборудование:** ПК, программное обеспечение –MS Visual Studio, MS Word, инструкции по выполнению работы.

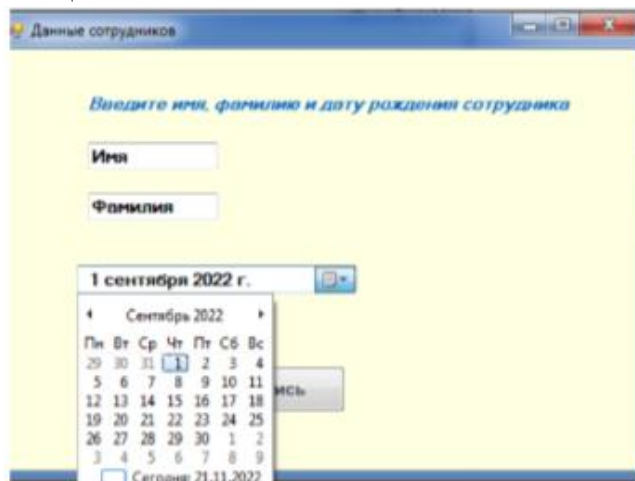
#### Содержание работы:

**Задание 1.** Протестируйте написанное приложение из Практической работы № 25. Составьте отчет по тестированию.

Отчет включает в себя:

- 1.Описание этапов тестирования
- 2.Результаты тестов
- 3.Отчёты о проблемах в случае несовпадения поведения системы и требований либо в случае отсутствия требований на отдельные интерфейсные элементы

**Оформление отчёта** может включать графический материал в виде рисунков, схем, таблиц.



## ПРАКТИЧЕСКАЯ РАБОТА № 31

### Тема: Разработка тестовых модулей проекта для тестирования отдельных модулей

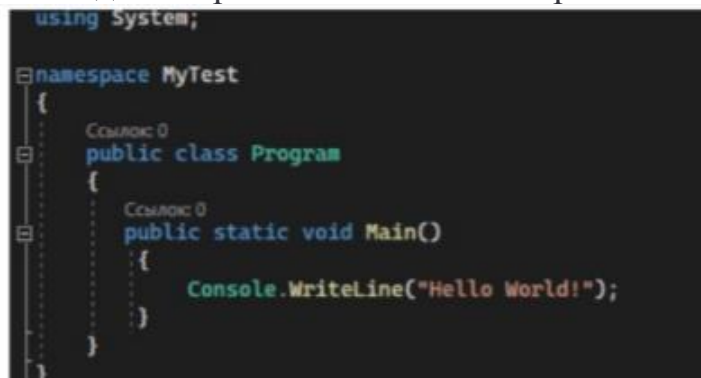
**Цель работы:** научиться выполнять модульное тестирование программного продукта средствами Visual Studio.

**Оборудование:** ПК, программное обеспечение – Visual Studio, MS Word, инструкции по выполнению работы.

#### Содержание работы:

##### Задание 1. Создать модульные тесты

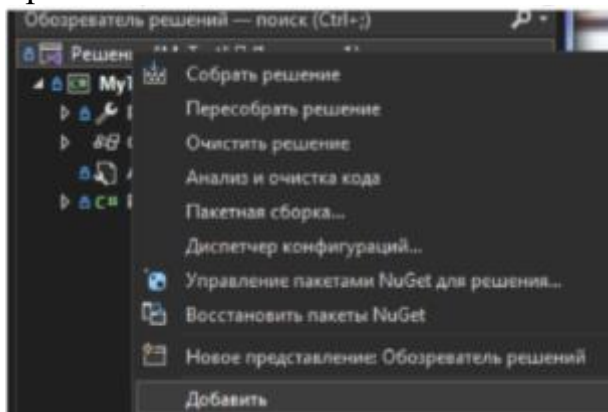
1. Запустите Visual Studio
2. Создайте простой консольный проект C# с именем MyTest.



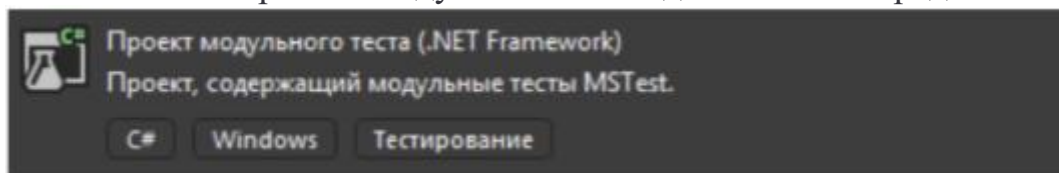
```
using System;

namespace MyTest
{
    // Ссылка 0
    public class Program
    {
        // Ссылка 0
        public static void Main()
        {
            Console.WriteLine("Hello World!");
        }
    }
}
```

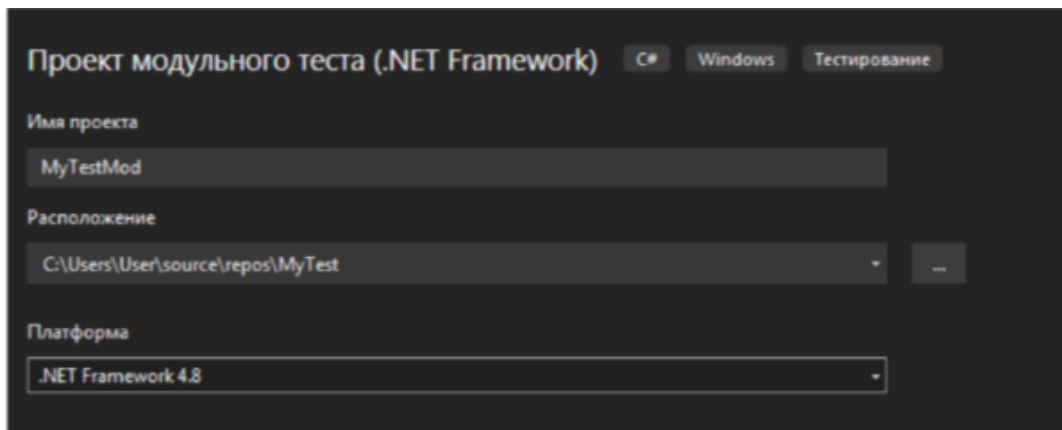
3. Выберите узел решения в Обозревателе решений. Затем с помощью щелкнув правой кнопкой мыши активизируйте команду Добавить - Создать проект...



4. В диалоговом окне нового проекта найдите проект модульного теста, который хотите использовать. Введите Тестирование в поле поиска, чтобы найти шаблон проекта модульного теста для тестовой среды C#.

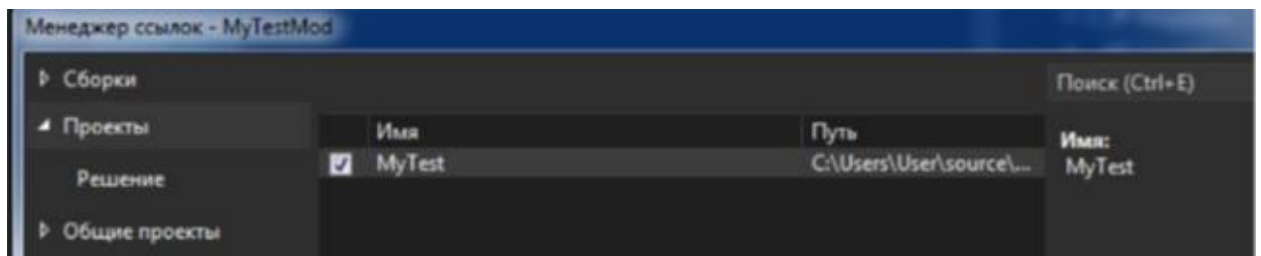


5. Нажмите Далее, выберите имя для тестового проекта и нажмите Создать.



6. В проекте модульного тестирования добавьте ссылку на проект, который вы хотите протестировать, щелкнув правой кнопкой мыши Ссылки или Зависимости, после чего выбрав Добавить ссылку или Добавить ссылку на проект.

7. Выберите проект, содержащий код, который будет тестироваться, и нажмите ОК.



8. Добавьте код в метод модульных тестов.

```
using Microsoft.VisualStudio.TestTools.UnitTesting;
using System;
using System.IO;

namespace MyTestMod
{
    [TestClass]
    Ссылка 0
    public class UnitTest1
    {
        public const string Expected = "Hello World!";

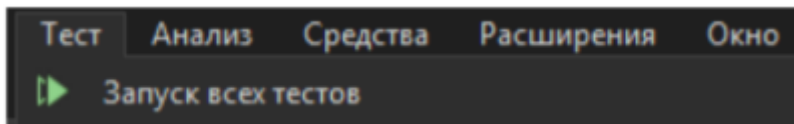
        [TestMethod]
        Ссылка 0
        public void TestMethod1()
        {
            using (var sw = new StringWriter())
            {
                Console.SetOut(sw);
                MyTest.program.Main();



                var result = sw.ToString().Trim();
                Assert.AreEqual(Expected, result);
            }
        }
    }
}
```

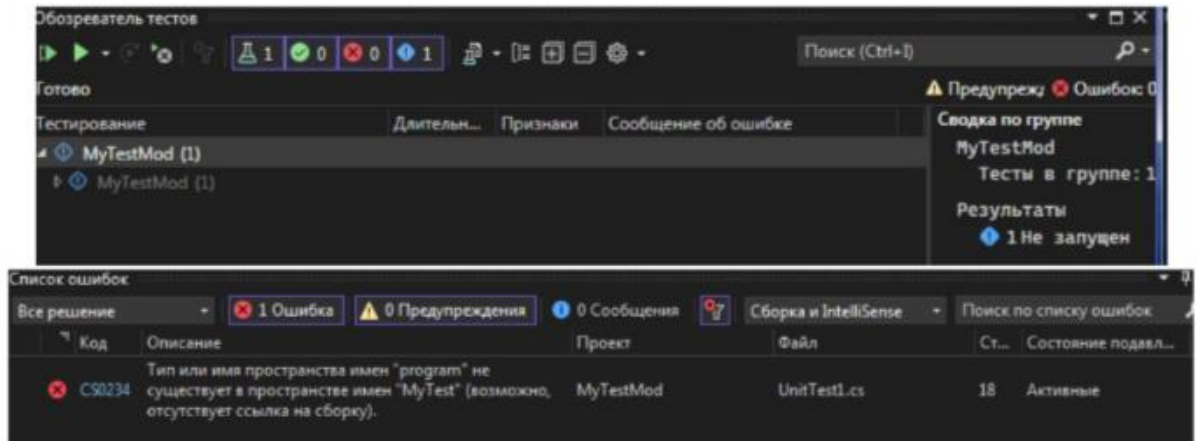
## Задание 2. Запуск модульного теста

1. Запустите модульные тесты, выбрав Тест - Запуск всех тестов





Иконка  показывает, что тест не запускался. Красный значок  указывает на сбой теста. В нижней части обозревателя теста должны выводиться ошибки в проекте.




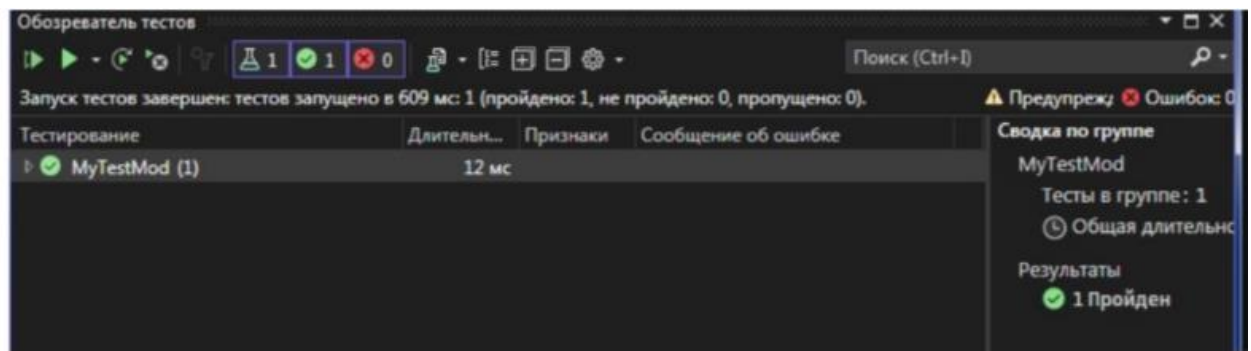
2. Исправьте ошибки и снова запустите тест.

```

1  using Microsoft.VisualStudio.TestTools.UnitTesting;
2  using System;
3  using System.IO;
4
5  namespace MyTestMod
6  {
7      [TestClass]
8      public class UnitTest1
9      {
10         public const string Expected = "Hello World!";
11
12         [TestMethod]
13         public void TestMethod1()
14         {
15             using (var sw = new StringWriter())
16             {
17                 Console.SetOut(sw);
18                 MyTest.Program.Main();
19
20                 var result = sw.ToString().Trim();
21                 Assert.AreEqual(Expected, result);
22             }
23         }
24     }
25 }
26

```

3. После завершения зеленый флажок  указывает, что тест пройден. В нижней части обозревателя теста не должно выводиться никаких ошибок.



4. После успешного завершения теста посмотрите время его исполнения (в нашем случае это 12 мс).

## ПРАКТИЧЕСКАЯ РАБОТА № 32

### Тема: Разработка тестовых модулей проекта для тестирования отдельных модулей

**Цель работы:** научиться выполнять модульное тестирование программного продукта средствами Visual Studio.

**Оборудование:** ПК, программное обеспечение – Visual Studio, MS Word, инструкции по выполнению работы.

#### **Справочный материал:**

*Модульные тесты* (Unit Tests) — это метод тестирования программного обеспечения, при котором отдельные модули или компоненты программы проверяются на корректность работы. Модули могут быть отдельными функциями, методами или классами, которые выполняют определённые задачи в рамках программы.

#### Основные принципы модульных тестов:

- Изоляция — каждый тест проверяет только один модуль.
- Повторяемость — тесты должны быть детерминированы, при одинаковых условиях они должны давать одинаковый результат.
- Быстрота — модульные тесты должны выполняться быстро, чтобы их можно было запускать часто.
- Точное определение проблемы — если тест не прошёл, это указывает на конкретный модуль, который нуждается в исправлении.

#### Цели

- Выявление ошибок на ранних стадиях разработки — это снижает затраты на их исправление.
- Упрощение отладки — при обнаружении ошибки в модульном тесте разработчик может легко локализовать проблему и быстро её исправить.
- Документация кода — модульные тесты служат дополнительной документацией, показывая, как должен работать код.
- Повышение уверенности в коде — автоматические тесты позволяют быть уверенным в том, что изменения в коде не нарушат его работу.

#### **Содержание работы:**

**Задание 1.** Создать 5 модульных теста для проверки программ, написанные в Практической работе № 2

## **ПРАКТИЧЕСКАЯ РАБОТА № 33**

### **Тема: Разработка тестовых модулей проекта для тестирования отдельных модулей**

**Цель работы:** научиться выполнять модульное тестирование программного продукта средствами Visual Studio.

**Оборудование:** ПК, программное обеспечение – Visual Studio, MS Word, инструкции по выполнению работы.

#### **Справочный материал:**

Модульное тестирование можно проводить вручную или с помощью специальных инструментов:

- Ручное — разработчик сам пишет тесты и выполняет их.
- Автоматизированное — используются специальные инструменты, которые генерируют тесты и выполняют их автоматически.

#### Этапы модульного тестирования:

1. Выбор модулей для тестирования — обычно тестируются все модули, которые могут быть изолированы от других модулей.
2. Разработка тестовых случаев — для каждого модуля разрабатывают набор тестовых случаев, которые проверяют все возможные сценарии использования модуля.
3. Выполнение тестов — тесты выполняются в среде разработки. Если тест не пройден, необходимо найти и устранить ошибку в коде.

#### Методы

- Тестирование по принципу белого и чёрного ящика:
  - Принцип белого ящика — тестировщик знает, как устроен модуль, что помогает создавать подробные тесты, проверяющие все возможные варианты выполнения кода.
  - Принцип чёрного ящика — тестировщик не знает, как устроен модуль, тесты создают на основе спецификаций и требований к модулю.
- Анализ покрытия кода — показывает, какие части исходного кода были выполнены (или «покрыты») во время тестов. Этот подход выявляет участки, которые остались непроверенными, и помогает найти участки кода, которые никогда не выполняются.

#### **Содержание работы:**

**Задание 1.** Создать 5 модульных теста для проверки программ, написанные в Практической работе № 3

## ПРАКТИЧЕСКАЯ РАБОТА № 34

### Тема: Выполнение функционального тестирования

**Цель работы:** получить практические навыки разработки тестов на основе внешней спецификации программы.

**Оборудование:** ПК, программное обеспечение – Visual Studio, MS Word, инструкции по выполнению работы.

#### Справочный материал:

Программа в случае тестирования с управлением по данным рассматривается как "черный ящик", и целью тестирования является выяснение обстоятельств, в которых поведение программы не соответствует спецификации. Различают следующие методы формирования тестовых наборов: эквивалентное разбиение; анализ граничных значений; анализ причинно-следственных связей; предположение об ошибке.

#### Эквивалентное разбиение.

Область всех возможных наборов входных данных программы по каждому параметру разбивают на конечное число групп - *классов эквивалентности*. Для составления классов эквивалентности нужно перебрать ограничения, установленные для каждого входного значения в техническом задании или при уточнении спецификации. Каждое ограничение разбивают на две или более групп.

#### Граничные значения.

Граничные значения – это значения на границах классов эквивалентности входных значений или около них.

#### Анализ причинно-следственных связей.

Метод *анализа причинно-следственных связей* позволяет системно выбирать тесты, используя алгебру логики. *Причиной* называют отдельное входное условие или класс эквивалентности. *Следствием* – выходное условие или преобразование системы. Идея заключается в отнесении всех следствий к причинам, то есть в уточнении причинно-следственных связей.

#### Предположение об ошибке.

Метод основан на интуиции программиста с большим опытом работы. Составляется список, в котором перечисляются возможные ошибки или ситуации, в которых они могут появиться, а затем на основе списка составляются тесты.

#### Содержание работы:

##### Задание.

1. На основе внешней спецификации задачи Практического занятия №1 составить набор тестов на основе подхода «черного ящика».
2. Провести тестирование программы.
3. Оформить отчет по практической работе.

##### Отчет по практической работе должен включать:

1. Внешнюю спецификацию.
2. Алгоритм решения задачи.
3. Текст программы на языке программирования.
4. Набор тестов на основе подхода «черного ящика» для отладки программы.

## ПРАКТИЧЕСКАЯ РАБОТА № 35

### Тема: Выполнение функционального тестирования

**Цель работы:** получить практические навыки разработки тестов на основе внешней спецификации программы.

**Оборудование:** ПК, программное обеспечение – Visual Studio, MS Word, инструкции по выполнению работы.

#### Содержание работы:

**Задание 1.** Реализуйте функциональное тестирование способом разбиения по эквивалентности

1. Определите входные данные
2. Определите выходные данные
3. Сформулируйте классы эквивалентности исходных данных алгоритма задачи. Результат занести в таблицу

Входное условие	Правильные классы эквивалентности	Неправильные классы эквивалентности

4. Разработайте тестовый вариант для каждого класса эквивалентности

#### Варианты задания:

1. Определить размер аванса, выплачиваемого каждому работнику подразделения с табельными номерами 11 по 51
2. Протестировать поле даты в форме.
3. Программа проверяет пароль пользователя. Он должен быть 8 символов и содержать заглавные и прописные латинские буквы, цифры.
4. В приложении Microsoft Paint есть опция «Изменить размер» — «Наклон», которая принимает значения -89... 89 Составьте классы эквивалентности.
5. Протестировать поля логин и пароль при регистрации пользователя, если граница 8
6. Протестировать поля логин и пароль при авторизации пользователя, если логин – test, пароль – 1234
7. Протестировать поле для ввода денежных средств.
8. Протестировать отправку письма в смартфоне.
9. Определить период отпусков работников подразделения с табельными номерами 25..100.
10. Протестировать текстовое поле в форме.

## **ПРАКТИЧЕСКАЯ РАБОТА № 36**

### **Тема: Выполнение функционального тестирования**

**Цель работы:** получить практические навыки разработки тестов на основе внешней спецификации программы.

**Оборудование:** ПК, программное обеспечение – Visual Studio, MS Word, инструкции по выполнению работы.

#### **Содержание работы:**

**Задание 1.** Реализуйте функциональное тестирование способом анализа граничных значений.

Реализуйте анализ граничных значений исходных данных поставленной задачи:

- определите минимальное граничное значение;
- определите максимальное граничное значение;
- определите допустимые входные данные;
- определите недопустимые входные данные;
- составьте тестовые варианты.

#### **Варианты задания:**

1. Система просит пользователя ввести возраст. В зависимости от того, является ли пользователь совершеннолетним или нет, отображается различный контент.
2. Протестировать текстовое поле «Имя», которое принимает длину от 6 до 12 символов.
3. Протестировать поле «Электронная почта» в форме «Гостевая книга» веб-сайта.
4. Онлайн-продажа билетов в кинотеатр на фильмы с рейтингом «18+».
5. Даны числа a, b, c, d. Требуется упорядочить их по возрастанию.
6. Даны числа a, b, c, d. Требуется упорядочить их по убыванию.
7. Протестировать функцию на интервале  $[-10; 15]$  с шагом 0,2.
8. Проверить дату как дату формата дд.мм.гггг.
9. Проверить услуги страховой компании: клиент может застраховать жизнь при возрасте 18 до 60 лет.
10. При нажатии на кнопку «Загрузить», система должна проверять размер загружаемого файла до 100 Мб и в соответствии с этим отправлять на определенный ftp-сервер.

**Задание 2.** Реализуйте функциональное тестирование способом таблицы решений

1. Сформулируйте классы эквивалентности исходных данных алгоритма задачи
2. Разработайте тестовый вариант для каждого класса эквивалентности
3. Составьте программу и протестируйте ее
4. Выработайте рекомендации для корректировки тестируемой программы

#### **Варианты задания:**

1. Найдите произведение нечетных чисел от –18 до 18

2. Вычислите функцию  $y=\ln(x+1)$ , для каждого  $x$  из  $[0, 10]$  изменяющегося с шагом 1
3. Найдите произведение чисел от -25 до 30, кратных 2. Исключить 0.



## ПРАКТИЧЕСКАЯ РАБОТА № 37

### Тема: Тестирование интеграции

**Цель работы:** получить практические навыки проведения интеграционного тестирования приложений.

**Оборудование:** ПК, программное обеспечение –MS Word, инструкции по выполнению работы.

#### Содержание работы:

**Задание 1.** Реализовать интеграционное тестирования любого Интернет-магазина. Требуется проверить, что совместимость web-сайта со сторонними сервисами:

- работу сторонних модулей: оплата, шаринг (совместное, коллективное использование и потребление различных вещей и услуг), карты.
- рекламу (просмотр, переходы по рекламе, аналитика).
- метрики (переходы по страницам, показы элементов, клики).

1. Разработайте тест-план:

- описание приложения;
- список функций и описание тестируемой системы и ее компонент;
- объекты тестирования.

2. Составьте тест-кейс

3. Протестируйте web-сайт

Тест-кейс «Название сайта»

Номер теста	Цель теста	Описание теста	Ожидаемый результат
1			

4. Сделайте выводы и рекомендации.

## ПРАКТИЧЕСКАЯ РАБОТА № 38

### Тема: Тестирование интеграции

**Цель работы:** получить практические навыки проведения интеграционного тестирования приложений.

**Оборудование:** ПК, программное обеспечение –MS Word, инструкции по выполнению работы.

#### Содержание работы:

**Задание 1.** Реализовать интеграционное тестирования любого Web-сайта. Требуется проверить, что совместимость web-сайта со сторонними сервисами:

- работу сторонних модулей
- рекламу (просмотр, переходы по рекламе, аналитика).
- метрики (переходы по страницам, показы элементов, клики).

1. Разработайте тест-план:

- описание приложения;
- список функций и описание тестируемой системы и ее компонент;
- объекты тестирования.

2. Составьте тест-кейс

3. Протестируйте web-сайт

Тест-кейс «Название сайта»

Номер теста	Цель теста	Описание теста	Ожидаемый результат
1			

4. Сделайте выводы и рекомендации.

## ПРАКТИЧЕСКАЯ РАБОТА № 39

### Тема: Тестирование интеграции

**Цель работы:** получить практические навыки проведения интеграционного тестирования приложений.

**Оборудование:** ПК, программное обеспечение – MS Visual Studio, MS Word, инструкции по выполнению работы.

#### Содержание работы:

**Задание 1.** Реализовать интеграционное тестирования приложения Флаги из Практической работы № 20.

1. Разработайте тест-план:

- описание приложения;
- список функций и описание тестируемой системы и ее компонент;
- объекты тестирования.

2. Составьте тест-кейс

3. Протестируйте Приложение

Тест-кейс «Название модуля»

Номер теста	Цель теста	Описание теста	Ожидаемый результат
1			

4. Сделайте выводы и рекомендации.

## ПРАКТИЧЕСКАЯ РАБОТА № 40

### Тема: Документирование результатов тестирования

**Цель работы:** изучение методов документирования результатов тестирования, научиться составлять документацию по результатам тестирования.

**Оборудование:** ПК, программное обеспечение – Visual Studio, MS Word, инструкции по выполнению работы.

#### Справочный материал:

Основная задача документирования – оценить текущее или финальное качество проекта и принять (если необходимо) – соответствующие решения и меры.

Отчёт о результатах тестирования (test result report, TRR) – часть тестовой документации, включающая в себя описание процесса тестирования, суммарную информацию о протестированных за подотчётный период билдах, информацию о деятельности тестировщиков, а также некоторые статистические данные.

Структура отчёта о результатах тестирования:

1. Команда тестировщиков
2. Описание процесса тестирования (testing process description)
3. Краткое описание (summary)
4. Расписание (testing timetable)
5. Рекомендации (recommendations)
6. Статистика по ошибкам (bugs statistics)
7. Список новых ошибок (new bugs found)
8. Статистика по всем ошибкам (all bugs statistics)

#### Содержание работы:

**Задание 1.** Провести документирование результатов ручного тестирования web-приложения.

1. Запустить ранее созданное или тестируемое приложение.
2. Составить итоговый отчет по результатам тестирования приложения:
  - Указать общую информацию о тестируемом продукте:
    - название проекта;
    - модули, которые подверглись тестированию;
    - количество обнаруженных дефектов;
  - Указать, кто и когда тестировал программный продукт.
  - Описать тестовое окружение: ссылку на проект, браузер, операционную систему и другую информацию, конкретизирующую особенности конфигурации.
  - Указать общую оценку качества протестированного приложения и ее обосновать. Уровни качества: высокое (High), среднее (Medium), низкое (Low).
  - Определить показатель успешного прохождения тест-кейсов с помощью метрики по формуле

$$TSP = \frac{T_{Success}}{T_{Total}} \cdot 100 \%,$$

– Графически (в виде круговой диаграммы) отразить процентное соотношение дефектов GUI и функциональных дефектов:

#### 1. Реализовать GUI – тестирование пользовательского интерфейса

№	Тест-кейсы	Результаты теста
1	Проверить все элементы графического интерфейса на размер, положение, ширину, длину и прием символов или цифр.	успешно/дефект
2	Проверить, можете ли вы выполнить намеченную функциональность приложения, используя графический интерфейс	
3	Проверить сообщения об ошибках отображаются правильно	
4	Проверить на четкое разграничение различных разделов на экране	
5	Проверить, что используемый в приложении шрифт читабелен	
6	Проверить правильность выравнивания текста	
7	Цвет шрифта и предупреждающих сообщений эстетичны	
8	Убедиться, что изображения имеют хорошую четкость	
9	Убедиться, что изображения правильно выровнены	
10	Проверить расположение элементов графического интерфейса для разных разрешений экрана.	

- Вычислить процент дефектов GUI по формуле:

$$\text{Процент дефектов} = \frac{\text{Количества дефектов}}{\text{количество тестов}} * 100 \%$$

#### 2. Реализовать Functional - тестирование, основанное на анализе спецификации функциональности

При функциональном тестировании проверяется реализация функциональных требований, т.е. возможность программного продукта выполнять те функции, которые были описаны в спецификациях на разработку. Каждый блок в отдельности необходимо тестировать на корректность реализации присущих ему функций.

№	Тест-кейсы	Результаты теста
1	Проверка баннера	успешно/дефект
2	Поиск товара	
3	.....	

- Вычислить процент функциональных дефектов

– Графически (в виде столбчатой диаграммы) отразить распределение дефектов по модулям.

– Произвести детальный анализ качества всех модулей протестированного приложения с аргументацией выставленных уровней качества.

Модули	Уровень качества	Комментарии
Главная		
Каталог товаров		
Корзина		

– Привести список 3-5 наиболее критичных дефектов.

– Сформулировать рекомендации по улучшению качества программного продукта.

## ПРАКТИЧЕСКАЯ РАБОТА № 41

### Тема: Документирование результатов тестирования

**Цель работы:** изучение методов документирования результатов тестирования, научиться составлять документацию по результатам тестирования.

**Оборудование:** ПК, программное обеспечение – Visual Studio, MS Word, инструкции по выполнению работы.

#### Содержание работы:

**Задание 1.** Провести документирование результатов ручного тестирования web-сайта.

1. Запустить ранее созданное или тестируемое приложение.

2. Составить итоговый отчет по результатам тестирования приложения:

– Указать общую информацию о тестируемом продукте:

- название проекта;
- модули, которые подверглись тестированию;
- количество обнаруженных дефектов;

– Указать, кто и когда тестировал программный продукт.

– Описать тестовое окружение: ссылку на проект, браузер, операционную систему и другую информацию, конкретизирующую особенности конфигурации.

– Указать общую оценку качества протестированного приложения и ее обосновать. Уровни качества: высокое (High), среднее (Medium), низкое (Low).

– Определить показатель успешного прохождения тест-кейсов с помощью метрики по формуле

$$T^{SP} = \frac{T^{Success}}{T^{Total}} \cdot 100 \%,$$

– Графически (в виде круговой диаграммы) отразить процентное соотношение дефектов GUI и функциональных дефектов:

1. Реализовать GUI – тестирование пользовательского интерфейса

№	Тест-кейсы	Результаты теста
1	Проверить все элементы графического интерфейса на размер, положение, ширину, длину и прием символов или цифр.	успешно/дефект
2	Проверить, можете ли вы выполнить намеченную функциональность приложения, используя графический интерфейс	
3	Проверить сообщения об ошибках отображаются правильно	
4	Проверить на четкое разграничение различных разделов на экране	
5	Проверить, что используемый в приложении шрифт читабелен	
6	Проверить правильность выравнивания текста	
7	Цвет шрифта и предупреждающих сообщений эстетичны	
8	Убедиться, что изображения имеют хорошую четкость	
9	Убедиться, что изображения правильно выровнены	
10	Проверить расположение элементов графического интерфейса для разных разрешений экрана.	

- Вычислить процент дефектов GUI по формуле:

$$\text{Процент дефектов} = \frac{\text{Количества дефектов}}{\text{количество тестов}} * 100 \%$$

2. Реализовать Functional - тестирование, основанное на анализе спецификации функциональности

При функциональном тестировании проверяется реализация функциональных требований, т.е. возможность программного продукта выполнять те функции, которые были описаны в спецификациях на разработку. Каждый блок в отдельности необходимо тестировать на корректность реализации присущих ему функций.

№	Тест-кейсы	Результаты теста
1	Проверка баннера	успешно/дефект
2	Поиск товара	
3	.....	

• Вычислить процент функциональных дефектов

– Графически (в виде столбчатой диаграммы) отразить распределение дефектов по модулям.

– Произвести детальный анализ качества всех модулей протестированного приложения с аргументацией выставленных уровней качества.

Модули	Уровень качества	Комментарии
Главная		
Каталог товаров		
Корзина		

– Привести список 3-5 наиболее критичных дефектов.

– Сформулировать рекомендации по улучшению качества программного продукта.

## ПРАКТИЧЕСКАЯ РАБОТА № 42

### Тема: Документирование результатов тестирования

**Цель работы:** изучение методов документирования результатов тестирования, научиться составлять документацию по результатам тестирования.

**Оборудование:** ПК, программное обеспечение – Visual Studio, MS Word, инструкции по выполнению работы.

#### Содержание работы:

**Задание 1.** Провести документирование результатов ручного тестирования приложения Флаги из Практической работы № 20.

1. Запустить ранее созданное или тестируемое приложение.

2. Составить итоговый отчет по результатам тестирования приложения:

– Указать общую информацию о тестируемом продукте:

- название проекта;
- модули, которые подверглись тестированию;
- количество обнаруженных дефектов;

– Указать, кто и когда тестировал программный продукт.

– Описать тестовое окружение: ссылку на проект, браузер, операционную систему и другую информацию, конкретизирующую особенности конфигурации.

– Указать общую оценку качества протестированного приложения и ее обосновать. Уровни качества: высокое (High), среднее (Medium), низкое (Low).

– Определить показатель успешного прохождения тест-кейсов с помощью метрики по формуле

$$T^{SP} = \frac{T^{Success}}{T^{Total}} \cdot 100 \%,$$

– Графически (в виде круговой диаграммы) отразить процентное соотношение дефектов GUI и функциональных дефектов:

1. Реализовать GUI – тестирование пользовательского интерфейса

№	Тест-кейсы	Результаты теста
1	Проверить все элементы графического интерфейса на размер, положение, ширину, длину и прием символов или цифр.	успешно/дефект
2	Проверить, можете ли вы выполнить намеченную функциональность приложения, используя графический интерфейс	
3	Проверить сообщения об ошибках отображаются правильно	
4	Проверить на четкое разграничение различных разделов на экране	
5	Проверить, что используемый в приложении шрифт читабелен	
6	Проверить правильность выравнивания текста	
7	Цвет шрифта и предупреждающих сообщений эстетичны	
8	Убедиться, что изображения имеют хорошую четкость	
9	Убедиться, что изображения правильно выровнены	
10	Проверить расположение элементов графического интерфейса для разных разрешений экрана.	

- Вычислить процент дефектов GUI по формуле:



$$\text{Процент дефектов} = \frac{\text{Количества дефектов}}{\text{количество тестов}} * 100 \%$$

2. Реализовать Functional - тестирование, основанное на анализе спецификации функциональности

При функциональном тестировании проверяется реализация функциональных требований, т.е. возможность программного продукта выполнять те функции, которые были описаны в спецификациях на разработку. Каждый блок в отдельности необходимо тестировать на корректность реализации присущих ему функций.

№	Тест-кейсы	Результаты теста
1	Проверка баннера	успешно/дефект
2	Поиск товара	
3	.....	

- Вычислить процент функциональных дефектов
  - Графически (в виде столбчатой диаграммы) отразить распределение дефектов по модулям.
  - Произвести детальный анализ качества всех модулей протестированного приложения с аргументацией выставленных уровней качества.
  - Привести список 3-5 наиболее критичных дефектов.
  - Сформулировать рекомендации по улучшению качества программного продукта.

## **Информационное обеспечение обучения**

### **Печатные и электронные издания:**

#### **Основные учебные издания:**

1. Долженко, А. И. Технологии командной разработки программного обеспечения информационных систем: учебное пособие / А. И. Долженко. — 4-е изд. — Москва: Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2024. — 300 с. — ISBN 978-5-4497-2486-1. — Текст: электронный // Электронный ресурс цифровой образовательной среды СПО PROОбразование: [сайт]. — URL: <https://profspo.ru/books/133985>
2. Емелина, Е. И., Поддержка и тестирование программных модулей: учебник / Е. И. Емелина. — Москва: КноРус, 2025. — 267 с. — ISBN 978-5-406-14483-1. — URL: <https://book.ru/book/957274>
3. Рощин, П. Г. Командная разработка программного обеспечения с помощью системы контроля версий GIT: конспект лекций: учебное пособие / П. Г. Рощин. — Москва: Национальный исследовательский ядерный университет «МИФИ», 2022. — 106 с. — ISBN 978-5-7262-2846-4. — Текст: электронный // Электронный ресурс цифровой образовательной среды СПО PROОбразование: [сайт]. — URL: <https://profspo.ru/books/132682>

#### **Интернет-ресурсы:**

##### **Электронно-библиотечная система:**

4. ЭБС «Znanium»
5. ЭБС «PROОбразование»
6. ЭБС «Book.ru»