

**Учреждение профессионального образования
«Колледж Казанского инновационного университета»
Альметьевский филиал**

УТВЕРЖДЕН
в составе Основной образовательной программы –
программы подготовки специалистов среднего звена
протокол № 1 от «22» января 2025 г.

Фонд оценочных средств по дисциплине

ОП.17 ПРАКТИКУМ ПО РАЗРАБОТКЕ КОМПЬЮТЕРНЫХ СИСТЕМ

программы подготовки специалистов среднего звена

по специальности

09.02.07 Информационные системы и программирование
(на базе основного общего образования)

Срок получения СПО по ППССЗ – 3 г.10 мес.

Форма обучения - очная

Присваиваемая квалификация
Программист

Альметьевск 2025

Фонд оценочных средств по дисциплине ОП17 «Практикум по разработке компьютерных систем» программы подготовки специалистов среднего звена по специальности 09.02.07 Информационные системы и программирование разработан на основе рабочей программы учебной дисциплины

1. Общие положения

Фонд оценочных средства (ФОС) предназначены для контроля и оценки образовательных достижений обучающихся, освоивших программу дисциплины «Практикум по разработке компьютерных систем».

ФОС включают контрольные материалы для проведения текущего и итогового контроля.

ФОС разработаны на основании положений:

- программы подготовки специалистов среднего звена по специальности 09.02.07

Информационные системы и программирование

- программы учебной дисциплины «Практикум по разработке компьютерных систем»

2. Показатели оценки результатов освоения дисциплин, формы и методы контроля и оценки

В результате контроля и оценки по дисциплине осуществляется комплексная проверка следующих умений (У) и знаний (З):

Результаты обучения (освоенные умения, усвоенные знания, практический опыт)	Основные показатели оценки результатов
В результате освоения дисциплины обучающийся должен знать: - актуальный профессиональный и социальный контекст, в котором приходится работать и жить; основные источники информации и ресурсы для решения задач и проблем в профессиональном и/или социальном контексте; - основные принципы технологии структурного и объектноориентированного программирования; - основные принципы отладки и тестирования программных продуктов; - методы и средства разработки технической документации. - методы описания схем баз данных в современных СУБД; - основные положения теории баз данных, хранилищ данных, баз знаний; - основные принципы структуризации и нормализации базы данных; - основные принципы построения концептуальной, логической и физической модели данных; - Основные принципы структуризации и нормализации базы данных. - структуры данных СУБД, общий подход к организации представлений, таблиц, индексов и кластеров; - методы описания схем баз данных в современных СУБД; - структуры данных СУБД, общий подход к организации представлений, таблиц, индексов и кластеров; - методы организации целостности данных; - основные принципы структуризации и нормализации базы данных; - основные принципы построения концептуальной, логической и физической модели данных; - технологии передачи и обмена данными в	- знает и понимает современные сетевые технологии, стандарты и протоколы; на профессиональном уровне владеет терминологией; знает архитектуры компьютерных сетей; - знает как осуществляется настройка сетевого оборудования, диагностика неисправностей; - владеет информацией об оптимизации производительности; - знает механизм работы с документацией; - знает профессиональные базы данных, техническую документацию; - на профессиональном уровне знает этапы построения плана, определение приоритетов, оценка ресурсов, контроль выполнения; - знает механизм работы с программным обеспечением, сетевым оборудованием, использования инструментов диагностики; - знает как применять системы мониторинга; - знает работу настройки сетевых служб, с системами виртуализации; использования анализаторов и применение средств автоматизации; - знает порядок разрешения конфликтов в коллективе; - анализирует коммуникативные навыки членов коллектива; - знает основы проектной деятельности; - на профессиональном уровне знает этапы построения технических текстов, профессиональную терминологию; - знает основные положения теории баз

компьютерных сетях; - алгоритм проведения процедуры резервного копирования; - алгоритм проведения процедуры восстановления базы данных; - методы организации целостности данных; - способы контроля доступа к данным и управления привилегиями; - основы разработки приложений баз данных; - основные методы и средства защиты данных в базе данных	данных, принципы структуризации и нормализации баз данных, методы построения концептуальных, логических и физических моделей данных, структуры данных в системах управления базами данных, принципы организации представлений, таблиц, индексов и кластеров; - основные положения теории баз данных и хранилищ данных; принципы построения концептуальной модели; методы создания логической модели данных, основы физической модели данных, современные подходы к проектированию баз данных; - знаете правильность определения типов данных для реквизитов; корректность выделения ключевых реквизитов, точность определения связей между объектами, эффективность организации индексов; - знает принципы организации и функционирования СУБД; методы обеспечения безопасности данных; алгоритмы резервного копирования и восстановления; принципы оптимизации производительности; методы мониторинга работы СУБД; - знает архитектуру выбранной СУБД, модели данных и их реализация, принципы обработки транзакций, механизмы блокировки и синхронизации, методы обеспечения целостности данных; - знает сущность и понятие информационной безопасности; характеристику составляющих информационной безопасности, место информационной безопасности в системе национальной безопасности, виды угроз информационной безопасности, основные положения комплексного подхода к защите информации
В результате освоения дисциплины обучающийся должен уметь: - распознавать задачу и/или проблему в профессиональном и/или социальном контексте; анализировать задачу и/или проблему и выделять её составные части; определять этапы решения задачи; выявлять и эффективно искать информацию, необходимую для решения задачи и/или проблемы; - осуществлять разработку кода программного модуля на современных языках программирования; - создать программу по разработанному	- умеет точно определить сетевую проблему; владеет корректностью выделения составных частей проблемы; - умеет определять информационные потребности; анализировать существующие системы; формировать требования к БД; - на профессиональной уровне умеет создавать таблицы и их структуры; определять типы данных, настраивать ограничения целостности, формировать индексы, разрабатывать представления; - умеет устанавливать и настраивать СУБД, создавать и управлять базами данных,

алгоритму как отдельный модуль; - выполнять отладку и тестирование программы на уровне модуля; - оформлять документацию на программные средства; - использовать инструментальные средства для автоматизации оформления документации; - основные этапы разработки программного обеспечения; - выполнять стандартные процедуры резервного копирования и мониторинга выполнения этой процедуры; - выполнять процедуру восстановления базы данных и вести мониторинг выполнения этой процедуры; - выполнять установку и настройку программного обеспечения для обеспечения работы пользователя с базой данных; - обеспечивать информационную безопасность на уровне базы данных	настраивать параметры хранения, управлять пользователями и ролями, настраивать права доступа; - владеет следующими навыками: настройка шифрования данных, управление криптографическими ключами, реализация аутентификации, настройка авторизации, управление цифровыми сертификатами
В результате освоения дисциплины обучающийся должен получить практический опыт: - разработки алгоритма поставленной задачи и реализации его средствами автоматизированного проектирования; - разработки кода программного продукта на основе готовой спецификации на уровне модуля; - использования инструментальных средств на этапе отладки программного продукта; - проведения тестирования программного модуля по определенному сценарию; - выполнять сбор, обработку и анализ информации для проектирования баз данных; - выполнять работы с документами отраслевой направленности; - работать с объектами баз данных в конкретной системе управления базами данных; - использовать стандартные методы защиты объектов базы данных; - работать с документами отраслевой направленности; - использовать средства заполнения базы данных; - использовать стандартные методы защиты объектов базы данных; - работать с объектами базы данных в конкретной системе управления базами данных;	- выполнять сбор, обработку и анализ информации для проектирования баз данных; - выполнять работы с документами отраслевой направленности; - работать с объектами баз данных в конкретной системе управления базами данных; - использовать стандартные методы защиты объектов базы данных; - работать с документами отраслевой направленности; - использовать средства заполнения базы данных; - использовать стандартные методы защиты объектов базы данных; - работать с объектами базы данных в конкретной системе управления базами данных; - выполнять работы с объектами базы данных в конкретной системе

В рамках программы учебной дисциплины обучающиеся получают первоначальный практический опыт (ПО), продолжают развивать общие компетенции (ОК), приступают к освоению элементов профессиональных компетенций (ПК):

Коды ОК, ПК	Содержание общих компетенций и осваиваемые элементы профессиональных 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
ПК 1.1.	Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием
ПК 1.2.	Разрабатывать программные модули в соответствии с техническим заданием.
ПК 1.3.	Выполнять отладку программных модулей с использованием специализированных программных средств.
ПК 1.4.	Выполнять тестирование программных модулей
ПК 11.1.	Осуществлять сбор, обработку и анализ информации для проектирования баз данных
ПК 11.2.	Проектировать базу данных на основе анализа предметной области
ПК 11.3	Разрабатывать объекты базы данных в соответствии с результатами анализа предметной области

3. Распределение типов контрольных заданий по элементам знаний и умений

Основной целью оценки освоения дисциплины является оценка умений и знаний. Оценка освоения умений и знаний и опыта практической деятельности осуществляется с использованием следующих форм и методов контроля: выполнение практических работ, тестирование.

Распределение оценивания результатов обучения

Результаты освоения учебной дисциплины	Результаты освоения профессионального модуля направлены на формирование общих компетенций (ОК)	Формы и методы оценки
ПК 1.1. Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием Знания: Основные этапы разработки программного обеспечения Основные принципы технологии структурного программирования Основные принципы технологии объектно-ориентированного программирования Способы оптимизации программного кода Приемы рефакторинга Основные принципы отладки программных продуктов Основные принципы тестирования программных продуктов Умения: Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием Оценивать сложность алгоритмов Создавать программу по разработанному алгоритму как отдельный модуль Осуществлять разработку кода программного модуля на языках низкого уровня Осуществлять разработку кода программного модуля на языках высокого уровня Оформлять техническую документацию на программные средства Проводить анализ алгоритмов Практический опыт: Разработка алгоритмов решения поставленных задач Реализация алгоритмов средствами автоматизированного проектирования Разработка кода программного продукта на основе готовой спецификации на уровне модуля	ОК 01 ОК 02	- Практические работы - Тестирование - Оценка правильности заполнения документации

Использование инструментальных средств на этапе отладки программного продукта Проведение тестирования программного модуля по определенному сценарию Разработка мобильных приложений Создание программ по разработанным алгоритмам как отдельные модули Выполнение отладки и тестирования программ на уровне модуля Осуществление разработки кода программного модуля на современных языках программирования Выполнение оптимизации и рефакторинга программного кода Оформление документации на программные средства		
ПК 1.2. Разрабатывать программные модули в соответствии с техническим заданием. Знания: Основные этапы разработки программного обеспечения Принципы структурного программирования Принципы объектно-ориентированного программирования Методы разработки программного обеспечения Современные языки программирования Типовые архитектуры программных продуктов Принципы модульного программирования Стандарты оформления кода Основы тестирования программного обеспечения Принципы работы с системами контроля версий Умения: Разрабатывать программные модули на основе технического задания Создавать программный код с использованием современных языков программирования Применять принципы структурного программирования Применять принципы объектно-ориентированного программирования Выполнять модульное тестирование Работать с системами контроля версий Оформлять техническую документацию Оптимизировать программный код Выполнять рефакторинг кода Интегрировать разработанные модули в	ОК 01 ОК 02	- Практические работы - Тестирование - Оценка правильности заполнения документации

общую систему Практический опыт: Разработка программного кода на основе технического задания Создание программных модулей с использованием современных технологий разработки Отладка и тестирование разработанных модулей Документирование разработанных программных модулей Работа с системами контроля версий Оптимизация производительности программных модулей Рефакторинг существующего кода Интеграция разработанных модулей в общую систему Проведение модульного тестирования Решение проблем совместимости		
ПК 1.3. Выполнять отладку программных модулей с использованием специализированных программных средств Знания: Основные принципы отладки и тестирования программных продуктов Инструментарий отладки программных продуктов Типы и форматы сообщений об ошибках Способы использования технологических журналов Современные компиляторы, отладчики и оптимизаторы программного кода Методы и приемы отладки программного кода Принципы работы с системой контроля версий Форматы и типы записей журналов Способы интерпретации сообщений об ошибках и предупреждений Умения: Выявлять ошибки в программном коде Применять методы и приемы отладки программного кода Интерпретировать сообщения об ошибках и предупреждения Работать с технологическими журналами Применять современные компиляторы и оптимизаторы Использовать отладчики Работать с системой контроля версий Выполнять действия согласно регламенту используемой системы управления	ОК 01 ОК 02	- Практические работы - Тестирование - Оценка правильности заполнения документации

версиями Обрабатывать записи технологических журналов Практический опыт: Использование инструментальных средств на этапе отладки программного продукта Проведение тестирования программного модуля по определенному сценарию Отладка программного кода на уровне программных модулей Отладка программного кода на уровне межмодульных взаимодействий Отладка взаимодействий с окружением Регистрация изменений исходного текста программного кода Слияние, разделение и сравнение исходных текстов Сохранение изменений в соответствии с регламентом управления версиями Анализ и проверка исходного программного кода		
ПК 1.4. Выполнять тестирование программных модулей Знания: Основные виды и принципы тестирования программных продуктов Методы тестирования программного обеспечения Виды тестирования (юнит-тестирование, интеграционное, системное) Критерии входа и выхода тестирования Типы тестовых сценариев и тест-кейсов Метрики тестирования Инструменты автоматизации тестирования Стандарты оформления документации по тестированию Принципы работы с баг-трекингowymi системами Умения: Составлять тестовую документацию Разрабатывать тестовые сценарии Выполнять ручное тестирование Выполнять автоматизированное тестирование Работать с баг-трекингowymi системами Составлять отчеты о тестировании Оценивать риски и приоритеты тестирования Анализировать результаты тестирования Документировать найденные дефекты Практический опыт: Проведение тестирования программного	ОК 01 ОК 02	- Практические работы - Тестирование - Оценка правильности заполнения документации

модуля по определенному сценарию Использование инструментальных средств на этапе тестирования программного продукта Создание и выполнение тестовых сценариев Работа с системами управления тестированием Документирование результатов тестирования Взаимодействие с разработчиками по исправлению дефектов Оценка готовности продукта к выпуску Составление отчетов о покрытии тестами		
ПК 11.2. Проектировать базу данных на основе анализа предметной области Знания: Основные принципы структуризации и нормализации базы данных. Структуры данных СУБД, общий подход к организации представлений, таблиц, индексов и кластеров. Умения: Работать с современными case-средствами проектирования баз данных Практический опыт: Выполнять работы с документами отраслевой направленности.	OK 01 OK 02	- Практические работы - Тестирование - Оценка правильности заполнения документации
ПК 11.3. Разрабатывать объекты базы данных в соответствии с результатами анализа предметной области Знания: Методы описания схем баз данных в современных СУБД. Структуры данных СУБД, общий подход к организации представлений, таблиц, индексов и кластеров. Методы организации целостности данных. Умения: Работать с современными case-средствами проектирования баз данных. Создавать объекты баз данных в современных СУБД. Практический опыт: Работать с объектами баз данных в конкретной системе управления базами данных. Использовать стандартные методы защиты объектов базы данных. Работать с документами отраслевой направленности. Использовать средства заполнения базы данных. Использовать стандартные методы защиты объектов базы данных	OK 01 OK 02 OK 03 OK 04 OK 05 OK06 OK07 OK 09	- Практические работы - Тестирование - Оценка правильности заполнения документации

2. Задания для оценки освоения дисциплины

Тема 1. Разработка модулей программного обеспечения для компьютерных систем

Задание 1.1. Практическая работа «Составление спецификации для разработки программного модуля»

Проверяемые результаты обучения: ОК1, ОК2, ПК1.1, ПК1.2, ПК1.3, ПК1.4

Цель занятия: Получить практические навыки составления спецификации для разработки программного модуля

Краткие теоретические сведения

Понятие спецификации

Под спецификацией понимается формальное описание функций и данных программы, с которыми эти функции оперируют. Различают видимые данные, т.е. входные и выходные параметры, а также скрытые данные, которые не привязаны к реализации и определяют интерфейс с другими функциями.

Предусловия - это ограничения на совокупность входных параметров и постусловия - ограничения на выходные параметры.

Спецификация программы должна иметь точное, однозначное и недвусмысленное описание программы с помощью математических понятий, терминов, правил синтаксиса и семантики языка спецификации. В языке спецификаций могут быть понятия и конструкции, которые нельзя выполнить на компьютере, они представляются последовательностью операций, функций, понятных для интерпретации. Спецификация программы должна содержать информацию, необходимую для построения алгоритма программы.

Спецификация содержит набор требований для программы в целом, позволяет зафиксировать детальные требования к разработке, указать роли и ответственности сторон, сроки и стоимость реализации. Так вы сможете четко понимать, что и когда будет реализовано, и, в случае, разногласий, иметь письменное подтверждение договоренностей.

Набор требований предназначен для того, чтобы установить базу для соглашения между заказчиком и разработчиком (или подрядчиками) о том, как должен функционировать программный продукт

Выделяют следующие необходимые свойства набора требований:

- однократное упоминание отдельных требований;
- отсутствие пересечений между требованиями;
- явное указание связей между требованиями;
- полнота;
- непротиворечивость;
- определение ограничений, области действия и контекста для каждого требования;
- модифицируемость, конфигурируемость, удобство поддержки.

Методика составления спецификаций требований к программному обеспечению (IEEE 830-1998) определяет содержание и качественные характеристики правильно составленной спецификации требований

Примерный состав спецификации

- Наименование модуля
- Цель модуля
- Функциональные возможности модуля
- Взаимодействие с другими модулями
- Взаимодействие с пользователем
- Входные данные
- Выходные данные

- Навигация внутри модуля
- Требования к макету
- Обработка ошибок

Пример спецификации:

Наименование экранной формы: Главная страница приложения Язык разработки: C#

Среда разработки: MSVisualStudio 2015

Цель: Создать головной модуль, позволяющий осуществить переход к выполнению других модулей программного продукта.

Входные данные: нажатия на кнопки (элементы навигации модуля). Навигация:

- во время запуска программы пользователь попадает на форму с навигацией, с помощью которой можно получить информацию о программе, либо запустить соответствующие модули.

Выходные параметры: запуск новой формы программного продукта с новым модулем

Действия:

- выбрать нужный элемент навигации;
- нажать на нужный элемент навигации;
- запустить нужный модуль с помощью элемента навигации. Требования к макету страницы:

На форме должно быть три кнопки, которые находятся посередине формы.

Кнопки имеют название:

- Пройти тест;
- О программе;
- Выход;

Шрифт всех кнопок TimesNewRoman, размер 14 пт. Кнопки зеленого цвета. Основной цвет формы – серый.

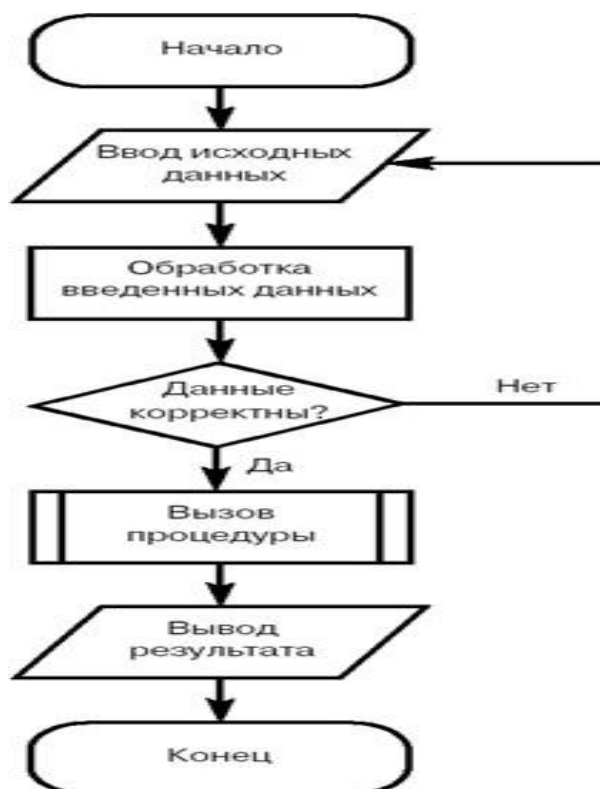
При нажатии на кнопку «Пройти тест», открывается модуль, с помощью которого необходимо зарегистрировать пользователя, после чего открывается новая форма с самим тестом.

При нажатии на кнопку «О программе», открывается новое окно с информацией о программе.

При нажатии на кнопку «Выход» все остальные окна формы закрываются.

Сообщения об ошибках: нет.

Для модуля чаще всего разрабатывается углубленная блок-схема



Задания

1. Изучить теоретические сведения и задание к работе
2. В соответствии с вариантом задания составить спецификацию и блок-схему модуля (MicrosoftVisio).

Порядок выполнения работы (Пример выполнения)

Исходные данные:

Есть массив целых чисел размером n . С клавиатуры вводится два числа

- порядковые номера элементов массива, которые необходимо суммировать.

Например, если ввели 3 и 5 - суммируются 3-й и 5-й элементы. Описать функцию целого типа, возвращающую сумму заданных элементов массива.

Решение

Наименование: Программа вычисления суммы двух заданных элементов массива

Язык разработки: C#

Среда разработки: MS Visual Studio 2015

Цель: Создать программу вычисления суммы заданных элементов массива.

Входные данные: Массив целых чисел, размер массива, номера элементов массива для суммирования.

Навигация: Во время запуска программы на экране для пользователя появляются указания и подсказки для ввода данных.

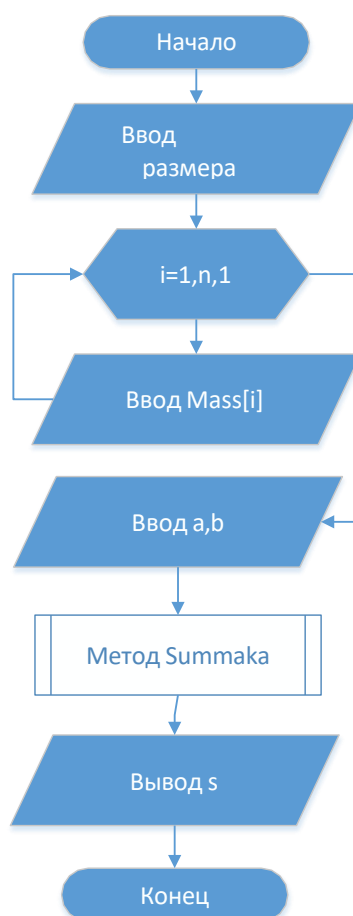
Выходные параметры: Результаты вычислений или сообщения об ошибках

Навигация внутри модуля: Программа содержит встроенный метод для вычисления суммы заданных элементов массива

Сообщения об ошибках:

- обработка ошибок ввода данных, если введено не число;
- ошибки выхода за пределы массива.

Блок-схема алгоритма программы



Содержание отчета

- 1 Название работы
- 2 Цель работы
- 3 Технические средства обучения
- 4 Задания (условия задач)
- 5 Порядок выполнения работы
- 6 Ответы на контрольные вопросы
- 7 Вывод

Варианты к практической работе:

Создайте спецификацию для разработки программы:

1. Вычислить периметр, площадь прямоугольного треугольника по данным катетам
2. Вычислить периметр и площадь треугольника, используя длины сторон (a, b, c) по формуле Герона: $S = (p * (p-a) * (p-b) * (p-c))^{1/2}$.
3. Вычислить периметр, площадь прямоугольного треугольника по гипотенузе и катету
4. Вычислить объем усеченного конуса по значениям высоты и радиусов оснований
5. Вычислить объем и площадь поверхности куба по значению диагонали

куба.

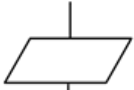
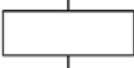
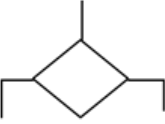
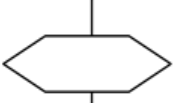
6. Вычислить объем цилиндра по высоте и радиусу оснований
7. Вычислить площадь поверхности правильной треугольной усеченной пирамиды по сторонам оснований и высоте.
8. Вычислить объем правильной треугольной пирамиды по высоте и стороне основания.
9. Вычислить объем шара по радиусу сечения, находящегося на заданном расстоянии от центра шара.
10. Вычислить объем конуса по значению радиус основания и образующей.
11. Вычислить площадь круга по длине окружности.
12. Вычислить площадь поверхности правильной шестиугольной призмы по высоте и длине стороны основания.
13. Вычислить объем октаэдра по площади его грани.
14. Вычислить площадь полной поверхности правильной четырехугольной пирамиды по высоте и стороне основания.
15. Вычислить площадь поверхности прямоугольного параллелепипеда по значениям его измерений.
16. Вычислить объем правильной треугольной усеченной пирамиды по сторонам оснований и высоте.

Задание 1.2. Практическая работа

Проверяемые результаты обучения: *OK1, OK2, ПК1.2, ПК1.3*

Цель: сформировать умения по составлению алгоритма программы; научит записывать его с помощью блок-схем. Оборудование, технические и программные средства: персональный компьютер, приложение DiagramDesigner. Порядок выполнения работы Теоретические сведения Предпочтительнее до записи на алгоритмическом языке представить алгоритм в виде блок-схемы. Для построения алгоритма в виде блок-схемы необходимо знать назначения каждого из блоков.

В таблице приводятся типы блоков и их назначения

№	Блок	Назначение блока
1		Начало или конец блок-схемы
2		Ввод или вывод данных
3		Процесс (в частности вычислительный)
4		условие
6		Цикл с параметром (for)

Алгоритмизация вычислений предполагает решение задачи в виде последовательности действий, т.е. решение, представленное в виде блок-схемы. Можно выделить типичные алгоритмы. К ним относятся: линейные алгоритмы, разветвляющиеся алгоритмы, циклические

алгоритмы.

Линейные алгоритмы

Линейный алгоритм является наиболее простым. В нём предполагается последовательное выполнение операций. В этом алгоритме не предусмотрены проверки условий или повторений. Пример: Вычислить функцию $z = (x-y)/x + y^2$.

Составить блок-схему вычисления функции по линейному алгоритму. Значения переменных x, y могут быть любые, кроме нуля, вводить их с клавиатуры.

Решение: Линейный алгоритм вычисления функции задан в виде блок-схемы на рисунке. При выполнении линейного алгоритма значения переменных вводятся с клавиатуры, подставляются в заданную функцию, вычисляется результат, а затем выводится результат.

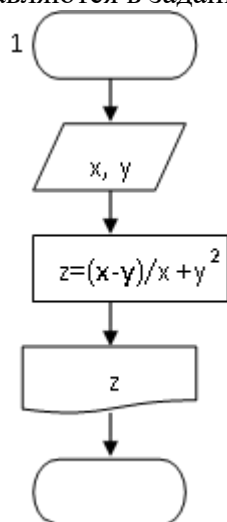


Рис.1. Линейный алгоритм

Назначение блоков в схеме на рис.1:

- Блок 1 в схеме служит в качестве логического начала.
- Блок 2 соответствует вводу данных.
- Блок 3 представляет арифметическое действие.
- Блок 4 выводит результат.
- Блок 5 в схеме служит в качестве логического завершения схемы.

Алгоритмы ветвлений

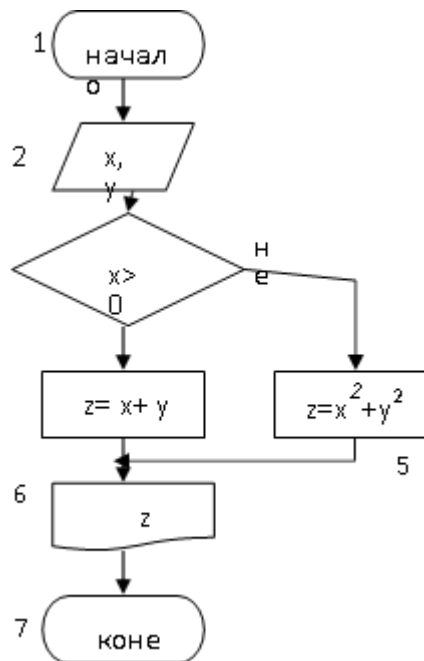
Разветвляющийся алгоритм предполагает проверку условий для выбора решения. Соответственно в алгоритме появятся две ветви для каждого условия.

В примере рассматривается разветвляющийся алгоритм, где в зависимости от условия выбирается один из возможных вариантов решений. Алгоритм представляется в виде блок-схемы.

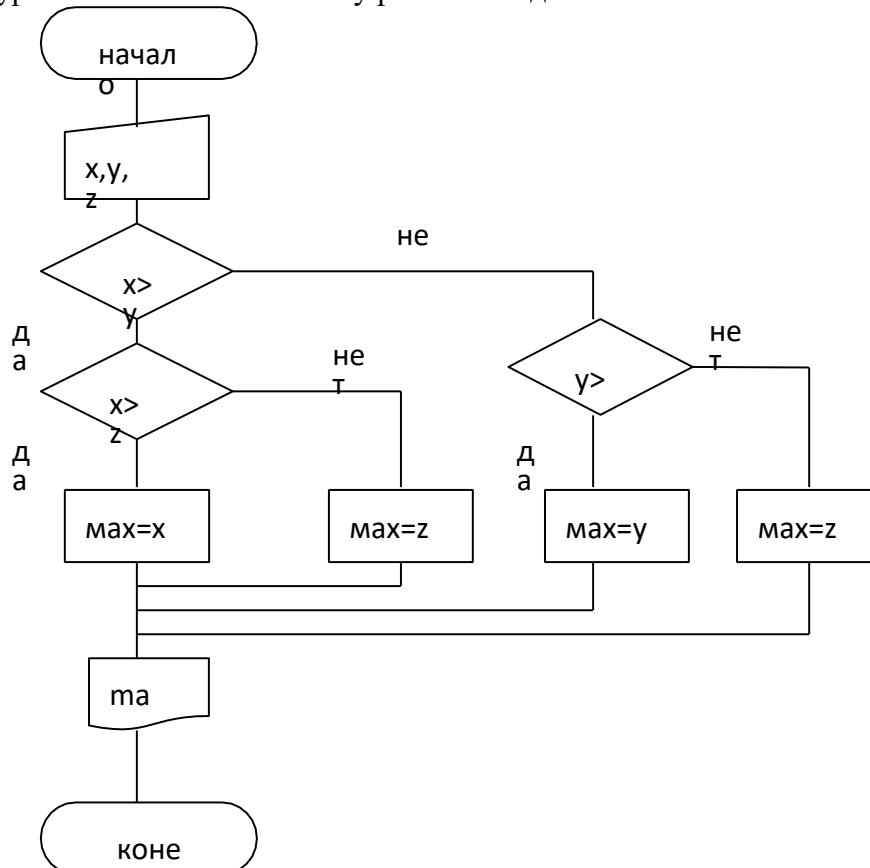
Пример: При выполнении условия $x > 0$ вычисляется функция: $z = x + y$, иначе, а именно, когда $x = 0$ или $x < 0$, вычисляется функция: $z = x^2 + y^2$. Составить блок-схему вычисления функции по алгоритму ветвления.

Значения переменных x, y могут быть любые, вводить их с клавиатуры.

Решение: На рис.2 представлен разветвляющийся алгоритм, где в зависимости от условия выполнится одна из веток. В блок-схеме появился новый блок 3, который проверяет условие задачи. Остальные блоки знакомы из линейного алгоритма.



Пример: Найти максимальное значение из трёх различных целых чисел, введенных с клавиатуры. Составить блок-схему решения задачи.



Решение: Данный алгоритм предполагает проверку условия. Для этого выбирается любая из трёх переменных и сравнивается с другими двумя. Если она больше, то поиск максимального числа окончен. Если условие не выполняется, то сравниваются две оставшиеся переменные. Одна из них будет максимальной. Блок-схема к этой задаче представлена на рис 3.

Циклические алгоритмы

Циклический алгоритм предусматривает повторение одной операции или нескольких операций в зависимости от условия задачи.

Из циклических алгоритмов выделяют два типа:

1) с заданным количеством циклов или со счётчиком циклов; 2) количество циклов неизвестно.

Пример: В цикле вычислить значение функции $z=x \cdot y$ при условии, что одна из переменных x меняется в каждом цикле на единицу, а другая переменная y не меняется и может быть любым целым числом. В результате выполнения цикла при начальном значении переменной $x=1$ можно получить таблицу умножения. Количество циклов может быть любым. Составить блок-схему решения задачи.

Решение: В примере количество циклов задаётся. Соответственно выбирается алгоритм циклов первого типа. Алгоритм этой задачи приводится на рис. 4.

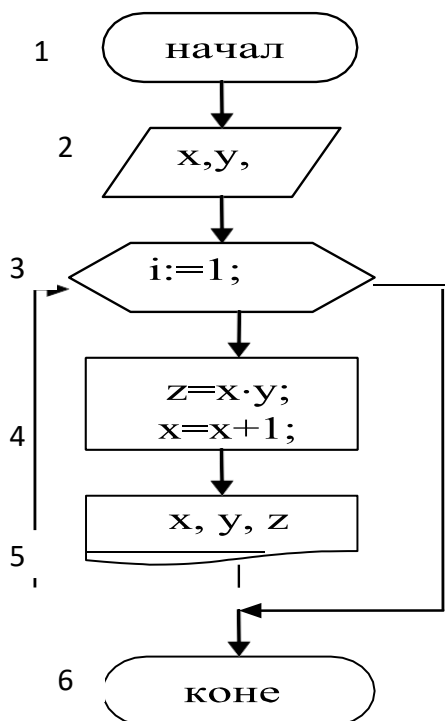
Во втором блоке вводятся количество циклов n и любые целые числа x, y .

В блок-схеме появился новый блок 3, в котором переменная i считает количество циклов, после каждого цикла увеличиваясь на единицу, пока счётчик не будет равен $i=n$. При $i=n$ будет выполнен последний цикл.

В третьем блоке указывается диапазон изменения счётчика цикла (от $i=1$ до $i=n$).

В четвёртом блоке изменяются значения переменных: z, x .

В пятом блоке выводится результат. Четвёртый и пятый блоки повторяются в каждом цикле.



Этот тип циклических алгоритмов предпочтителен, если дано количество циклов.

Если количество циклов неизвестно, то блок-схемы циклических алгоритмов могут быть представлены в виде рисунков 5, 6.

Пример: Вычислить $y=y-x$ пока $y>x$, если $y=30, x=4$. Подсчитать количество выполненных циклов, конечное значение переменной y . В цикле вывести значение переменной y , количество выполненных циклов. Составить блок-схему решения задачи.

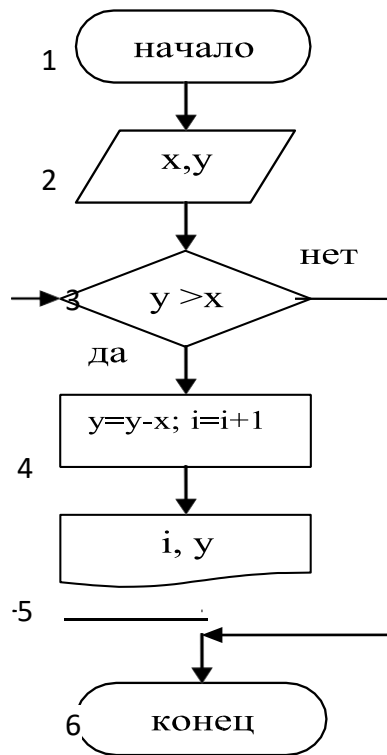
Решение: В примере количество циклов неизвестно. Соответственно выбирается алгоритм циклов второго типа. Алгоритм этой задачи приводится на рис. 5.

Условие проверяется на входе в цикл. В теле цикла выполняется два блока:

- 1) $y=y-x; i=i+1;$
- 2) вывод значений переменных i, y .

Цикл выполняется до тех пор, пока выполняется условие $y>x$. При условии равенства этих переменных $y=x$ или $y<x$ цикл заканчивается.

Алгоритм, представленный на рис.5, называется *циклический алгоритм с предусловием*, так как условие проверяется в начале цикла или на входе в цикл.



Во втором блоке вводятся $y=30$, $x=4$.

В третьем блоке проверяется условие $y > x$ на входе в цикл. Если условие выполняется, то переход к блоку 4, иначе на блок 6.

В четвертом блоке вычисляется значение переменной y , подсчитывается количество выполненных циклов $i=i+1$.

В пятом блоке выводится результат:

- значение переменной y ,
- количество выполненных циклов i .

Пример: Составить блок-схему примера (рисунок 5), проверяя условие выхода из цикла. В этом примере условие задачи не меняется, и результат выведется тот же, но блок-схема будет другой.

Решение: В этом случае проверяется условие на выход из цикла: $y \leq x$. При этом условии цикл не выполняется. Условие в блок-схеме следует перенести в конец цикла, после вывода на печать. Цикл выполняется до тех пор, пока выполняется условие $y > x$.

Алгоритм, если условие перенести в конец цикла, называется *алгоритмом цикла с постусловием*.

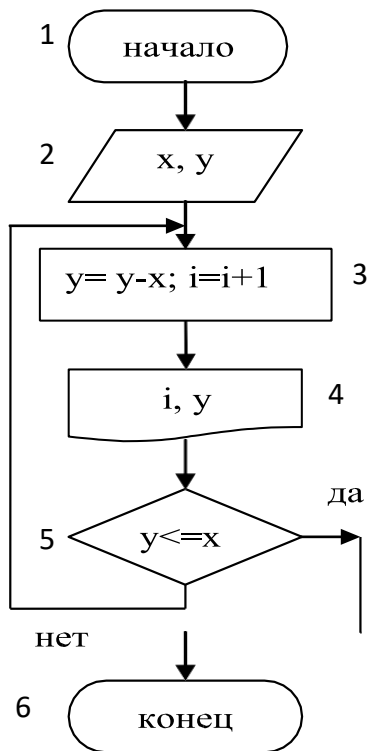
Алгоритм этой задачи приводится на рис. 6. Во втором блоке вводятся $y=30$, $x=4$.

В третьем блоке вычисляется значение переменной y , подсчитывается количество выполненных циклов $i=i+1$.

В четвертом блоке выводится результат:

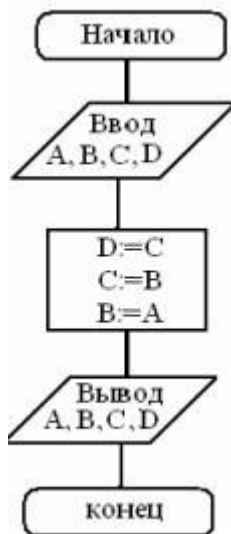
- значение переменной y ,
- количество выполненных циклов i .

В пятом блоке проверяется условие $y \leq x$ на выход из цикла. Если условие выполняется, то переход к блоку 6, иначе на блок 3 и цикл повторяется.



Задания:

1. Найти результат работы алгоритма:



Входные данные по вариантам

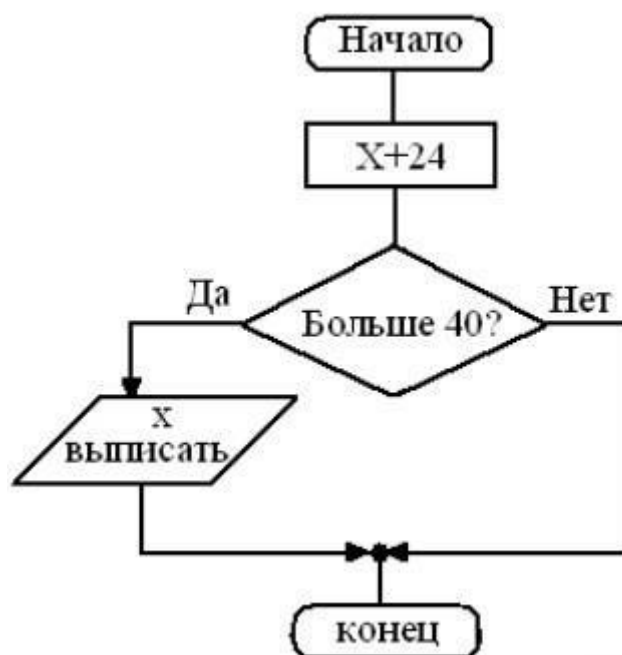
№	A	B	C	D
1	0	-1	-2	-3
2	1	0	-1	-2
3	2	1	0	-1
4	3	2	1	0
5	4	3	2	1

6	5	4	3	2
7	6	5	4	3
8	7	6	5	4
9	-3	7	6	5
10	-4	-3	7	6
11	-5	-4	-3	7
12	-6	-5	-4	-3
13	-7	-6	-5	-4
14	9	-7	-6	-5
15	8	7	-7	-6

2. При заданном X условие выполняется? Написать результат вычисления и ответ попадаем в условие или нет.

Входные данные по вариантам

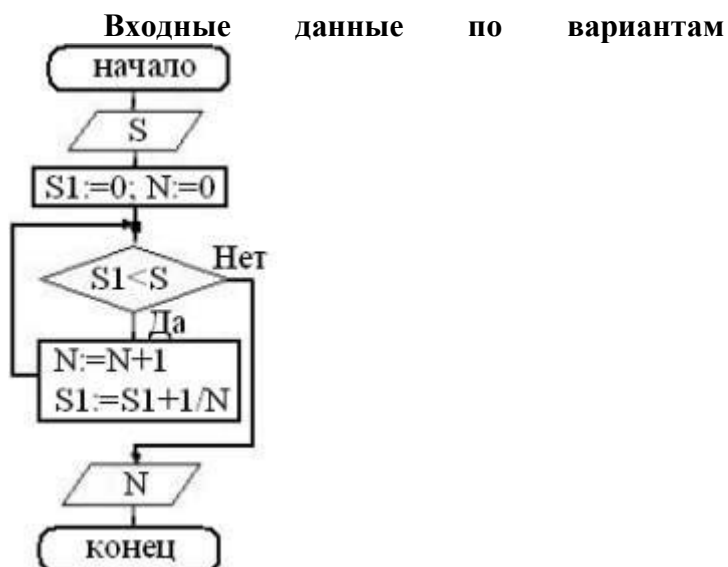
№	X1	X1
1	55	12
2	85	13
3	24	17
4	65	15
5	17	54
6	15	67
7	26	3
8	27	21
9	92	34
10	12	23
11	45	22
12	66	45
13	71	46
14	13	76
15	45	67



3. Написать результат выполнения алгоритма с указанными входными данными

№	S
1	1,5

2	1,8
3	2,4
4	1,6
5	1,7
6	1,3
7	2,6
8	2,37
9	1,92
10	1,12
11	1,45
12	2,66
13	2,71
14	2,13
15	1,45



4. Написать результат выполнения алгоритма с указанными входными данными

Входные данные по вариантам

№	X
1	-1
2	0
3	1
4	2
5	3
6	4
7	5
8	6
9	7
10	-3
11	-4
12	-5
13	-6
14	-7
15	7

5. Построить блок схему к задаче(по вариантам). Указать тип алгоритма, что дано и что нужно найти.

№	Задача
1	Дано двузначное число. Найти: число десятков в нем;
2	Дано двузначное число. Найти: число единиц в нем;
3	Дано двузначное число. Найти: сумму его цифр;
4	Дано двузначное число. Найти: произведение его цифр.
5	Дано двузначное число. Получить число, образованное при перестановке цифр заданного числа.
6	Дано трехзначное число. Найти: число единиц в нем;
7	Дано трехзначное число. Найти: число десятков в нем;

8	Дано трехзначное число. Найти: сумму его цифр;
9	Дано трехзначное число. Найти: произведение его цифр.
10	Дано трехзначное число. Найти число, полученное при прочтении его цифр справа налево.
11	Дано трехзначное число. В нем зачеркнули первую слева цифру и приписали ее в конце. Найти полученное число.
12	Дано трехзначное число. В нем зачеркнули последнюю справа цифру и приписали ее в начале. Найти полученное число.
13	Дано трехзначное число. Найти число, полученное при перестановке первой и второй цифр заданного числа
14	Дано трехзначное число. Найти число, полученное при перестановке второй и третьей цифр заданного числа.
15	Дано трехзначное число, в котором все цифры различны. Получить шесть чисел, образованных при перестановке цифр заданного числа.

6. Построить блок схему к задаче (по вариантам). Указать тип алгоритма, что дано и что нужно найти.

№	Задача
1	Определить максимальное и минимальное значения из двух различных вещественных чисел.
2	Известны два расстояния: одно в километрах, другое — в футах (1 фут 0,305 м). Какое из расстояний меньше?
3	Известны две скорости: одна в километрах в час, другая — в метрах в секунду. Какая из скоростей больше?
4	Даны радиус круга и сторона квадрата. У какой фигуры площадь больше?
5	Даны объемы и массы двух тел из разных материалов. Материал какого из тел имеет большую плотность?
6	Известны сопротивления двух несоединенных друг с другом участков электрической цепи и напряжение на каждом из них. По какому участку протекает меньший ток?
7	Даны вещественные числа a , b , c ($a \neq 0$). Выяснить, имеет ли квадратное уравнение с данными параметрами решение
8	Известны площади круга и квадрата. Определить: уместится ли круг в квадрате?
9	Известны площади круга и квадрата. Определить: уместится ли квадрат в круге?
10	Известны площади круга и равностороннего треугольника. Определить: уместится ли круг в треугольнике?
11	Известны площади круга и равностороннего треугольника. Определить: уместится ли треугольник в круге?

12	Дано двузначное число. Определить: какая из его цифр больше: первая или вторая;
13	Дано двузначное число. Определить: одинаковы ли его цифры
14	Дано двузначное число. Определить: кратна ли трем сумма его цифр;
15	Дано двузначное число. Определить: кратна ли сумма его цифр числу а.

Порядок выполнения практической работы

1. Повторить теоретический материал по данной теме.
2. Выполнить задание.
3. Подготовить ответы на теоретические вопросы.
4. Оформить отчет.
5. Защитить практическую работу.

Контрольные вопросы:

1. Что такое алгоритм?
2. Какие существуют способы представления алгоритмических решений?
3. Перечислите достоинства и недостатки описания алгоритма на естественном языке.
4. Перечислите свойства алгоритма.
5. Какие алгоритмы называют линейными?
6. Для чего в блок-схемах используют комментарии?
7. Для чего в блок-схемах используют блок соединитель?
8. Что в блок-схеме означают стрелки?
9. Что нужно сделать при необходимости перенести оформление блок-схемы на другую страницу?

Задание 1.3. Практическая работа

Проверяемые результаты обучения: ОК1, ОК2, ПК1.2., ПК1.3, ПК1.4.

Цель: выполнение автоматизации модульного тестирования

Тест

В каждом тестовом задании может быть несколько вариантов ответа. После проведения теста, студенты могут попробовать обосновать свои неверные ответы.

1. Тестовое окружение может использоваться для:
 1. запуска и выполнения тестируемого модуля
 2. передачи входных данных
 3. сбора ожидаемых выходных данных
 4. сравнения реальных выходных данных с ожидаемыми
 5. поддержки отчуждения отдельных модулей системы от всей системы
- Ответ: 1, 2, 4, 5

2. Тестовое окружение для программного кода на структурных языках программирования состоит из:
 1. драйвера
 2. тестов
 3. заглушек
 4. исходного кода
- Ответ: 1, 3

3. Модульное тестирование проводится для того, чтобы:
1. удостовериться в корректной работе системы в целом
 2. удостовериться в корректной работе набора модулей
 3. удостовериться в корректной работе отдельного модуля

Ответ: 3

4. Модуль – это (с точки зрения наших семинарских занятий):

1. часть программного кода, выполняющая одну функцию с точки зрения функциональных требований
2. программный модуль, т.е. минимальный компилируемый элемент программной системы
3. задача в списке задач проекта
4. участок кода, который может уместиться на одном экране или одном листе бумаги
5. один класс или их множество с единым интерфейсом.

Ответ: 2

5. Какие основные задачи решаются в ходе модульного тестирования?

1. Поиск и документирование несоответствий требованиям
2. Поддержка разработки и рефакторинга низкоуровневой архитектуры системы и межмодульного взаимодействия
3. Рефакторинг модулей
4. Поддержка рефакторинга модулей
5. Отладка
6. Поддержка устранения дефектов и отладки

Ответ: 1, 2, 4, 6

Возможности MVSTE по автоматизации модульного тестирования

Замечание. Подробнее о модульном тестировании можно почитать по адресу [http://msdn2.microsoft.com/en-us/library/ms182515\(VS.80\).aspx](http://msdn2.microsoft.com/en-us/library/ms182515(VS.80).aspx)

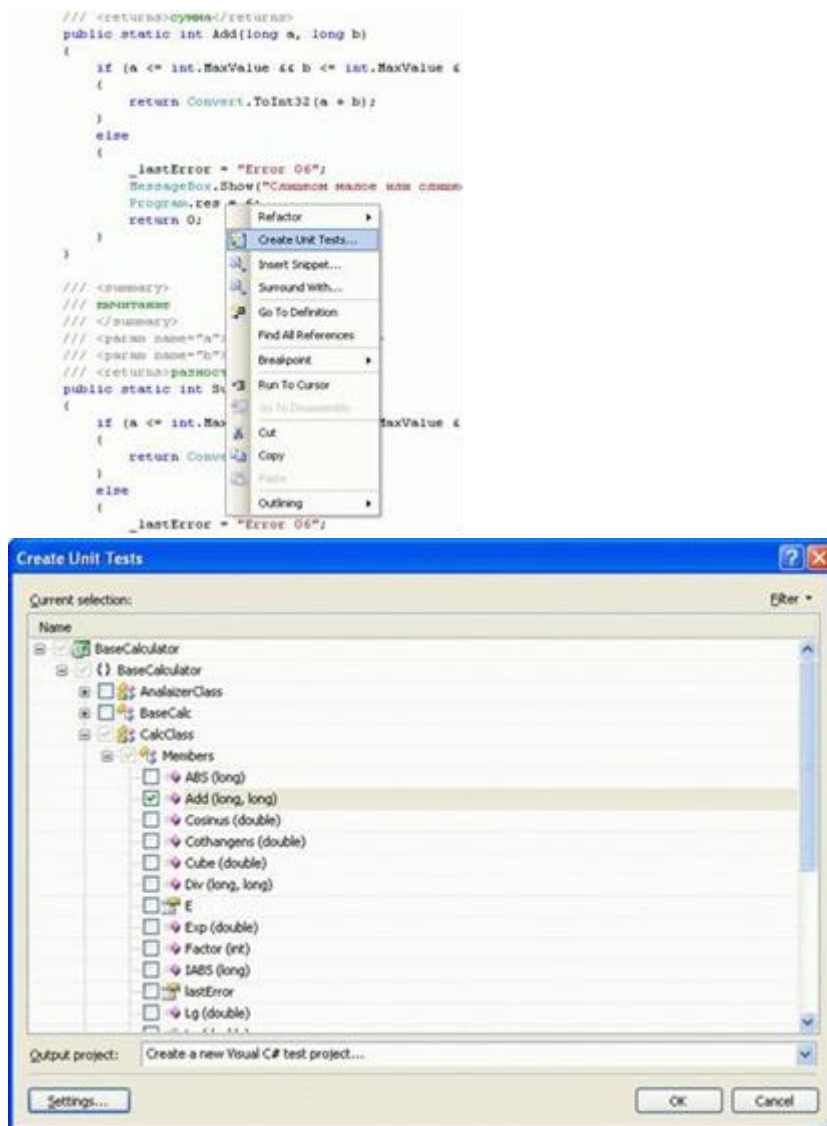
До сих пор мы выполняли часть работы вручную. Но при написании тестов тестирущик также может ошибиться, из-за чего в программе могут остаться различные ошибки. В случае, если программисты ведут разработку по методике экстремального программирования (XP), следуя практике написания тестов перед кодом (test driven development, TDD), количество тестов, которые нужно написать, становится по объему даже большим, чем сам код системы. Однако очевидно, что большую часть работы по разработке тестов отдельных методов (модульное тестирование, unit testing) можно автоматизировать. В MVSTE разработаны специальные средства для автоматизации модульного тестирования. Именно о них и пойдет речь дальше.

Начало работы

К моменту написания тестов мы уже имеем полностью готовый код. Можем приступить к созданию тестов.

Создание тестов

Для создания теста нажимаем правой кнопкой мыши на методе Add() и выбирая пункт меню Create Unit Tests.... Появится диалоговое окно, позволяющее создать тесты в другом проекте. По умолчанию, создаваемый проект — новый проект на Visual Basic, но также доступны тестовые проекты на C# и C++. Выбираем Visual C# и нажимаем кнопку ОК, перед тем введя имя проекта BaseCalculator.Test



Созданный тестовый проект содержит четыре файла, связанных с тестированием.

Имя файла	Примечание
Authoring Test.txt	Примечания о создании тестов, включающие инструкции по добавлению дополнительных тестов к проекту
CalcClassTest.cs	Включает в себя сгенерированный тест для тестирования метода Add () наряду с методами для тестовой инициализации и очистки
ManualTest1.mht	Шаблон, который заполняется инструкциями при ручном тестировании
UnitTest1.cs	Пустая структура unit test класса, куда помещаются дополнительные тесты

Так как ручное тестирование мы уже провели, и файл для тестов у нас уже есть, то удалим ManualTest1.mht и UnitTest1.cs.

В раздел References при генерации тестового проекта добавляется ссылки на Microsoft.VisualStudio.QualityTools.UnitTestFramework и проект BaseCalculator, который и будет тестироваться. Первое – сборка, которую использует "движок" модульного тестирования при выполнении тестов. Второе — это ссылка на ту сборку, которую мы тестируем. По умолчанию, сгенерированный тест-метод – это шаблон со следующей реализацией:

```
/// <summary>
///A test for Add (long, long)
```

```

///</summary>
[DeploymentItem("BaseCalculator.exe")]
[TestMethod()] public void AddTest()
{
    long a = 0; // TODO: Initialize to an appropriate value

    long b = 0; // TODO: Initialize to an appropriate value

    int expected = 0;    int actual;

    actual = BaseCalculator.Test.
        BaseCalculator_CalcClassAccessor.Add(a, b);

    Assert.AreEqual(expected, actual,
        "BaseCalculator.CalcClass.Add did not return
        the expected value.");
    Assert.Inconclusive("Verify the correctness of this test method.");
}

```

Замечание. Сгенерированный код теста будет сильно зависеть от типа и сигнатуры того метода, который планируется тестировать. Например, мастер сгенерирует код, основанный на технологии reflection ("отражение"), для тестирования private функций. В нашем конкретном случае это не потребовалось, так как метод Add() объявлен как public().

Прежде всего, отметим, что сгенерированный код помечен атрибутом TestMethod типа TestMethodAttribute, а сам класс помечен атрибутом TestClassAttribute, которые объявлены в Microsoft.VisualStudio.QualityTools.UnitTesting.Framework. При помощи технологии

Reflection движок модульного тестирования находит все тестовые классы в проекте, помеченные соответствующим атрибутом, а внутри все необходимые для тестирования методы.

Замечание. Об атрибутах можно почитать подробнее по адресу [http://msdn2.microsoft.com/en-us/library/system.attribute\(VS.80\).aspx](http://msdn2.microsoft.com/en-us/library/system.attribute(VS.80).aspx)

В начале теста объявляется значение всех необходимых переменных, а также ожидаемое выходное значение. Затем происходит вызов нужного метода, которому передаются необходимые параметры. В нашем случае это actual = BaseCalculator.Test.BaseCalculator_CalcClassAccessor.Add(a, b);

Затем идет вызов двух методов класса Assert. Прежде всего рассмотрим второй метод. Assert.Inconclusive("Verify the correctness of this test method."); Наличие этого метода в тесте говорит о том, что реализация теста еще не закончена. Сделаем реализацию нашего метода:

```

/// <summary>
///A test for Add (long, long)
///</summary>
[DeploymentItem("BaseCalculator.exe")]
[TestMethod()] public void AddTest()
{
    long a = 150;

    long b = 350;

    int expected = 500;
    int actual;

    actual = BaseCalculator.
        Test.BaseCalculator_CalcClassAccessor.Add(a, b);

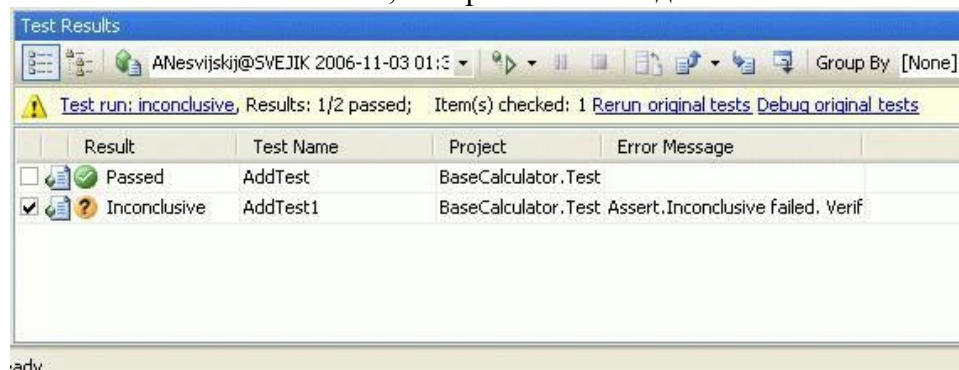
    Assert.AreEqual(expected, actual,        "BaseCalculator.CalcClass.

```

Add did not return the expected value."); } Создание тестов

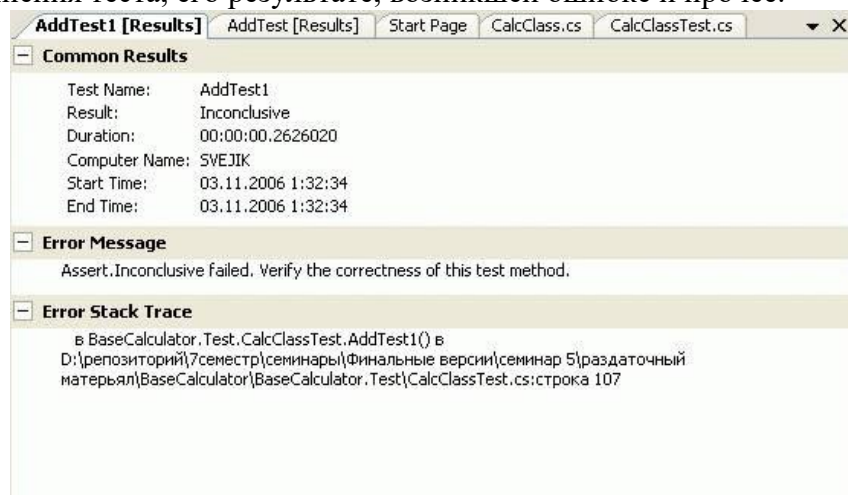
Чтобы запустить все тесты в рамках проекта, необходимо просто запустить тестовый проект. Один из возможных способов сделать это — кликнуть правой кнопкой мыши на проекте BaseCalculator.Test в Solution explorer и выбрать Set as StartUp Project. Затем используем пункты меню Debug->Start (F5) или Debug->Start Without Debugging (Ctrl+F5), чтобы начать запуск тестов.

В окне Test Results будет показан список со всеми тестами проекта. В момент начала выполнения теста в нашем проекте содержалось два теста: один полностью реализованный тест AddTest, второй – неоконченный AddTest1. В момент запуска оба теста будут в состоянии "неоконченный" (Pending), но как только тесты будут выполнены, появятся результаты выполнения Passed и Inconcluseve, которые мы и ожидали.



Замечание. На рисунке показывает окно Test Results. На этом скриншоте в дополнение к колонкам по умолчанию изображена колонка Error Message. Колонки могут быть добавлены или удалены правым щелчком мыши по меню на заголовках колонки и выборе пункта меню Add/Remove Columns... .

Чтобы посмотреть дополнительные детали о тесте, мы можем дважды щелкнуть на нем в окне Test Results и открыть окно AddTest[Results]. В нем можно узнать информацию о скорости выполнения теста, его результате, возникшей ошибке и прочее.



Кроме того, мы можем кликнуть правой кнопкой мыши на отдельных тестах и выбирать пункт меню Open Test, чтобы переместиться на код теста.

Обработка исключений

На прошлом семинаре мы обнаружили, что метод RunEstimate() класса AnalaizerClass не достаточно хорошо проверяет объекты, с которыми он работает. Если инициализировать список орз значением {2,2,+,+}, то выполнение метода RunEstimate() приводит к генерации исключения. Действительно, реализуем тест:

```
/// <summary>  
///A test for RunEstimate ()  
///</summary>
```

```

[DeploymentItem("BaseCalculator.exe")]
[TestMethod()]
public void RunEstimateTest()
{
    string expected = null;    string actual;
    // Подготовка тестового окружения
    BaseCalculator.Test.BaseCalculator_CalcClassAccessor._lastError = "";
    BaseCalculator.Test.BaseCalculator_AnalaizerClassAccessor.opz =
new System.Collections.ArrayList();
    BaseCalculator.Test.BaseCalculator_AnalaizerClassAccessor.opz.Add("2");
    BaseCalculator.Test.BaseCalculator_AnalaizerClassAccessor.opz.Add("2");
    BaseCalculator.Test.BaseCalculator_AnalaizerClassAccessor.opz.Add("+");
    BaseCalculator.Test.BaseCalculator_AnalaizerClassAccessor.opz.Add("+");

```

```

    actual = BaseCalculator.Test.BaseCalculator_AnalaizerClassAccessor.RunEstimate();

```

```

    Assert.AreEqual(expected, actual,

```

```

        "BaseCalculator.AnalaizerClass.RunEstimate did not return the expected value.");

```

} Замечание. Для работы этого теста необходимо создать начальное тестовое окружение, при этом значение `_lastError` необходимо очистить, так как оно будет "испорчено" тестом `AddTest1()`. Подробнее о зависимости тестов от порядка выполнения и тестового окружения мы поговорим на девятом семинаре.

Несмотря на то, что явных блоков try-catch не стоит, сгенерированное исключение не приведет к прекращению работы тестов, а будет корректно обработано. В этом можно убедиться, загляну в окно на `RunEstimateTest[Result]`.



Предположим теперь, что при неверных входных параметрах метод `RunEstimate()` действительно должен генерировать исключение, которое будет перехватываться в другом месте. Создадим еще один тест:

```

/// <summary>
/// A test for RunEstimate ()
/// </summary>
[DeploymentItem("BaseCalculator.exe")]
[TestMethod()]
[ExpectedException(typeof(ArgumentOutOfRangeException),
    "Была обработана неверная синтаксическая конструкция")] public void

```

```

RunEstimateTest1()
{
    BaseCalculator.Test.BaseCalculator_CalcClassAccessor._lastError = "";
    BaseCalculator.Test.BaseCalculator_AnalaizerClassAccessor.opz =
new System.Collections.ArrayList();
    BaseCalculator.Test.BaseCalculator_AnalaizerClassAccessor.opz.Add("2");
    BaseCalculator.Test.BaseCalculator_AnalaizerClassAccessor.opz.Add("2");
    BaseCalculator.Test.BaseCalculator_AnalaizerClassAccessor.opz.Add("+");
    BaseCalculator.Test.BaseCalculator_AnalaizerClassAccessor.opz.Add("+");

    BaseCalculator.Test.BaseCalculator_AnalaizerClassAccessor.RunEstimate(); }

```

Отметим, что, опять же, нет блока try-catch с явным тестом на ArgumentException. Вместо этого тест включает дополнительный атрибут, ExpectedException, который принимает тип параметра, и произвольное сообщение об ошибке, которое будет показано, если исключение не было брошено. Когда тесты выполняются, среда будет явно следить за тем, чтобы исключение ArgumentException было сгенерировано, и если метод не будет генерировать такое исключение, то тест будет провален.

Программа

Будут выданы исходные файлы модулей для тестирования методом "белого ящика" средствами MVSTE, пример тестового драйвера.

Составить тест-план и провести модульное тестирование (средствами MVSTE) следующих методов:

1. public static int Mod(long a, long b)
2. public static bool CheckCurrency()
3. public static int ABS(long a)
4. public static string Format()
5. public static int IABS(long a)
6. public static string Format()
7. public static int Sub(long a, long b)
8. public static System.Collections.ArrayList CreateStack()
9. public static int Mult(long a, long b)
10. public static System.Collections.ArrayList CreateStack()
11. public static int Div(long a, long b)
12. public static bool CheckCurrency()

Критерии оценки:

Оценка «отлично» – практические задачи выполнены в полном объеме, студент отвечает на все поставленные вопросы. Все задания выполнены правильно. Работа оформлена аккуратно и в соответствии с требованиями. Продемонстрировано глубокое понимание материала.

Оценка «хорошо» – студент допускает незначительные неточности, правильно применены теоретические знания. Задания выполнены с небольшими ошибками. Оформление работы имеет несущественные недочеты. Продемонстрировано хорошее понимание материала

Оценка «удовлетворительно» – отсутствие полного объема работ; низкое качество выполнения работ, часть заданий выполнена с ошибками; Оформление работы имеет существенные недочеты; продемонстрировано базовое понимание материала

Оценка «неудовлетворительно» – отсутствие полного объема работ; в работе допущены серьезные ошибки и нарушение всех перечисленных выше требований; отсутствует понимание материала.

Задание 1.4. Тестирование

Формируемые компетенции: ОК1, ОК2, ПК1.1, ПК1.2, ПК 11.3, ПК1.4.

Цель тестового задания - контроль знаний освоения дисциплины, получение ответа от

испытуемого, на основе которого может быть сделан вывод о его знаниях, представлениях из определенной области содержания дисциплины.

Задание: перечень вопросов, соответствующих содержанию дисциплины.

Инструкция: выберите один/несколько правильных ответов из предложенных

1. Когда необходимо составлять блок-схему программы?

- а) До начала составления самой программы
- б) После составления программы
- в) В процессе составления программы

2. Перевод программ с языка высокого уровня на язык более низкого уровня обеспечивает:

- а) Java
- б) Паскаль
- в) Компилятор
- г) Фортран

3. Элемент-кнопку добавляет на форму метод:

- а) `frame.add(myButton);`
- б) `frame.add(myField);`
- в) `frame.add(myLabel);`

4. Переменная строкового типа обозначается:

- а) String
- б) Word
- в) Char
- г) Byte

5. Методология программирования, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определённого класса:

- а) Инкапсуляция
- б) Наследование
- в) Полиморфизм
- г) Объектно-ориентированное программирование

6. Какие методы может выполнять Scanner?

- а) Ввод данных с консоли
- б) Чтение из файла
- в) Анализ строк
- г) Все вышеперечисленные

7. Что такое жизненный цикл программного обеспечения?

- а) Совокупность этапов создания и эксплуатации ПО
- б) Время работы программы
- в) Процесс тестирования
- г) Период разработки

8. Какие основные этапы включает жизненный цикл ПО?

- а) Планирование, разработка, тестирование
- б) Анализ требований, проектирование, реализация
- в) Внедрение, эксплуатация, поддержка
- г) Все вышеперечисленные

9. Что такое цикл for?

- а) Цикл с предусловием
- б) Цикл с постусловием
- в) Параметрический цикл
- г) Бесконечный цикл

10. Какие существуют типы коллекций в Java?

- а) List
- б) Set
- в) Map
- г) Все вышеперечисленные

11. Что такое сериализация?

- а) Процесс сохранения объекта в файл
- б) Чтение объекта из файла
- в) Преобразование объекта в последовательность байт
- г) Все вышеперечисленное

12. Какие основные компоненты включает класс File?

- а) Работа с файлами
- б) Работа с каталогами
- в) Получение атрибутов
- г) Все вышеперечисленное

13. Что такое регулярные выражения?

- а) Набор символов для поиска
- б) Шаблоны для обработки строк
- в) Методы класса String
- г) Все вышеперечисленное

14. Какие основные операции включает JDBC?

- а) Подключение к базе данных
- б) Выполнение запросов
- в) Обработка результатов
- г) Все вышеперечисленное

15. Что такое событийно-управляемое программирование?

- а) Обработка событий пользователя
- б) Работа с графическим интерфейсом
- в) Обработка сигналов системы
- г) Все вышеперечисленное

№ задания	Верный ответ	Критерии
1	В	1б – полное правильное соответствие 0 б – остальные случаи
2	В	1б – полное правильное соответствие 0 б – остальные случаи
3	А	1б – полное правильное соответствие 0 б – остальные случаи
4	А	1б – полное правильное соответствие 0 б – остальные случаи
5	Г	1б – полное правильное соответствие

		0 б – остальные случаи
6	Г	1б – полное правильное соответствие 0 б – остальные случаи
7	А	1б – полное правильное соответствие 0 б – остальные случаи
8	Г	1б – полное правильное соответствие 0 б – остальные случаи
9	В	1б – полное правильное соответствие 0 б – остальные случаи
10	Г	1б – полное правильное соответствие 0 б – остальные случаи
11	Г	1б – полное правильное соответствие 0 б – остальные случаи
12	Г	1б – полное правильное соответствие 0 б – остальные случаи
13	Б	1б – полное правильное соответствие 0 б – остальные случаи
14	Г	1б – полное правильное соответствие 0 б – остальные случаи
15	Г	1б – полное правильное соответствие 0 б – остальные случаи

Критерии оценки:

соответствие ответов обучающихся ключу теста

Оценка «**отлично**» - если обучающийся правильно выполнил от 80% до 100% тестовых заданий в отведенное время

Оценка «**хорошо**» - если обучающийся правильно выполнил от 60% до 80% тестовых заданий в отведенное время

Оценка «**удовлетворительно**» - если обучающийся правильно выполнил от 40% до 60% тестовых заданий в отведенное время

Оценка «**неудовлетворительно**» ставится в случае выполнения менее 40% тестовых заданий

Время выполнения: 35-40 минут

Тема 2. Разработка, администрирование и защита баз данных

Задание 2.1.1 Практическая работа

Проверяемые результаты обучения: *ОК1, ОК2, ПК11.2, ПК11.3*

Цель – оценить: умения и знания выбирать способы решения задач профессиональной деятельности; использовать современные средства поиска, анализа и интерпретации информации и информационных технологий для выполнения работы, пользоваться профессиональной документацией при выполнении работы; формирование практических умений в соответствии с требованиями рабочей программой, включающих организацию и конфигурирование компьютерных сетей; построение и анализ моделей компьютерных сетей; эффективное использование аппаратных и программных компонентов; работу с протоколами разных уровней; установку и настройку параметров протоколов; проверку правильности передачи данных

Задание:

1. Создать таблицу **ПОКУПАТЕЛЬ** следующей структуры (в скобках указаны свойства поля):

Поле	Тип данных
Код покупателя	Счетчик (ключ)
Фамилия ИО	Текстовый (25 символов)
Телефон	Текстовый (8 символов)
Адрес	Текстовый (30 символов)
Примечание	Поле МЕМО

2. Создать таблицу ЗАКАЗ следующей структуры:

Поле	Тип данных
Код заказа	Счетчик (ключ)
Код товара	Числовой (Длинное целое)
Код покупателя	Числовой (Длинное целое)
Количество товара	Числовой (Целое)
Дата продажи	Дата/Время

Замечание: Свойства Длинное целое и Целое выбираются в строке Свойств Размер поля.

1. Создать таблицу ТОВАР следующей структуры:

2. Заполнить таблицу ТОВАР данными, так как на нее будут ссылки в последующем изложении.

3. Заполнить таблицу ПОКУПАТЕЛИ. Ввести 5-7 записей, содержание записей (фамилии, адреса, телефоны и т.п.) придумать самим.

4. Заполнить таблицу ЗАКАЗ, ввести 15-20 записей. В этой таблице будут содержаться данные о том, какой товар купил покупатель. В качестве ссылок на товары и покупателей используются их коды из первых двух таблиц, поэтому записи в таблице ЗАКАЗ должны быть согласованы по полям Код товара и Код покупателя, то есть содержать номера-коды из диапазона соответствующих полей таблиц ТОВАР и ПОКУПАТЕЛЬ.

Если в ней будут содержаться ссылки на товары или покупателей с несуществующими номерами, то в дальнейшем невозможно будет установить связи между ними.

1. Выполнить корректировку записей.

2. Удалить по 1-2 записи из таблиц и дополнить их другими записями.

3. Создать таблицу КАТЕГОРИЯ ПОКУПАТЕЛЯ с двумя полями: Код и Название, аналогичную описанной выше. Покупателей будем делить на три категории: Организация, Частное лицо, Неизвестно.

Заполнить таблицу.

1. Добавить в таблицу ПОКУПАТЕЛЬ Числовое поле Категория типа Длинное целое. Заполнить его согласно таблице КАТЕГОРИЯ ПОКУПАТЕЛЯ, т.е. записать туда значения 1, 2 или 3.

2. Аналогично создать таблицу с именем КАТЕГОРИЯ ТОВАРА с двумя полями: Код и Значение. Поле Код будет типа Счетчик и первичным ключом, поле

Значение текстовым. Будем сокращать названия значений: Выч. техн., Копир., Progr., Канц., поэтому длину поля выберем, скажем, равной 10. Заполним таблицу КАТЕГОРИЯ ТОВАРА, исходя из тех товаров, которые имеются в таблице ТОВАР.

3. Добавить в таблицу Товар Числовое поле Категория типа Длинное целое. Заполнить его согласно таблице КАТЕГОРИЯ ТОВАРА, т.е. записать туда значения 1, 2, 3, 4.

Задание 2.1.2 Практическая работа

Проверяемые результаты обучения: ОК1, ОК2, ПК11.2, ПК11.3

Цель – Получить теоретические знания и практические навыки реализации баз данных (БД). Осуществить анализ предметной области. Освоить концептуальное проектирование и научиться определять сущности и атрибуты БД. Научиться разрабатывать инфологическую модель БД в виде ER-диаграмм. Получить теоретические знания и практические навыки при физическом проектировании баз данных (БД). Научиться создавать даталогическую модель БД.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

1. ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ БАЗЫ ДАННЫХ

Понятие БД и СУБД Информационная система - система, реализующая автоматизированный сбор, обработку и манипулирование данными и включающая технические средства обработки данных, программное обеспечение и соответствующий персонал.

Цель любой информационной системы - обработка данных об объектах реального мира. Основой информационной системы является база данных. В широком смысле слова база данных - это совокупность сведений о конкретных объектах реального мира в какой-либо предметной области.

Под предметной областью принято понимать часть реального мира, подлежащего изучению для организации управления и в конечном счете автоматизации, например, предприятие, вуз и т.д. Создавая базу данных, пользователь стремится упорядочить информацию по различным признакам и быстро производить выборку с произвольным сочетанием признаков. Большое значение при этом приобретает структурирование данных. Структурирование данных - это введение соглашений о способах представления данных. Неструктурированными называют данные, записанные, например, в текстовом файле. Ниже приведен пример неструктурированных и структурированных данных, содержащих сведения о студентах (номер личного дела, фамилию, имя, отчество и год рождения). Неструктурированные данные:

Личное дело № 16493. Сергеев Петр Михайлович, дата рождения 1 января 1976 г.; Л/д № 16593, Петрова Анна Владимировна, дата рожд. 15 марта 1975 г.; № личн.дела 16693, д.р. 14.04.76, Анохин Андрей Борисович

Легко убедиться, что сложно организовать поиск необходимых данных, хранящихся в неструктурированном виде. Структурированные данные:

№ личного	Фамилия	Имя	Отчество	Дата рождения
16493	Сергеев	Петр	Михайлович	01.01.76
16593	Петрова	Анна	Владимировна	15.03.75
16693	Анохин	Андрей	Борисович	14.04.76

В современной технологии баз данных предполагается, что создание базы данных, ее поддержка и обеспечение доступа пользователей к ней осуществляются централизованно с помощью специального программного инструментария – системы управления базами данных

(СУБД).

База данных (БД) - это поименованная совокупность данных, отражающая состояние объектов и их отношений в рассматриваемой предметной области.

Объектом называется элемент предметной области, информацию о котором мы сохраняем.

Объект может быть реальным (например, человек, изделие; или населенный пункт) и абстрактным (например, событие, счет покупателя или изучаемый студентами курс).

Так, в области продажи автомобилей примерами объектов могут служить МОДЕЛЬ АВТОМОБИЛЯ, КЛИЕНТ и СЧЕТ. На товарном складе – это ПОСТАВЩИК, ТОВАР, ОТПРАВЛЕНИЕ и т. д.

Понятие базы данных тесно связано с такими понятиями структурных элементов как поле, запись, файл (таблица)



Структурные элементы базы данных

Поле - элементарная единица логической организации данных, которая соответствует неделимой единице информации - реквизиту. Для описания поля используются следующие характеристики:

- имя, например, Фамилия, Имя, Отчество, Дата рождения;
- тип, например, символьный, числовой, денежный;
- длина, например, 15 байт, причем будет определяться максимально возможным количеством символов;
- точность для числовых данных, например два десятичных знака для отображения дробной части числа,

Запись - совокупность логически связанных полей.

Экземпляр записи - отдельная реализация записи, содержащая конкретные значения ее полей.

Файл (таблица) - совокупность экземпляров записей одной структуры.

Описание логической структуры записи файла содержит последовательность расположения полей записи и их основные характеристики, как это показано

Имя файла					
Поле		Признак ключа	Формат поля		
Имя (обозначение)	Полное наименование		Тип	Длина	Точность (для чисел)
имя 1					
имя N					

В структуре записи файла указываются поля, значения которых являются ключами: первичными (ПК) и внешними (ВК),

Первичный ключ (ПК) - это одно или несколько полей, однозначно идентифицирующих запись. Если первичный ключ состоит из одного поля, он называется простым, если из нескольких полей - составным ключом.

Внешний ключ (ВК) - это одно или несколько полей, которые выполняют роль поисковых или группировочных признаков. В отличие от первичного, значение внешнего ключа может повторяться в нескольких записях файла, то есть он не является уникальным.

Если по значению первичного ключа может быть найден один единственный экземпляр записи, то по внешнему - несколько.

Имя файла: СТУДЕНТ					
Поле		Признак ключа	Формат поля		
Обозначение	Наименование		Тип	Длина	Точность
Номер	№ личного дела	•	Симв	5	
Фамилия	Фамилия студента		Симв	15	
Имя	Имя студента		Симв	10	
Отчество	Отчество студента		Симв	15	
Дата	Дата рождения		Дата	8	

Понятие модели данных

Для того, чтобы спроектировать структуру базы данных, необходима исходная информация о предметной области. Желательно, чтобы эта информация была представлена в формализованном виде.

Такое формализованное описание предметной области будем называть инфологической (infological) моделью предметной области (ИЛМ) или концептуальной моделью (КМ).

Ядром любой базы данных является модель данных. Модель данных представляет собой множество структур данных, ограничений целостности и операций манипулирования данными. С помощью модели данных могут быть представлены объекты предметной области и взаимосвязи между ними.

Модель данных - совокупность структур данных и операций их обработки.

СУБД основывается на использовании иерархической, сетевой или реляционной модели, на комбинации этих моделей или на некотором их подмножестве.

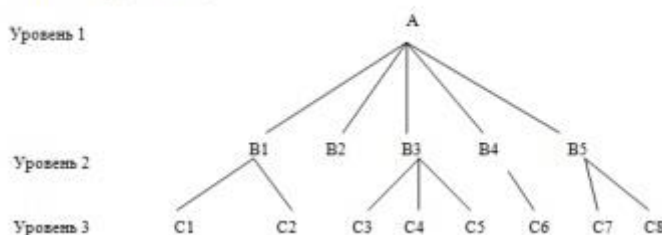
Самой распространенной моделью данных является - реляционная.

Иерархическая модель данных

Иерархическая модель организует данные в виде древовидной структуры

К основным понятиям иерархической структуры относятся: уровень, элемент (узел), связь. Дерево представляет собой иерархию элементов, называемых узлами. Узел - это совокупность атрибутов данных, описывающих некоторый объект. На самом верхнем уровне иерархии имеется один и только один узел - корень. Каждый узел, кроме корня, связан с одним узлом на более высоком уровне, называемым исходным для данного узла. Ни один элемент не имеет более одного исходного. Каждый элемент может быть связан с одним или несколькими элементами на более низком уровне. Они называются порожденными

(испыми (рис. 7)).

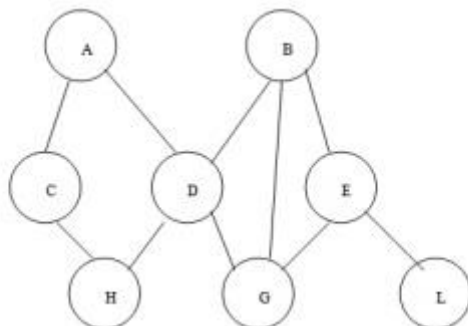


Сетевая модель данных

Сетевая модель организует данные в виде сетевой структуры.

Структура называется сетевой, если в отношениях между данными порожденный элемент имеет более одного исходного.

В сетевой структуре при тех же основных понятиях (уровень, узел, связь) каждый элемент может быть связан с любым другим элементом.



Реляционная модель данных

Реляционная модель данных является совокупностью взаимосвязанных двумерных таблиц объектов модели. Например, реляционной таблицей можно представить информацию о студентах, обучающихся в вузе

(рис. 6).

№ личного дела	Фамилия	Имя	Отчество	Дата рождения	Группа
16493	Сергеев	Петр	Михайлович	01.01.76	111
16593	Петрова	Анна	Владимировна	15.03.75	112
16693	Анохин	Андрей	Борисович	14.04.76	111

Связи между двумя логически связанными таблицами в реляционной модели устанавливаются по равенству значений одинаковых атрибутов этих таблиц.

Каждая реляционная таблица представляет собой двумерный массив и обладает следующими свойствами:

- каждый элемент таблицы - один элемент данных;
- все столбцы в таблице однородные, т.е. все элементы в столбце имеют одинаковый тип (числовой, символьный и т.д.) и длину;
- каждый столбец имеет уникальное имя;
- одинаковые строки в таблице отсутствуют;
- порядок следования строк и столбцов может быть произвольным.

При описании реляционной модели часто используют следующие термины: отношение, кортеж, домен.

Отношения представлены в виде таблиц, строки которых соответствуют записям (кортежам), а столбцы полям, атрибутам отношений (доменам).

Поле, каждое значение которого однозначно определяет соответствующую запись, называется простым ключом (ключевым полем). Если записи однозначно определяются значениями нескольких полей, то такая таблица базы данных имеет составной ключ.

Между двумя реляционными таблицами могут быть сформированы связи.

Различные таблицы, могут быть связаны между собой через общее поле данных.



Таблица СТУДЕНТ имеет поля: Номер, Фамилия, Имя, Отчество, Дата рождения, Группа;

СЕССИЯ - Номер, Оценка 1, Оценка 2, Оценка 3, Оценка 4, Результат;

СТИПЕНДИЯ - Результат, Процент

Таблицы СТУДЕНТ И СЕССИЯ имеют совпадающие ключи (Номер), что дает возможность легко организовать связь между ними.

Таблица СЕССИЯ имеет первичный ключ Номер и содержит внешний ключ Результат, который обеспечивает ее связь с таблицей СТИПЕНДИЯ.

Благодаря имеющимся связям достигаются следующие преимущества:

1. Удастся избежать дублирования информации. Все необходимые данные можно хранить только в одной таблице. Так, например, нет необходимости в таблице СЕССИЯ хранить номер группы каждого студента, сдающего экзамены, достаточно задать связь с таблицей СТУДЕНТ.

2. В реляционных базах данных легко производить изменения. Если в таблице СЕССИЯ изменить какие-нибудь значения, то правильная информация автоматически будет связана с другими таблицами, ссылающимися на первую (например, таблица СТИПЕНДИЯ).

3. В нереляционных базах данных сложно передать все имеющиеся зависимости, т.е. связать друг с другом данные из различных таблиц. Реляционная база данных выполняет все эти действия автоматически.

4. В реляционных базах данных удастся легко избежать установления ошибочных связей между различными таблицами данных, а необходимый объем памяти сокращен до минимума.

Задание 2.1.3: Практическая работа

Проверяемые результаты обучения: ОК1, ОК2, ПК11.2, ПК11.3

Цель: изучить технологию импорта данных пользователя в базу данных

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Импорт - копирует данные в БД СУБД из внешних файлов, которые были созданы утилитой экспорта.

Import - дополнительная утилита в основном применяется для резервного копирования и миграции БД (между серверами либо из более старой версии в более новую). Ниже приведены другие возможности утилиты Import :

1. хранение данных в файлах ОС для архивирования;
2. выборочное резервное копирование частей БД;
3. перемещение данных из одной пользовательской схемы в другую;
4. перемещение данных с одной аппаратной платформы или ОС в другую.
5. экономия пространства и повышение производительности за счет уменьшения фрагментации.

FILE expdat.dmp Имя файла, из которого будут импортироваться данные. По умолчанию именем файла, из которого осуществляется импорт, будет expdat.dmp.

FROMUSER Если указан этот параметр, то импортируются только те объекты, владельцем которых является пользователь с идентификационным кодом FROMUSER

FULL N Если FULL=Y, то импортироваться будет вся БД.

GRANTS Y Указывает, будут ли заданы все полномочия для экспортированных

объектов.

HELP N Если задано **HELP=Y**, то другие параметры не требуются. На экране будет выведена справочная информация.

IGNORE NO Если задано **IGNORE=Y**, то ошибки при создании объектов игнорируются и строки вставляются в таблицу. Будьте внимательны, поскольку, если для таблицы не определено ограничение уникальности значений, то могут появиться дублирующие записи.

Отметим, что о других ошибках, не связанных с созданием объектов (например, проблемах с ОС), пользователь будет информирован в обычном режиме.

INDEXES Y Указывает, экспортируются ли определяемые пользователем индексы. Системные индексы, созданные посредством определения ограничений (первичный ключ, уникальный ключ) импортируются независимо от значения параметра **INDEXES**.

LOG Имя файла, в который будет записан журнал импорта. Если не указано иное, Oracle даст файлу расширение **LOG**.

TABLES Указывает список таблиц (с запятой в качестве разделителя), которые должны быть импортированы. Этот параметр используется совместно с параметром

FROMUSER. В не-UNIX среде, как, например, Windows, необходимо заключать список таблиц в круглые скобки.

TOUSER Параметр **TOUSER** указывает имя пользователя, который будет владельцем импортируемых объектов. Данный параметр необходимо использовать совместно с параметром **FROMUSER**.

USERID Указывает имя и пароль пользователя, который осуществляет процесс импорта. Формат параметра — «имя пользователя/пароль@сервер».

ХОД РАБОТЫ

1. Подключитесь к учебной БД под учетной записью **student**. Создайте двух новых пользователей (**USER1**) и (**USER2**). Создайте новую роль. Присвойте роли права подключаться, создавать таблицы, создавать последовательности, создавать триггеры и роль **DBA**.

2. Создайте таблицу-справочник стран: **ID** (первичный ключ), название страны (символьное, уникальное). Добавьте в таблицу две-три записи.

3. Создайте таблицу-справочник городов: **ID** (первичный ключ), страна (внешний ключ к таблице стран), название города (символьный). Создайте последовательность. Создайте триггер к таблице, который перед вставкой записи заполняет первичный ключ. Добавьте в таблицу пять записей.

4. Импортируйте таблицу из файла в схему пользователя **USER1**. Подключитесь к учебной БД под учетной записью **USER1**. Напишите запрос, который бы возвращал все записи таблицы стран. Добавьте три записи в таблицу. Удалите таблицу стран.

5. Подключитесь под учетной записью **student**. Создайте хранимую процедуру, которая бы возвращала список городов с привязанными странами, т.е. город и страна. Для этого примените курсорный цикл. Вызовите хранимую процедуру.

6. Импортируйте схему пользователя **student** из файла в схему пользователя **USER2**. Подключитесь к учебной БД под учетной записью **USER2**. Напишите запрос, который бы возвращал все записи таблицы городов. Добавьте три записи в таблицу. Удостоверьтесь, что триггер заполнил первичный ключ.

7. Создайте текстовый файл или новый документ в **EXCEL**. Заполните десять строк для переноса в таблицу городов с помощью **SQL*Loader**. Создайте в текстовом редакторе управляющий файл и импортируйте в таблицу городов данные. Напишите запрос, который бы возвращал все записи таблицы городов. Убедитесь, что после импорта появились новые записи.

Задание 2.1.4: Практическая работа

Проверяемые результаты обучения: *OK1, OK2, ПК11.2, ПК11.3*

Цель: Освоение некоторых возможностей автоматизации управления базой данных

ХОД РАБОТЫ

При работе с базой данных часто приходится многократно выполнять одинаковые порой рутинные операции. Вполне естественно было бы автоматизировать их выполнение. Для этого СУБД располагает достаточными средствами, позволяющими во многом автоматизировать и упорядочить работу с базой данных. К числу таких средств относятся:

- пользовательские меню и инструментальные панели;
- кнопочные формы управления базой данных;
- средства настройки параметров запуска базы данных;
- макросы и модули.

Задание 1. Создайте пользовательское меню для управления базой данных, содержащее категории Формы и Отчеты с пунктами (командами) для открытия ранее составленных форм и отчетов.

1. Для создания новой строки меню откройте окно Настройка. Для этого выполните команду ВИД/Панели инструментов/Настройка или, щелкнув правой клавишей по любой панели инструментов, выберите в контекстном меню пункт Настройка.

2. В окне Настройка на вкладке Панели инструментов щелкните по кнопке Создать.

3. В окне Создание панели инструментов введите имя панели инструментов: Управление базой данных. Нажмите кнопку Ок. В окне БД появится небольшая миниатюра панели. Перетащите созданную панель инструментов в верхнюю часть окна установив над строкой меню.

4. В окне Настройка нажмите кнопку Свойства и определите тип созданной панели
- Строка меню. Закройте окно установки свойств.

5. Добавьте в строку созданного меню категорию Формы. Для этого в окне Настройка откройте вкладку Команды и в списке категорий щелкните по категории Новое меню. Перетащите команду Новое меню из списка команд в правом подокне на строку меню Управление базой данных. Не закрывая окно Настройка, щелкните правой клавишей в строке меню по категории Новое меню и в контекстном меню замените имя категории на Формы.

6. Добавьте в меню категорию Отчеты аналогично пункту 5.

7. Аналогично добавьте в меню Формы новое подменю, назвав его Простые.

8. В окне Настройка на вкладке Команды выделите категорию Все формы. Перетащите строку с названием одной из созданных ранее форм - Студент простая в область команд (пунктов) категории Формы строки меню Управление базой данных. Включив контекстное меню новой команды, установите стиль отображения - Только текст.

9. Аналогично добавьте в область команд категории Формы/Простые пункты с названием форм – Группа и Простая форма по запросу.

10. Добавьте в категорию Отчеты меню Управление базой данных пункты с названиями отчетов. Закройте окно Настройка. Проверьте работу меню.

11. Выполните команду СЕРВИС/Параметры запуска и установите в окне Параметры запуска следующие параметры запуска при открытии базы данных:

- введите в качестве заголовка приложения название Академия;
- выберите в качестве строки меню строку Управление базой данных.

– отмените вывод на экран окна базы данных, строки состояния, полного набора меню встроенных панелей инструментов.

Закройте окно Параметры запуска. Закройте базу данных, затем повторно откройте. Откроется окно базы данных, содержащее только одну строку пользовательского меню Управление базой данных с категориями Формы и Отчеты. Убедитесь в правильной работе команд меню.

12. Восстановите для базы данных Академия отображение окна базы данных, полного набора меню, встроенных панелей инструментов. Для этого перезагрузите базу данных и при повторном открытии держите нажатой клавишу SHIFT. Выполните команду СЕРВИС/Параметры запуска и восстановите исходное состояние флажков.

Задание 2. Создайте макрос для автоматического формирования экзаменационных

ведомостей, рассмотренных ранее. Отдельные таблицы должны быть созданы для каждой группы студентов, имеющейся в базе данных, и для выбранной дисциплины.

Целью разработки макроса является исключение необходимости каждый раз перед созданием новой таблицы (экзаменационной ведомости) вручную переименовывать ранее созданную таблицу, чтобы предотвратить ее удаление. Макрос должен сам создавать таблицы с именами, соответствующими номерам групп и кодам дисциплин, по схеме: Ведомость NNN-К, где NNN - номер группы, введенный в диалоговом окне, К - код дисциплины.

Для того, чтобы передать параметры создаваемых таблиц (номер группы и код дисциплины) можно, например, использовать форму, созданную на основе временной таблицы и выбирать эти параметры из первой строки этой формы.

Кроме того, для того чтобы избежать лишних остановок при выполнении макроса, поручите макросу не выводить вспомогательные служебные сообщения и сообщения-предупреждения. При выполнении макроса пользователь должен будет вводить только номер группы и код дисциплины.

При конструировании макроса можно использовать ранее созданный запрос с именем Запрос на создание экзаменационной ведомости.

Поскольку условия выполнения макрокоманд могут определяться только значениями полей или элементов управления форм и отчетов предварительно следует создать вспомогательную табличную форму на основании таблицы Ведомость 1.

Автоматически создайте новую табличную форму на основании таблицы Ведомость Для этого в окне База данных выберите объект Формы и щелкните по кнопке Создать.

В диалоговом окне Новая форма выберите способ создания - Автоформа: табличная, а в качестве источника данных - таблицу Ведомость Щелкните по кнопке Ok. После появления на экране формы закройте ее и сохраните под именем Форма для макроса.

Для того, чтобы форма не зависела от таблицы Ведомость 1 замените имя источника записей в окне свойств формы на Ведомость 00. Для этого откройте форму в режиме конструктора и щелкните по кнопке Свойства, расположенной инструментальной панели Конструктор форм. Отредактируйте значение свойства Источник записей на вкладке Данные.

В окне База данных выберите объект Макросы и щелкните по кнопке Создать. Появится окно конструктора макросов. Добавьте в окне еще один столбец - Условия. Для этого выполните команду ВИД/Условия или щелкните по кнопке инструментальной панели с соответствующим названием.

Щелкните внутри ячейки первой строки и столбца Макрокоманда. Появится поле со списком макрокоманд. Выберите макрокоманду: УстановитьСообщения. В нижней части окна появится аргумент этой команды: Нет. Оставьте его без изменения. Введите в графу Примечание краткий комментарий: Отключение системных и предупреждающих сообщений.

Перейдите на следующую строку и выберите для нее макрокоманду Открыть Запрос. Определите аргументы макрокоманды в нижней части окна. Раскройте список в поле Имя запроса и выберите в нем имя Запрос на создание экзаменационной ведомости. Сохраните значение аргументов Режим - таблица и Режим данных - изменение. Введите комментарий к этой макрокоманде: на создание экзаменационной ведомости. При выполнении данной макрокоманды будет создана таблица Ведомость 1.

После того, как запрос на создание ведомости отработал, его можно закрыть, поэтому в третьей строке выберите макрокоманду Закрыть. Определите аргументы макрокоманды: Тип объекта - Запрос, Имя объекта - Запрос на создание экзаменационной ведомости, СохранениеПодсказка. Введите комментарий: Закрытие запроса.

Перед открытием созданной ранее Формы для макроса необходимо произвести копирование таблицы Ведомость1 в другую таблицу, которая и будет использована для открытия формы - Ведомость 00. Для этого выберите макрокоманду КопироватьОбъект. Определите аргументы макрокоманды: Новое имя - Ведомость 00, Тип объекта - таблица, Имя объекта - Ведомость 1. Введите комментарий: копирование таблицы Ведомость 1 в Ведомость 00.

Примечание. При определении для макрокоманды в качестве аргументов имен объектов

совершенно не обязательно, чтобы соответствующие объекты существовали в базе данных в момент конструирования макроса. Важно, чтобы они были в базе данных к моменту выполнения этой макрокоманды.

В следующей строке выберите макрокоманду ОткрытьФорму. Определите аргументы макрокоманды: Имя формы - Форма для макроса, Режим - Форма, Режим окна - Обычное. Введите комментарий: Открыть форму на основе таблицы Ведомость 00.

Следующая макрокоманда должна осуществить переход на первую строку таблицы формы для проверки значений полей N группы и Код дисциплины, введенных при выполнении запроса в диалоговые окна. Поэтому в очередной строке выберите макрокоманду НаЗапись.

Выберите из списка для аргумента Запись значение Первая. Введите комментарий: Перейти на 1-ую запись табличной формы.

Выберите для следующей строки макрокоманду Переименовать для переименования ведомости для группы 851 и дисциплины с кодом 1. Определите аргументы макрокоманды: Новое имя - Ведомость 851_1, Тип объекта - Таблица, Старое имя - Ведомость 1. В графу Условие введите с помощью построителя для этой строки выражение :

[Формы]![Форма для макроса]![N группы]=851 And [Формы]![Форма для макроса]![Код дисциплины]=1

Введите комментарий: Переименовать Ведомость 1 в Ведомость 851_1

Введите следующую макрокоманду для переименования ведомости для группы 851 и дисциплины с кодом 2. Определите аргументы макрокоманды: Новое имя - Ведомость

851_2, Тип объекта - Таблица, Старое имя - Ведомость 1. Введите Условие:

[Формы]![Форма для макроса]![N группы]=851 And [Формы]![Форма для макроса]![Код дисциплины]=2

Примечание. Для ввода условий и комментариев к макрокомандам целесообразно использовать приемы копирования в буфер, а затем вставки с последующей корректировкой в окне области ввода, открываемом при нажатии сочетания клавиш SHIFT и F2.

Введите следующие строки макроса по аналогии с двумя предыдущими для переименования ведомости для группы 851 и дисциплины с кодом 3, а также для других групп и дисциплин. При 3-х дисциплинах должно быть 3 макрокоманды на одну группу студентов.

Введите последнюю макрокоманду Закрывать с аргументами: Тип объекта - Форма, Имя объекта - Форма для макроса, Сохранение - Подсказка. Окончательный вид макроса представлен на рис. 7.1

Сохраните макрос, щелкнув по кнопке Сохранить инструментальной панели, под именем Макрос для создания ведомостей. Запустите макрос на выполнение в пошаговом режиме.

Для этого, находясь в режиме конструктора щелкните по кнопке инструментальной панели По шагам, а затем по кнопке Запуск. Проследите по шагам правильность исполнения макрокоманд.

В случае неверного выполнения или аварийного завершения найдите ошибку и исправьте макрос.

После отладки макроса отключите пошаговый режим запуска, повторно щелкнув по кнопке По шагам, закройте макрос. Произведите запуск макроса из окна базы данных. Для этого в окне База данных выберите объект Макросы, выделите Макрос для создания ведомостей и щелкните по кнопке Запуск на инструментальной панели окна. Запустите макрос для формирования экзаменационных ведомостей по разным группам и дисциплинам.

Задание 2.1.5 Практическая работа

Проверяемые результаты обучения: ОК01, ОК02, ПК11.2, ПК11.3

Цель: ознакомиться с основными конструкциями для резервного копирования БД

ХОД РАБОТЫ

Задание 1. необходимо создать резервные копии базы данных «МММ» с

использованием полного резервного копирования, разностного резервного копирования и резервного копирования журнала транзакций.

1. Запустите SQL Server Management Studio (SSMS), подключитесь к своему экземпляру SQL Server, используя технологию 1.

2. Создайте папку с именем c:\Student\ВашаПапка\test.

3. Откройте окно нового запроса. Измените контекст на базу данных master, используя технологию 6. Наберите и выполните следующую команду, чтобы создать полную резервную копию базы данных: `BACKUP DATABASE MMM TO DISK = 'C:\.....TEST\AW.BAK'`

Ознакомьтесь с результатами запроса – какая информация обработана, сколько страниц, сколько файлов.

4. Внесите изменение в таблицу «Модель» базы данных MMM. Добавьте одну запись (придумайте сами)/

5. Откройте окно нового запроса наберите и выполните следующую команду, чтобы создать резервную копию журнала транзакций и сохранить только что внесенное изменение: `BACKUP LOG MMM TO DISK = 'C:\.....TEST\AW1.TRN'`

Ознакомьтесь с результатами запроса – какая информация обработана, сколько страниц, сколько файлов.

6. Внесите еще одно изменение в таблицу «Модель».

7. Откройте окно нового запроса наберите и выполните следующую команду, чтобы создать разностную резервную копию базы данных:

`BACKUP DATABASE MMM TO DISK = 'C:\.....\TEST\AWDIFF1.BAK' WITH DIFFERENTIAL`

Ознакомьтесь с результатами запроса – какая информация обработана, сколько страниц, сколько файлов.

8. Внесите еще одно изменение в таблицу «Модель».

9. Откройте окно нового запроса наберите и выполните следующую команду, чтобы создать полную резервную копию базы данных в указанном месте на диске:

`BACKUP LOG MMM TO DISK = 'C:\....TEST\AW2.TRN'`

Ознакомьтесь с результатами запроса – какая информация обработана, сколько страниц, сколько файлов.

Задание 2. Необходимо организовывать со стороны клиентского приложения, созданного в Visual Studio удаленное администрирование БД (резервное копирование).

Ход работы:

1. Создайте новый проект Windows Application и сохраните его в своей папке под именем Лабы_MMM_2 семестр.

2. В главную форму добавьте меню - Файл (Открыть, Закрыть, Выход)

Справочники (Модель, Магазин, Дерево моделей)

Заказы (Работа с заказами)

Отчеты (Прайс-лист, Бланк заказов)

Администрирование БД (Резервное копирование, Безопасность)

Сервис (Калькулятор)

Помощь (Справка, О программе)

3. Добавьте новую форму в проект

4. Добавьте на только что созданную форму компоненты

5. Обеспечьте функциональную работу формы (напишите обработчик кнопки «Резервное копирование» с использованием объектов SMO. Описание объектов SMO, их свойств и методов см. в лекционном материале.)

5. Добавьте возможность открытия данной формы при выборе в главной форме пункта меню Администрирование БД □ Резервное копирование

6. Запустите проект, проверьте работу формы.

7. Закройте проект

8. Убедитесь в появлении файла резервной копии на диске (файл, который указан в

тексте программы).

9. Откройте SSMS. Добавьте в таблицу «Модель» новую строку данных (самостоятельно).

10. Средствами оболочки SSMS, выполните восстановление БД из резервной копии, созданной вашей программой

11. Убедитесь, что после восстановления добавленных строк в таблице «Модель» нет

Задание 2.1.6. Практическая работа

Проверяемые результаты обучения: *OK01, OK02, ПК11.2, ПК11.3*

Цель: изучение способов доступа пользователей к базе данных

ТЕОРЕТИЧЕСКИЙ МАТЕРИАЛ

Понятие пользователь базы данных относится к базе (или базам) данных, к которым может получить доступ отдельный пользователь. После успешного подключения сервер определяет, имеет ли этот пользователь разрешение на работу с базой данных, к которой обращается.

Единственным исключением из этого правила является пользователь guest (гость). Особое имя пользователя guest разрешает любому подключившемуся к SQL Server пользователю получить доступ к этой базе данных. Пользователю с именем guest назначена роль public.

Права доступа

Для управления правами доступа в SQL Server используются следующие команды:

1. GRANT. Позволяет выполнять действия с объектом или, для команды — выполнять ее;

2. REVOKE. Аннулирует права доступа для объекта или, для команды — не позволяет выполнить ее;

3. DENY. Не разрешает выполнять действия с объектом (в то время, как команда REVOKE просто удаляет эти права доступа).

Объектные права доступа позволяют контролировать доступ к объектам в SQL Server, предоставляя и аннулируя права доступа для таблиц, столбцов, представлений и хранимых процедур.

Чтобы выполнить по отношению к некоторому объекту некоторое действие, пользователь должен иметь соответствующее право доступа. Например, если пользователь хочет выполнить оператор SELECT * FROM table, то он должен иметь права выполнения оператора SELECT для таблицы table.

Командные права доступа определяют тех пользователей, которые могут выполнять административные действия, например, создавать или копировать базу данных.

Ниже приведены командные права доступа:

CREATE DATABASE — право создания базы данных;

CREATE DEFAULT — право создания стандартного значения для столбца таблицы;

CREATE PROCEDURE — право создания хранимой процедуры.

CREATE ROLE — право создания правила для столбца таблицы;

CREATE TABLE — право создания таблицы;

CREATE VIEW — право создания представления;

BACKUP DATABASE — право создания резервной копии;

BACKUP TRANSACTION — право создания резервной копии журнала транзакций.

Роли

Назначение пользователю некоторой роли позволяет ему выполнять все функции, разрешенные этой ролью. По сути роли логически группируют пользователей, имеющих одинаковые права доступа.

Роли, определяемые пользователем, позволяют группировать пользователей и назначать каждой группе конкретную функцию безопасности. Существуют два типа ролей уровня базы

данных, определяемых пользователем:

1. стандартная роль;
2. роль уровня приложения.

Стандартная роль предоставляет зависящий от базы данных метод создания определяемых пользователем ролей. Самое распространенное назначение стандартной роли — логически

сгруппировать пользователей в соответствии с их правами доступа. Например, в приложениях

выделяют несколько типов уровней безопасности, ассоциируемых с тремя категориями пользователей. Опытный пользователь может выполнять в базе данных любые операции; обычный

пользователь может модифицировать некоторые типы данных и обновлять данные; неквалифицированному пользователю обычно запрещается модифицировать любые типы данных.

Роль уровня приложения позволяет пользователю выполнять права некоторой роли. Когда пользователь принимает роль уровня приложения, он берет на себя выполнение новой роли

и временно отказывается от всех других назначенных ему прав доступа к конкретной базе данных. Роль уровня приложения имеет смысл применять в среде, где пользователи делают запросы и модифицируют данные с помощью клиентского приложения.

Управление разрешениями на объекты реляционной базы данных несколько отличается от аналогичных операций в отношении объектов файловой системы. В данной лабораторной

работе на примере СУБД SQL Server эти отличия будут показаны.

При работе с разрешениями в SQL Server используется понятие участников (principals), которые могут запрашивать ресурсы SQL Server, и которым могут предоставляться разрешения

на использование таких ресурсов. Выделяются следующие группы участников:

- участники уровня Windows, к которым относятся локальные и доменные учетные записи

пользователей и группы;

- участники уровня SQL Server, к которым относятся учетные записи SQL Server и роли уровня сервера;

- участники уровня базы данных — пользователи базы данных и роли уровня базы данных

и приложения.

Необходимо отметить, что SQL Server разделяет понятие учетной записи (login) и пользователя (user). Сервер может быть сконфигурирован на использование только аутентификации

Windows (англ. Windows Authentication Mode, используется по умолчанию) или на использование смешанного режима аутентификации (англ. SQL Server and Windows Authentication mode).

В первом случае login можно создать только для пользователя или группы Windows. Во втором

случае также возможно использовать собственные учетные записи SQL Server — логин и пароль хранятся самой СУБД, и ее же средствами выполняется проверка подлинности. При использовании смешанной аутентификации становится доступной административная учетная запись sa, которую рекомендуется переименовать и назначить ей надежный пароль. Учетная запись авторизуется на выполнение одной из серверных ролей

Роли уровня сервера

Роль	Описание возможностей
sysadmin	Разрешено выполнять любые действия на сервере.
dbcreator	Разрешено создавать базы данных.
bulkadmin	Могут выполнять инструкцию BULK INSERT.
diskadmin	Позволяет управлять файлами на диске.
processadmin	Позволяет управлять подключениями, запускать и приостанавливать экземпляры SQL Server.
securityadmin	Создание и управление учетными записями, право «сбросить» пароль учетной записи. Управление разрешениями на уровне сервера и на уровне базы данных (при наличии доступа к базе данных).
serveradmin	Включает возможности ролей diskadmin и processadmin, позволяет изменять параметры конфигурации на уровне сервера и выключать сервер.
setupadmin	Добавление и удаление связанных серверов.
public	Каждая учетная запись принадлежит этой роли, членство в роли public изменить нельзя.

На уровне базы данных учетной записи сопоставляется пользователь (user). Для одной и той же учетной записи в различных базах данных сервера могут создаваться пользователи с разными именами.

Пользователи получают разрешения на работу с объектами базы данных или напрямую, или путем авторизации пользователя на выполнение одной из ролей уровня базы данных. Последний способ является более предпочтительным и позволяет организовать управление доступом в соответствии с ролевой моделью, описанной в первой главе пособия.

Список ролей уровня сервера предопределен, и новые создавать нельзя (табл. 2). Также есть предопределенный набор ролей уровня базы данных (табл. 3), но в этом случае имеется возможность создавать новые роли.

Роли уровня базы данных

Роль	Описание возможностей
db_owner	Владелец базы данных, можно выполнять все действия по настройке и обслуживанию базы данных, а также удалять базу данных.
db_securityadmin	Управление составом ролей (кроме роли db_owner) и связанными с ними разрешениями.
db_accessadmin	Добавление и удаление пользователей базы данных.
db_backupoperator	Возможность создавать резервные копии базы данных.
db_ddladmin	Выполнение DDL-инструкций (создание, изменение, удаление объектов базы данных, таких как таблицы, представления и т. д.).
db_datareader	Чтение данных (SELECT) из всех пользовательских таблиц, представлений и функций.
db_denydatareader	Запрет на чтение данных (SELECT) из всех пользовательских таблиц, представлений и функций.
db_datawriter	Право добавлять, удалять или изменять данные во всех пользовательских таблицах.
db_denydatawriter	Запрет добавлять, изменять или удалять данные в пользовательских таблицах.
public	Роль по умолчанию, имеющаяся в каждой базе данных. Каждый пользователь БД авторизован на эту роль.

Более подробную информацию об организации системы безопасности СУБД SQL Server можно получить из справочной системы TechNet: <http://technet.microsoft.com/ruru/library/bb510589.aspx>.

ХОД РАБОТЫ

1. Используя указанную преподавателем доменную или локальную учетную запись Windows, с помощью SQL Server Management Studio подключитесь к используемому экземпляру SQL Server. Проверьте установленный на сервере режим аутентификации.

2. В окне Object Explorer (по умолчанию — левая часть окна Management Studio) откройте список учетных записей (logins). На выполнение каких серверных ролей авторизована используемая вами учетная запись?

3. В каких базах данных сервера вашей учетной записи сопоставлены пользователи? На выполнение каких ролей они авторизованы?

4. В среде Management Studio создайте новую базу данных. Откройте список пользователей и ролей. Убедитесь, что учетная запись, под которой вы работаете, сопоставлена пользователю dbo, авторизованному на роль db owner.

5. Используя приведенный ниже скрипт, создайте в базе данных таблицы. Перед тем как запустить скрипт, уберите символы комментария («--») из первой строки и после ключевого слова use укажите имя вашей базы данных.

```
use MyTest1 GO
CREATE TABLE dbo.Book (
book_id int IDENTITY (1, 1) primary key,
Title varchar(50) NOT NULL, —название книги Author varchar(50), — автор Publisher
varchar(50), — издательство [Year] smallint) — год издания GO
CREATE TABLE dbo.Status (
Status_id int IDENTITY (1, 1) primary key, Status_name varchar(50) NOT NULL ) —
статус: выдана, в библиотеке и т.д.
GO
CREATE SCHEMA libr GO
CREATE TABLE libr.Book_in_lib (
lib_id int primary key , —номер экземпляра book_id int references dbo.Book , status_id int
references dbo.[Status])
```

Обратите внимание, что приведенный скрипт создает не только три таблицы, но и схему libr. В SQL Server схема является контейнером логического уровня, к которому относятся объекты базы данных. Во вновь созданной БД уже будет несколько схем: dbo, sys, information_schema и т. д. Схема dbo — это схема по умолчанию для новых пользовательских объектов, sys и information_schema используются системными объектами. Оператором CREATE SCHEMA в БД можно создавать новые схемы.

Защищаемым объектом, на действия с которым пользователю предоставляются разрешения, может быть база данных, схема или объект базы данных. Определенное для схемы разрешение неявным образом распространяется на все объекты схемы, разрешение для базы данных - на все схемы и объекты этих схем.

6. Для указанной преподавателем учетной записи SQL Server (при самостоятельном выполнении работы создайте учётную запись Windows и учётную запись SQL Server для нее) создайте пользователя в вашей базе данных, в качестве схемы по умолчанию выберите dbo. В

Management Studio это можно сделать из графического интерфейса (контекстное меню узла Security для выбранной БД, там New...-> User) или выполнив оператор CREATE USER. Например (если схема не указана, подразумевается dbo): USE MyTest1 до CREATE USER ns FOR LOGIN [HOME s]

Добавьте этого пользователя в роль db_datareader. Это можно сделать или через графический интерфейс или с помощью системной хранимой процедуры sp_addrolemember, первым параметром которой будет имя роли, а вторым — имя пользователя.

```
EXEC sp_addrolemember 'db_datareader', 'ns 1
```

Введите в таблицы тестовый набор данных. Подключитесь к серверу с учетной записью другого пользователя. Убедитесь, что можно получить доступ к базе данных и читать записи из всех таблиц, а добавлять или изменять данные нельзя.

7. Создадим новую роль уровня базы данных и добавим ей разрешение на удаление (DELETE), изменение (UPDATE) и добавление данных (INSERT) в объектах схемы libr.

Добавим нашего пользователя к этой роли. Указанные действия надо выполнять с правами администратора или владельца базы данных. Как и в предыдущем случае, все это можно сделать в графическом интерфейсе или запуском скрипта.

```
CREATE ROLE libr_writer GO
GRANT INSERT, UPDATE, DELETE ON SCHEMA :: libr TO
libr_writer
Go
EXEC sp_addrolemember 'libr_writer', 'ns'
```

Используемый в приведенном скрипте оператор GRANT позволяет предоставить разрешения. Оператор DENY позволяет запретить выполнение каких-то действий, а оператор REVOKE отменяет установленные оператором GRANT или DENY настройки разрешений. Таким образом, у разрешения может быть три состояния: «разрешено», «запрещено», «не задано».

Действие можно выполнить, только если оно разрешено непосредственно пользователю или одной из ролей, на которые он авторизован. Запрещение более приоритетно, чем разрешение: если пользователь авторизован на выполнение двух ролей, одной из них действие разрешено, а другой - запрещено, то пользователь это действие выполнить не сможет. В SQL Server

Management Studio можно просмотреть эффективные разрешения для пользователя.

Конкретный набор возможных разрешений зависит от типа объекта.

Выполните описанные действия. Убедитесь, что пользователь с ограниченными правами может изменять данные в таблице Book_in_lib, относящейся к схеме libr

8. Иногда нужно предоставить пользователю права на изменение отдельных столбцов. Как отмечается в документации SQL Server, на столбец могут быть предоставлены только разрешения SELECT, REFERENCES и UPDATE. Например: GRANT UPDATE ON dbo.Book(Title) TO libr_writer. Выполните аналогичные действия в своей базе данных, проверьте, что пользователь получил указанные разрешения. 9. Самостоятельно по справке ознакомьтесь с форматом оператора CREATE VIEW, особое внимание обратите на задаваемые дополнительные параметры. Создайте представление, выбирающее из таблицы Book книги, изданные не ранее 2000 года. Предоставьте пользователю с ограниченными правами возможность изменять и добавлять подобные книги. Возможности изменять прочие записи таблицы и добавлять книги, изданные до 2000 года, он иметь не должен

Критерии оценки:

Оценка «отлично» – практические задачи выполнены в полном объеме, студент отвечает на все поставленные вопросы. Все задания выполнены правильно. Работа оформлена аккуратно и в соответствии с требованиями. Продемонстрировано глубокое понимание материала.

Оценка «хорошо» – студент допускает незначительные неточности, правильно применены теоретические знания. Задания выполнены с небольшими ошибками. Оформление работы имеет несущественные недочеты. Продемонстрировано хорошее понимание материала

Оценка «удовлетворительно» – отсутствие полного объема работ; низкое качество выполнения работ, часть заданий выполнена с ошибками; Оформление работы имеет существенные недочеты; продемонстрировано базовое понимание материала

Оценка «неудовлетворительно» – отсутствие полного объема работ; в работе допущены серьезные ошибки и нарушение всех перечисленных выше требований; отсутствует понимание материала.

Задание 2.1.7. Тестирование

Формируемые компетенции: ОК1, ОК2, ОК9, ПК11.2, ПК 11.3

Цель тестового задания - контроль знаний освоения дисциплины, получение ответа от испытуемого, на основе которого может быть сделан вывод о его знаниях, представлениях из определенной области содержания дисциплины.

Задание: перечень вопросов, соответствующих содержанию дисциплины.

Инструкция: выберите один/несколько правильных ответов из предложенных

1. Прочитайте текст, выберите все правильные ответы и запишите аргументы, обосновывающие выбор ответа

Как называется центральное место, в котором хранятся консолидированные данные из нескольких баз данных?

- А) банк данных;
- Б) хранилище данных;
- В) информационная система;
- Г) СУБД.

Ответ:

2. Прочитайте текст, выберите все правильные ответы и запишите аргументы, обосновывающие выбор ответа

Какая модель создаётся без ориентации на какую-либо конкретную СУБД?

- А) концептуальная модель;
- Б) внешняя модель;
- В) логическая модель;
- Г) внутренняя модель.

3. Прочитайте текст и установите соответствие

Сопоставьте виды моделей и их краткие характеристики

К каждой позиции, данной в левом столбце, подберите соответствующую позицию из правого столбца:

А	Описывает реализацию модели данных, специфичную для конкретной базы данных	1	Концептуальная модель
Б	Устанавливает сущности, их атрибуты и взаимосвязи	2	Логическая модель
В	Определяет структуру элементов данных и устанавливает связи между ними	3	Физическая модель
		4	Оптимизационная модель

Запишите выбранные цифры под соответствующими буквами:

А	Б	В

4. Прочитайте текст и установите соответствие

Сопоставьте предложенные характеристики нормальных форм БД и названия форм.

К каждой позиции, данной в левом столбце, подберите соответствующую позицию из правого столбца:

А	предполагает, что сохраняемые данные на пересечении строк и столбцов должны представлять скалярное значение, а таблицы не должны содержать повторяющихся строк	1	Вторая нормальная форма (2NF)
Б	предполагает, что каждый столбец, не являющийся ключом, должен зависеть от первичного ключа.	2	Шестая нормальная форма (6NF)
В	предполагает, что каждый столбец, не являющийся ключом, должен зависеть только от первичного ключа.	3	Четвертая нормальная форма (4NF)

Г	применяется для устранения многозначных зависимостей (multivalued dependencies) – таких зависимостей, где столбец с первичным ключом имеет связь один-ко-многим со столбцом, который не является ключом.	4	Третья нормальная форма (3NF)
		5	Первая нормальная форма (1NF)

Запишите выбранные цифры под соответствующими буквами:

А	Б	В	Г

5. Прочитайте текст, выберите все правильные ответы и запишите аргументы, обосновывающие выбор ответа

База данных – это:

- А) Набор связанных файлов
- Б Структура для хранения информации
- В) Совокупность взаимосвязанных данных, организованных по определенным правилам
- Г) Программа для работы с информацией

Ответ:

6. Прочитайте текст, выберите все правильные ответы и запишите аргументы, обосновывающие выбор ответа

СУБД – это:

- А) Система для создания баз данных
- Б) Программа для работы с базами данных
- В) Комплекс программ для создания, управления и обработки баз данных
- Г) Набор инструментов для работы с таблицами

Ответ:

7. Прочитайте текст, выберите все правильные ответы и запишите аргументы, обосновывающие выбор ответа

Ключевое поле должно быть:

- А) Обязательно числовым
- Б) Уникальным
- В) Не должно содержать длинные записи
- Г) Обязательно счетчиком

Ответ:

8. Прочитайте текст, выберите все правильные ответы и запишите аргументы, обосновывающие выбор ответа

Записями в базе данных считаются:

- А) Заголовки
- Б) Строки
- В) Столбцы
- Г) Таблицы

Ответ:

9. Прочитайте текст, выберите все правильные ответы и запишите аргументы, обосновывающие выбор ответа

Какой тип данных выбрать для ввода суммы 57\$?

- А) Числовой
- Б) Финансовый
- В) Денежный
- Г) Текстовый

Ответ:

10. Прочитайте текст, выберите все правильные ответы и запишите аргументы, обосновывающие выбор ответа

Условие поиска и отбора записей в таблице называется:

- А) Фильтр
- Б) Запрос
- В) Форма
- Г) Критерий поиска

Ответ:

11. Прочитайте текст, выберите все правильные ответы и запишите аргументы, обосновывающие выбор ответа

Для чего используется ключевое поле?

- А) Сортировка данных
- Б) Фильтрация данных
- В) Создание новых таблиц
- Г) Создание связей между таблицами

Ответ:

12. Прочитайте текст, выберите все правильные ответы и запишите аргументы, обосновывающие выбор ответа

Структура таблицы определяется с помощью колонок:

- А) Имя поля
- Б) Тип данных
- В) Описание
- Г) Размер поля

Ответ:

13. Прочитайте текст, выберите все правильные ответы и запишите аргументы, обосновывающие выбор ответа

Первичный ключ таблицы – это:

- А) Строка с уникальной информацией
- Б) Столбец с неуникальной информацией
- В) Совокупность полей, однозначно определяющих запись
- Г) Столбец с уникальной информацией

Ответ:

14. Прочитайте текст, выберите все правильные ответы и запишите аргументы, обосновывающие выбор ответа

Базы данных со связанными таблицами называются:

- А) Реляционными
- Б) Связанными
- В) Структурными
- Г) Простейшими

Ответ:

15. Прочитайте текст, выберите все правильные ответы и запишите аргументы, обосновывающие выбор ответа

Как называется набор инструментальных средств, позволяющих в наглядной форме моделировать предметную область, анализировать эту модель на всех стадиях разработки и сопровождения ИС, и разрабатывать приложения в соответствии с информационной потребностью пользователя?

- А) RAD-средство;
- Б) CASE-средство;
- В) IDEF3 средство.

Ответ:

16. Ответьте на вопрос

Необходимо ответить на поставленный вопрос. Вопрос с развернутым ответом.

1. Опишите основные типы данных, используемые в базах данных. Для каких целей применяется каждый тип?

Ответ

17. Ответьте на вопрос

Необходимо ответить на поставленный вопрос. Ответ должен быть кратким

1. Как называется поле, которое однозначно определяет запись в таблице?

Ответ

18. Прочитайте текст и установите последовательность

Установите правильную последовательность этапов создания базы данных:

- А) Определение структуры таблиц
- Б) Создание связей между таблицами
- В) Проектирование схемы данных
- Г) Создание таблиц
- Д) Ввод данных

Запишите соответствующую последовательность цифр слева направо

--	--	--	--	--

19. Прочитайте текст и установите последовательность

Цель создания запроса в базе данных - точно указать, какие данные необходимы и как их следует извлекать или изменять.

Запрос позволяет найти информацию в таблицах, просмотреть её в одной или отфильтровать данные, чтобы отображались только нужные записи. Также с помощью запросов можно вычислить или свести данные, автоматизировать управление данными, например, регулярно просматривать актуальные данные.

Расположите этапы создания запроса по порядку:

- А) Указание условий отбора

- Б) Выбор типа запроса
- В) Выбор таблиц для запроса
- Г) Определение полей для вывода
- Д) Выполнение запроса

Запишите соответствующую последовательность цифр слева направо

--	--	--	--	--

20. Прочитайте текст и установите последовательность

Форма создания запроса является важным инструментом для эффективной работы с базой данных, обеспечивая гибкость и удобство при выполнении различных операций с данными. С помощью формы запроса можно просматривать информацию из определённых полей таблицы или из нескольких таблиц одновременно, осуществить быстрый поиск определённых данных путём фильтрации с применением критериев (условий), найти информацию, соответствующую конкретным параметрам, выполнить различные вычисления над данными и т.д. В задании Вам необходимо:

Установите последовательность действий при создании формы:

- А) Выбор полей для формы
- Б) Запуск мастера форм
- В) Настройка внешнего вида
- Г) Выбор типа формы
- Д) Предварительный просмотр

Запишите соответствующую последовательность цифр слева направо

--	--	--	--	--

21. Прочитайте текст и установите последовательность

Нормализация базы данных - это процесс организации данных в базе определённым образом в соответствии с рекомендациями по проектированию. То есть все таблицы, а также необходимые связи между ними создаются по определённым правилам. Основными целями нормализации являются: устранение избыточности — явления, при котором одни и те же записи хранятся в нескольких местах базы; уменьшение размера базы; сохранение целостности данных при их изменении.

В процессе нормализации данные преобразуют, чтобы они занимали меньше места, а поиск по элементам был быстрым и результативным. Для этого создают дополнительные таблицы и связывают друг с другом ключами - колонками, в которых нет повторяющихся элементов. В предложенном задании Вам необходимо:

Расположите этапы нормализации базы данных в правильном порядке:

- А) Удаление транзитивных зависимостей
- Б) Приведение к первой нормальной форме
- В) Удаление частичных зависимостей
- Г) Приведение ко второй нормальной форме
- Д) Приведение к третьей нормальной форме

Запишите соответствующую последовательность цифр слева направо

--	--	--	--	--

22. Ответьте на вопрос

Необходимо ответить на поставленный вопрос. Ответ должен быть кратким

1. Укажите порядок выполнения операций при поиске данных:

Ответ

23. Прочитайте текст, выберите все правильные ответы и запишите аргументы, обосновывающие выбор ответа

Как называется атрибут или набор атрибутов, уникально идентифицирующий единственный экземпляр сущности?

- А) первичный ключ;
- Б) внешний ключ;
- В) основной ключ;
- Г) уникальный ключ.

Ответ:

Обоснование

24. Ответьте на вопрос

Необходимо ответить на поставленный вопрос. Ответ должен быть кратким

1. Укажите английский термин для обозначения «внешнего ключа».

Ответ

25. Прочитайте текст, выберите все правильные ответы и запишите аргументы, обосновывающие выбор ответа

Какая структура данных используется для оптимизации поиска?

- А) Индекс
- Б) Таблица
- В) Представление
- Г) Хранимая процедура

Ответ:

Ключ к тестовым заданиям

№ задания	Верный ответ	Критерии
1	Б	1б – полное правильное соответствие 0 б – остальные случаи
2	А	1б – полное правильное соответствие 0 б – остальные случаи
3	1Б2В3А	1б – полное правильное соответствие 0 б – остальные случаи
4	1Б5А3Г4В	1б – полное правильное соответствие 0 б – остальные случаи
5	В	1б – полное правильное соответствие 0 б – остальные случаи
6	В	1б – полное правильное соответствие 0 б – остальные случаи
7	Б	1б – полное правильное соответствие 0 б – остальные случаи
8	Б	1б – полное правильное соответствие 0 б – остальные случаи
9	В	1б – полное правильное соответствие 0 б – остальные случаи
10	А	1б – полное правильное соответствие 0 б – остальные случаи

11	Г	1б – полное правильное соответствие 0 б – остальные случаи
12	Г	1б – полное правильное соответствие 0 б – остальные случаи
13	В	1б – полное правильное соответствие 0 б – остальные случаи
14	А	1б – полное правильное соответствие 0 б – остальные случаи
15	Б	1б – полное правильное соответствие 1 б – остальные случаи
16	1. Числовые типы. Применяются для хранения чисел, которые могут быть целыми или дробными; 2. Строковые типы. Используются для хранения текстовой информации. В зависимости от длины и характера текста, могут использоваться разные типы данных; 3. CHAR для строк фиксированной длины. Подходит для хранения данных, длина которых всегда одинакова, например, кодов стран или почтовых индексов; 4. VARCHAR для строк переменной длины. Используется для хранения имён пользователей, адресов или комментариев, где длина текста у каждого пользователя будет разной; 5. TEXT для больших объёмов текста. Подходит для хранения описаний, комментариев и других текстовых данных, длина которых может быть значительной	1б – полное правильное соответствие 0 б – остальные случаи
17	Ключевое поле	1б – полное правильное соответствие 0 б – остальные случаи
18	В-А-Г-Б-Д	1б – полное правильное соответствие 0 б – остальные случаи
19	Б-В-Г-А-Д	1б – полное правильное соответствие 0 б – остальные случаи
20	Б-Г-А-В-Д	1б – полное правильное соответствие 0 б – остальные случаи
21	Б-Г-В-А-Д	1б – полное правильное соответствие 0 б – остальные случаи
22	Открытие базы данных → Выбор таблицы → Установка критериев поиска → Выполнение запроса → Просмотр результатов	1б – полное правильное соответствие 0 б – остальные случаи

23	Г	1б – полное правильное соответствие 0 б – остальные случаи
24	Foreign key	1б – полное правильное соответствие 0 б – остальные случаи
25	А	1б – полное правильное соответствие 0 б – остальные случаи

Критерии оценки:

соответствие ответов обучающихся ключу теста

Оценка «**отлично**» - если обучающийся правильно выполнил от 80% до 100% тестовых заданий в отведенное время

Оценка «**хорошо**» - если обучающийся правильно выполнил от 60% до 80% тестовых заданий в отведенное время

Оценка «**удовлетворительно**» - если обучающийся правильно выполнил от 40% до 60% тестовых заданий в отведенное время

Оценка «**неудовлетворительно**» ставится в случае выполнения менее 40% тестовых заданий

Время выполнения: 35-40 минут

3. Вопросы для самоконтроля и подготовки к зачету ОК01, ОК02, ПК1.1, ПК1.2, ПК1.3, ПК1.4, ПК11.2, ПК11.3

1. Базы данных и информационные системы.
2. Основные определения БД.
3. Этапы развития технологий обработки данных.
4. Системы управления базами данных.
5. Основные функции СУБД.
6. Технологии работы с БД.
7. Архитектура базы данных.
8. Физическая и логическая независимость.
9. Логическая и физическая независимость данных.
10. Понятие модели данных.
11. Теоретико-графовые модели данных.
12. Иерархическая модель БД.
13. Сетевая модель БД.
14. Реляционная модель.
15. Многомерная модель данных.
16. Объектно-ориентированная модель.
17. Типы моделей данных.
18. Реляционная модель данных.
19. Особенности реляционной модели данных: основные понятия и компоненты, свойства отношений.
20. Основы реляционной алгебры.
21. Индексирование.
22. Связывание таблиц.
23. Понятие ссылочной целостности.
24. Принципы поддержки целостности в реляционной модели данных.
25. Реляционная алгебра.
26. Основные понятия реляционной алгебры.
27. Замкнутость реляционной алгебры
27. Чем отличается мониторинг трафика от фильтрации?
28. Ограничения на операции.
29. Операции реляционной алгебры.
30. Основные этапы проектирования БД.
31. Задачи и основные этапы проектирования баз данных.
32. Анализ предметной области.
33. Концептуальное проектирование БД.
34. Концептуальное моделирование.
35. Логическое проектирование и физическая модель баз данных.
36. Нормализация БД.
37. Нормальные формы: первая нормальная форма, вторая нормальная форма, третья нормальная форма, нормальная форма Бойса-Кодда, четвертая нормальная форма, пятая нормальная форма, доменно-ключевая нормальная форма, шестая нормальная форма.
38. Средства проектирования структур БД.
39. Роль проектирования данных в жизненном цикле информационных систем.
40. Составные части процесса проектирования данных.
41. Наиболее популярные средства проектирования данных.
42. Организация интерфейса с пользователем.
43. Разработка пользовательских интерфейсов.
44. Организация интерфейса с пользователем.
45. Основные понятия языка SQL.
46. Синтаксис операторов, типы данных.

47. Введение в язык SQL.
48. Работа с таблицами.
49. Ограничения целостности.
50. Выборка данных.
51. Изменение данных.
52. Создание, модификация и удаление таблиц.
53. Операторы манипулирования данными.
54. Хранимые процедуры и триггеры.
55. Работа с индексами.
56. Генераторы.
57. Организация запросов на выборку данных при помощи языка SQL.
58. Формирование запросов на языке SQL.
59. DML: Команды модификации данных.
60. DML: Выборка данных.
61. DML: Выборка из нескольких таблиц.
62. DML: Вычисления внутри SELECT.
63. Организация запросов на выборку данных при помощи языка SQL.
64. DML: Вычисления внутри SELECT.
65. Использование представлений.
66. Другие возможности SQL.
67. Сортировка и группировка данных в SQL.
68. DML: Группировка данных.
69. DML: Сортировка данных.
70. DML: Операция объединения.
71. Назовите элементы интерфейса программы
72. Сформулируйте технологию ввода кода программы
73. Перечислите этапы алгоритма сохранения и запуска проекта
74. Сформулируйте назначение вкладок «Свойства», «События»
75. Назовите основные свойства компонентов «LABEL», «BUTTON»
76. Назначение целочисленных типов данных
77. Назначение вещественного типа данных
78. Назначение денежного типа данных
79. Назначение вариантного типа данных
80. Назначение символьного типа данных
81. Назначение интервального типа данных
82. Назначение перечисляемого типа данных
83. Основные стандартные математические функции
84. Основные свойства компоненты «EDIT»
85. Формат записи составного оператора
86. Формат записи условного оператора
87. Форма записи оператора варианта
88. Свойства компоненты «TListBox»
89. Назначение компоненты TComboBox.
90. Режимы работы компоненты «Поле со списком».
91. Назначение компоненты TCheckBox.
92. Назначение стандартных панелей сообщений.
93. Назначение компоненты TScrollBar и ее свойства
94. Назначение компоненты TPanel и ее свойства

Критерии оценки

Оценка «зачтено» выставляется обучающемуся, если студент демонстрирует: всестороннее знание основного учебного материала, систематичность знаний, глубокое

понимание предмета, умение свободно выполнять задания по программе. Студент показывает: полное знание учебного материала, способность к самостоятельному пополнению знаний, умение применять знания на практике, успешное выполнение предусмотренных программой заданий. Студент владеет: базовыми знаниями в необходимом объеме, навыками выполнения заданий программы, способен устранять погрешности под руководством преподавателя

Оценка «**незачтено**» выставляется обучающемуся, который не знает значительной части программного материала, допускает существенные ошибки, неуверенно, с большими затруднениями выполняет практические работы. Как правило, оценка «незачтено» ставится студентам, которые не могут продолжить обучение без дополнительных занятий по соответствующей дисциплине.