АРХАНГЕЛЬСКИЙ КОЛЛЕДЖ ТЕЛЕКОММУНИКАЦИЙ (ФИЛИАЛ)
СПбГУТ
(АКТ (ф) СПбГУТ)

Отчеты по лабораторным и практическим работам по МДК.01.01

Студент: Туйкова Анна Евгеньевна
Группа: ИСПП-35

Преподаватель: Маломан Юлия Сергеевна

Архангельск 2024

Лабораторная работа №1
Оценка сложности алгоритмов сортировки

1 Цель работы

1.1 Научиться реализовывать и оценивать сложность алгоритмов сортировки массивов на C#.

2 Контрольные вопросы

2.1 Что такое «массив»?

Массив представляет собой совокупность перемен- ных одного типа с общим для обращения к ним именем.

2.2 Как описывается одномерный массив?

типДанных[] названиеМассива;

2.3 Как обратиться к некоторому элементу одномерного массива?

Для обращения к элементам массива используются индексы.
названиеМассива[индекс];

2.4 Как можно задать одномерный массив?

Явная инициализация задаётся константным массивом. Пример: double[] x= {5.5, 6.6, 7.7};

Создание и инициализация массива в объектном стиле с вызовом конструктора массива. Пример: int[] d= new int[5];

Если массив объявляется без инициализации, то создаётся только висячая ссылка со значением Null.

2.5 Что такое «сортировка»?

Сортировка массива — это упорядочивание набора однотипных данных по возрастанию или убыванию.

2.6 Что такое «алгоритм сортировки»?

Алгоритм сортировки — это алгоритм для упорядочивания элементов в списке.

2.7 Какие виды сортировки массивов существуют?

Сортировка пузырьком (Bubble Sort), шейкерная/коктейльная (Cocktail Sort), сортировка вставками (Insertion Sort), сортировка Шелла (Shell Sort), сортировка выбором (Selection Sort).

3 Вывод

3.1 Научилась реализовывать и оценивать сложность алгоритмов сортировки массивов на C#

Лабораторная работа №2
Оценка сложности алгоритмов поиска

1 Цель работы

1.1 Научиться реализовывать и оценивать сложность алгоритмов поиска элементов массивов на C#.

2 Контрольные вопросы

2.1 Что такое «алгоритм сортировки»?

Алгоритм сортировки — это алгоритм для упорядочивания элементов в списке.

2.2 Какие виды поиска элементов массивов существуют?

Линейный поиск, двоичный поиск, поиск прыжками

2.3 В чем особенность алгоритма линейного поиска и какова его временная сложность?

Поиск значения функции осуществляется простым сравнением очередного рассматриваемого значения (как правило, поиск происходит слева направо, то есть от меньших значений аргумента к большим). Сложность — O(N)

2.4 В чем особенность алгоритма двоичного поиска и какова его временная сложность?

Изначально алгоритм поиска сравнивает искомое значение со средним элементом в массиве. Если значения не равны, то он отбрасывает ту часть массива, в которой целевое значение гарантированно не может находиться. Сложность — O(log(N))

2.5 В чем особенность алгоритма поиска прыжками и какова его временная сложность?

Поиск прыгает через определенное количество индексов и ищет первый элемент, который больше искомого значения, отпрыгивает назад и осуществляет линейный поиск с этого места. Поиск заканчивается, когда был или не был найден элемент. Условие поиска - предварительно сортированный массив по возрастанию. Сложность — O(sqrt(N))

3 Вывод

3.1 Научилась реализовывать и оценивать сложность алгоритмов поиска элементов массивов на C#

Лабораторная работа №3
Оценка сложности рекурсивных алгоритмов

1 Цель работы

1.1 Научиться разрабатывать и оценивать сложность рекурсивных функций в программах на C#.

2 Контрольные вопросы

2.1 Что такое «рекурсия»?

Рекурсией в программировании называется процесс, когда функция (процедура) вызывает сама себя или другие функции и процедуры.

С помощью рекурсии можно сложные процессы представлять через простые.

2.2 Какие проблемы могут возникать при реализации рекурсивных алгоритмов на электронных вычислительных машинах?

При большой глубине рекурсии быстро расходуется стек.

2.3 Какое определение функции может быть названо рекурсивным? Привести примеры.

Рекурсия — это поведение функции, при котором она вызывает сама себя. Примеры рекурсивных функций:

Вычисление факториала числа — последовательных умножений на предыдущее число.

Вычисления с числовыми рядами.

Математические операции, требующие повторяющихся действий с разными значениями.

2.4 Что такое «глубина рекурсии»?

Глубиной рекурсии называется количество последовательных вызовов себя без возврата значения.

2.5 Что такое «рекурсивный спуск»?

Рекурсивный спуск — движение вычислительного процесса в направлении последовательных рекурсивных вызовов.

2.6 Что такое «рекурсивный подъём»?

Рекурсивный подъем — движение вычислительного процесса в направлении возврата из ранее вызванных копий рекурсивной процедуры.

3 Вывод

3.1 Научилась разрабатывать и оценивать сложность рекурсивных функций в программах на C#.

Лабораторная работа №4
Оценка сложности эвристических алгоритмов

1 Цель работы

1.1 Научиться разрабатывать и оценивать сложность рекурсивных функций в программах на C#.

2 Контрольные вопросы

2.1 Что такое «многомерный массив»?

Многомерным называется такой массив, который отличается двумя или более измерениями

2.2 Как описывается двумерный массив?

типДанных[,] имяМассива = new типДанных[количествоСтрок, количествоСтолбцов];

2.3 Как обратиться к некоторому элементу двумерного массива?

Доступ к каждому элементу массива осуществляется с помощью определенной комбинации двух или более индексов.

2.4 Как узнать количество строк двумерного массива?

С помощью метода .GetLength(0)

2.5 Как узнать количество столбцов двумерного массива?

С помощью метода .GetLength(1)

2.6 Как вывести двумерный массив на консоль в виде таблицы?

Перебрать строки и столбцы и отформатировать их по табуляции и переносу строки

3 Вывод

3.1 Научилась разрабатывать и оценивать сложность рекурсивных функций в программах на C#.

Лабораторная работа №5
Работа с классами

1 Цель работы

1.1 Изучить процесс разработки и применения классов на языке C#;

1.2 Изучить реализацию механизма инкапсуляции на языке C#

2 Контрольные вопросы

2.1 Какова общая форма объявления класса в C#?

```
class Имя_класса {
  свойства, методы класса и т.д.
}
```

2.2 На какие виды делятся данные класса в C#?

В C# все типы данных подразделяются на две большие группы — ссылочные типы и типы значений.

2.3 На какие виды делятся функции класса в C#?

Функции что записаны вне классов называют функциями, а функции что записаны внутри классов называются методами.

2.4 Для чего применяются конструкторы классов?

Конструкторы классов выполняют инициализацию объекта, то есть создание нового экземпляра класса, подставляя значения из аргументов конструктора.

2.5 Что такое «цепочка конструкторов»?

Цепочка конструкторов – это процесс вызова одного конструктора из другого по отношению к текущему объекту.

2.6 Для чего применяются свойства классов?

Свойства класса позволяют экземплярам объектов иметь состояние, при котором каждый объект управляет своими собственными данными

2.7 Когда используются автоматически реализуемые свойства классов?

Автоматически реализуемые свойства имеет смысл использовать тогда, когда нет необходимости накладывать какие-либо ограничения на возможные значения неявного поля свойства.

2.8 В чем отличие вызова статические членов классов от членов экземпляра класса?

Считается, что статические члены принадлежат к типам классов, а члены экземпляра принадлежат к объектам (экземплярам типов классов).

3 Вывод

3.1 Изучен процесс разработки и применения классов на языке C#;

3.2 Изучена реализация механизма инкапсуляции на языке C#

Лабораторная работа №6
Перегрузка методов

1 Цель работы

1.1 Изучить процесс выполнения перегрузки методов на языке C#;

1.2 Изучить различные способы передачи параметров в метод

2 Контрольные вопросы

2.1 Что такое «перегрузка методов»?

Возможность создания в классе нескольких методов с одинаковым именем, но с разными параметрами.

2.2 К какому виду полиморфизма относится перегрузка методов?

Перегрузка метода — статический полиморфизм

2.3 Какие типы функций класса можно перегружать?

Перегрузить можно только функции, которые отличаются либо типом, либо числом своих аргументов. Перегрузить функции, которые отличаются только типом возвращаемого значения, нельзя.

2.4 Какие требования предъявляются к сигнатуре перегружаемых функций класса?

В одном классе не должно существовать двух методов с одинаковой сигнатурой

2.5 Какие типы функций класса нельзя перегружать?

Нельзя выполнить перегрузку локальных функций, нельзя перегружать методы, если они отличаются только по типу возвращаемого значения

3 Вывод

3.1 Изучен процесс выполнения перегрузки методов на языке C#;

3.2 Изучены различные способы передачи параметров в метод

Лабораторная работа №8
Определение операций в классе

1 Цель работы

1.1 Изучить процесс определения операций в классе на языке C#.

2 Контрольные вопросы

2.1 Какое ключевое слово определяет операторный метод?

Для перегрузки оператора служит ключевое слово operator, определяющее операторный метод, который, в свою очередь, определяет действие оператора относительно своего класса

2.2 Какие унарные операторы можно перегружать в C#?

Операторы инкремента (++) и декремента (--).

2.3 Какие бинарные операторы можно перегружать в C#?

В языке C# допускается перегружать следующие бинарные операторы:

+ – сложение (a + b)

— – вычитание (a — b)

* – умножение (a * b)

/ – деление (a / b)

2.4 Какие операторы сравнения можно перегружать в C#?

<, > – операторы сравнения (меньше, больше). В классе эти операторы обязательно перегружаются парой (оба)

<=, >= – операторы сравнения (меньше или равно, больше или равно). Эти операторы перегружаются парой

2.5 Какие операторы требуется перегружать в C# попарно?

True/False, операторы сравнения

3 Вывод

3.1 Изучен процесс определения операций в классе на языке C#.

Лабораторная работа №8
Создание наследованных классов

1 Цель работы

1.1 Изучить процесс разработки дочерних классов на языке C#;

1.2 Изучить реализацию механизма наследования на языке C#.

2 Контрольные вопросы

2.1 Что такое «наследование»?

Наследование — механизм объектно-ориентированного программирования (наряду с инкапсуляцией, полиморфизмом и абстракцией), позволяющий описать новый класс на основе уже существующего (родительского), при этом свойства и функциональность родительского класса заимствуются новым классом.

2.2 Сколько родительских классов может быть у дочернего класса в C#?

Классы C# допускают только одиночное наследование других классов, но множественное наследование интерфейсов.

2.3 Какое ключевое слово позволяет обратиться к реализации родительского класса из дочернего?

base

2.4 Что такое «переопределение метода» и как оно выполняется?

Переопределение метода в объектно-ориентированном программировании — одна из возможностей языка программирования, позволяющая подклассу или дочернему классу обеспечивать специфическую реализацию метода, уже реализованного в одном из родительских классов.

2.5 Что такое «абстрактный класс»?

Абстрактный класс — это базовый класс в объектно-ориентированном программировании, который не предполагает создания экземпляров.

Абстрактные классы реализуют на практике один из принципов ООП — полиморфизм. Они могут содержать (и не содержать) абстрактные методы и свойства.

2.6 Для чего предназначены модификаторы virtual, override, abstract, new?

Модификатор new явно скрывает метод базового класса, который имеет такое имя, как и метод производного

Модификатор virtual используется в базовом классе и указывает, что метод может быть переопределен в производном классе.

Модификатор override используется в производном классе и указывает новую реализацию метода, унаследованного от базового класса.

Абстрактный метод создается с помощью модификатора abstract в абстрактном базовом классе.

3 Вывод

3.1 Изучен процесс разработки дочерних классов на языке C#;

3.2 Изучена реализация механизма наследования на языке C#.

Лабораторная работа №9
Работа с объектами через интерфейсы

1 Цель работы

1.1 Изучить процесс разработки и реализации интерфейсов на языке C#.

2 Контрольные вопросы

2.1 Что такое «интерфейс»?

Это схема методов и свойств, которые должен реализовать класс, что позволяет четко разделить определение интерфейса и реализацию в классах.

2.2 Чем отличается интерфейс от абстрактного класса?

Интерфейсы описывают только часть функциональности объекта — определённые признаки.

Интерфейс описывает только поведение (методы), и у него нет полей.

Наследник абстрактного класса обязан наследовать все его составляющие

2.3 Есть ли у класса ограничения по количеству реализуемых интерфейсов?

В классе допускается реализовывать несколько интерфейсов. В этом случае все реализуемые в классе интерфейсы указываются списком через запятую.

2.4 Какова общая форма объявления интерфейсов и их элементов?

```
interface IИмяИнтерфейса {
типВозврата ИмяМетода1 (списокПараметров);
типВозврата ИмяМетода2 (списокПараметров);
// ...
типВозврата ИмяМетодаN (списокПараметров);
}
```

3 Вывод

3.1 Изучен процесс разработки и реализации интерфейсов на языке C#.

Лабораторная работа №10
Использование стандартных интерфейсов

1 Цель работы

1.1 Изучить процесс реализации стандартных интерфейсов на языке C#

2 Контрольные вопросы

2.1 Для чего используется интерфейс IComparable?

IComparable - интерфейс, предназначенный для реализации метода сравнения двух объектов произвольного типа.

2.2 Для чего используется интерфейс IEquatable?

Интерфейс IEquatable служит для определения равенства двух объектов.

2.3 В чем отличие между обобщенным и необобщенным интерфейсами?

Необобщённые интерфейсы — это структуры данных общего назначения, которые оперируют ссылками на объекты.

Они позволяют хранить данные любого типа, причём в одной коллекции допускается наличие разнотипных данных. Такие коллекции не типизированы, поскольку в них хранятся ссылки на данные типа object.

Обобщённые интерфейсы — это интерфейсы, которые объявляют методы с обобщёнными параметрами.

Такие интерфейсы указываются аналогично обобщённым классам.

3 Вывод

3.1 Изучен процесс реализации стандартных интерфейсов на языке C#

Лабораторная работа №11
Коллекции. Параметризованные классы

1 Цель работы

1.1 Изучить процесс создания и применения параметризованных классов коллекций (обобщенных списков и словарей) на языке C#

2 Контрольные вопросы

2.1 Что такое «коллекция»?

Это объединение произвольного количества объектов, возможно, объектов разного типа.

2.2 Какие классы описаны в пространстве имен System.Collections.Generic?

В пространстве имён System.Collections.Generic описаны следующие классы обобщённых коллекций:

Dictionary<Tkey, TValue> — сохраняет пары «ключ-значение».

HashSet<T> — сохраняет ряд уникальных значений, используя хештаблицу.

LinkedList<T> — сохраняет элементы в двунаправленном списке.

List<T> — создаёт динамический массив.

Queue<T> — создаёт очередь.

SortedDictionary<TKey, TValue> — создаёт отсортированный список из пар «ключ-значение».

SortedList<TKey, TValue> — создаёт отсортированный список из пар «ключ-значение».

SortedSet<T> — создаёт отсортированное множество.

Stack<T> — создаёт стек.

2.3 Что такое List<T>?

List<T> — класс из пространства имен System.Collections.Generic, список однотипных объектов. В отличие от массива, предоставляет набор методов, облегчающих работу, таких как добавление новых элементов

2.4 Как можно обратиться к элементу списка?

Списки поддерживают индексы, с помощью которых можно обратиться к определенным элементам

2.5 Что такое Dictionary<TKey, TValue>?

Словарь (dictionary) — это обобщенная версия Hashtable содержащая в себе объект структуры KeyValuePair<TKey, TValue>. Главное свойство dictionary — быстрый поиск с помощью ключей. Можно также добавлять и удалять элементы, наподобие того как это делается в List<T>, но без расходов производительности, связанных с необходимостью смещения последующих элементов в памяти.

2.6 Как можно обратиться к элементу словаря?

Для обращения к элементам из словаря применяется их ключ, который передается в квадратных скобках

3 Вывод

3.1 Изучен процесс создания и применения параметризованных классов коллекций (обобщенных списков и словарей) на языке C#

Лабораторная работа №12
Работа с типом данных структура

1 Цель работы

1.1 Изучить процесс создания и применения структур на языке C#.

2 Контрольные вопросы

2.1 В какой области памяти хранятся типы значения?

Хранит свои значения в области памяти «стек» - все числовые значения, bool, char, enum, struct.

2.2 Что такое «структура»?

Структура (struct) в C# — это пользовательский тип данных, который используется наряду с классами и может содержать какие-либо данные и методы. Структурами также являются такие типы данных как int, double и т.д.

2.3 Чем отличается структура от класса?

Основное отличие структуры (struct) от класса (class) заключается в том, что структура — это тип значений, а класс — это ссылочный тип.

2.4 Что такое «перечисление»?

Тип данных, состоящий из набора именованных значений. Переменной, перечисляемого типа, может быть присвоен любой из элементов перечисления в качестве значения, в результате чего код станет более читабельным. Перечисления определяются с помощью ключевого слова enum.

2.5 Для чего используются перечисления?

Перечисления в C# помогают разработчикам писать более читаемый и легко масштабируемый код. Их также часто используют для ограничения количества вариантов.

2.6 Какова общая форма объявления перечисления в C#?

enum имяПеречисления {список_перечисления};

3 Вывод

3.1 Изучен процесс создания и применения структур на языке C#.

Лабораторная работа №13
Обработка и форматирование строк

1 Цель работы

1.1 Изучить процесс обработки строк на языке C#;

1.2 Научиться применять стандартные методы классов String, StringBuilder и Char для обработки строковых и символьных данных в программах на языке C#.

2 Контрольные вопросы

2.1 К какому типу переменных относятся переменные типа string?

Класс String относится к ссылочным типам. Над строками - объектами этого класса - определен широкий набор операций, соответствующий современному представлению о том, как должен быть устроен строковый тип.

2.2 Какие операции допустимы над строковыми данными?

Над строками в C# определены следующие операции:

Присваивание (=).

Две операции проверки эквивалентности (= =) и (!=).

Сцепление строк (+).

Взятие индекса ([]).

2.3 В чем отличие переменной типа string от массива символов?

Тип String похож на одномерный массив символов, но в отличие от массива символов, количество символов в строке может меняться от 0 до N.

2.4 Что такое «интерполяция строк»?

Интерполяция строк — это процесс размещения значений в шаблоне строки во время выполнения

2.5 Изменяют ли методы класса String исходную строку?

Строковые объекты неизменяемы: их нельзя изменить после их создания. Может показаться, что все методы String и операторы изменяют строку, но в действительности они возвращают результаты в новый строковый объект.

3 Вывод

3.1 Изучен процесс обработки строк на языке C#;

3.2 Научилась применять стандартные методы классов String, StringBuilder и Char для обработки строковых и символьных данных в программах на языке C#.

Лабораторная работа №14
Использование регулярных выражений

1 Цель работы

1.1 Научиться составлять шаблоны регулярных выражений в программах на C#;

1.2 Научиться применять регулярные выражения для поиска и замены подстрок в программах на языке C#.

2 Контрольные вопросы

2.1 Что такое «регулярное выражение»?

Регулярные выражения являются стандартом сопоставления с шаблоном для синтаксического анализа и изменения строк, и позволяют пользователю выразить, как компьютерная программа должна искать указанный шаблон в тексте, а затем, что она должна делать, когда найдено каждое совпадение с данным шаблоном. Иногда их сокращают как «regex».

2.2 Для чего используются регулярные выражения?

Для поиска совпадений в тексте по заданным шаблонам

2.3 Для чего используется класс Regex?

Класс Regex в C# используется для реализации регулярных выражений. Он предлагает методы и свойства для анализа большого текста с целью поиска шаблонов символов.

2.4 Каков алгоритм поиска подстроки при помощи регулярного выражения?

Вызов метода Match для поиска совпадений

2.5 Каков алгоритм замены подстроки при помощи регулярного выражения?

Вызов метода Replace для замены совпадений и передачу соответствующего делегата для определения заменяемой строки.

2.6 Для чего в регулярных выражениях применяются escape-символы?

Применяются для обозначения специальных символов, которые иначе интерпретировались бы как часть самого выражения, такие как \, который обозначает начало управляющей последовательности, или . для обозначения любого символа.

2.7 Для чего в регулярных выражениях применяются классы символов?

Специальные классы символов позволяют сократить запись в регулярных выражениях и делают их более читаемыми.

2.8 Для чего в регулярных выражениях применяются квантификаторы?

Квантификаторы в регулярных выражениях — это метасимволы, позволяющие задать количество повторений символа или группы символов в строке

3 Вывод

3.1 Научилась составлять шаблоны регулярных выражений в программах на C#;

3.2 Научилась применять регулярные выражения для поиска и замены подстрок в программах на языке C#.

Лабораторная работа №15
Разработка делегатов

1 Цель работы

1.1 Научиться разрабатывать и применять делегаты на C#

2 Контрольные вопросы

2.1 Что такое «делегат» в C#?

Делегат в C# — это тип, который представляет ссылки на методы со списком параметров и типом возвращаемого значения.

2.2 Как объявить делегат на C#?

Для объявления делегата используется ключевое слово delegate, после которого идет возвращаемый тип, название и параметры

2.3 Какие способы вызова метода через делегат существуют?

Как обычный метод; с помощью Invoke()

2.4 Какие встроенные делегаты имеются в C# и для чего они используются?

Action — представляет действие, которое ничего не возвращает, то есть имеет тип void в качестве возвращаемого типа. Как правило, передаётся в качестве параметра метода и предусматривает вызов определённых действий в ответ на произошедшие события.

Predicate — принимает один параметр и возвращает значение типа bool. Используется для сравнения и сопоставления некоторого объекта определённому условию. В качестве выходного результата возвращается значение true, если условие соблюдено, и false, если не соблюдено.

Func — возвращает результат действия и может принимать параметры. Часто используется в качестве параметра в методах.

3 Вывод

3.1 Научилась разрабатывать и применять делегаты на C#

Лабораторная работа №16
Разработка событий

1 Цель работы

1.1 Научиться создавать, вызывать и обрабатывать события на C#.

2 Контрольные вопросы

2.1 Что такое «событие» в C#?

возможность класса или объекта уведомлять другие классы или объекты о возникновении каких-либо ситуаций.

2.2 Как объявить событие на C#?

В языке программирования C# события объявляют в классе, используя для этого ключевое слово event.
После этого слова указывают тип делегата, который представляет событие.

2.3 Как создать обработчик события?

Создание события. Это событие отображается внешне для класса и используется для создания подписок на события.

Создание метода внутри класса. Метод содержит код, который запускает событие.

2.4 Какой класс является родительским для всех классов данных события?

EventArgs

2.5 Какие классы делегатов являются стандартными для создания событий в C#?

Те, в названии которых в конце стоит EventHandler

3 Вывод

3.1 Научилась создавать, вызывать и обрабатывать события на C#.

Лабораторная работа №17
Операции со списками

1 Цель работы

1.1 Изучить процесс обработки списков стандартными методами на языке C#

2 Контрольные вопросы

2.1 Что такое «LINQ»?

LINQ (Language-Integrated Query) — это простой и удобный язык запросов к источнику данных.

2.2 Что такое «анонимный тип»?

Позволяет типам данных инкапсулировать набор свойств в едином объекте без необходимости предварительного явного указания типа.

2.3 Для чего используется var?

Ключевое слово **var** в языке программирования **C#** предназначено для объявления переменных без явного указания их типа данных

2.4 Для чего используется метод Select?

Операция Select используется для создания выходной последовательности одного типа элементов из входной последовательности элементов другого типа.

2.5 Для чего используются методы OrderBy, OrderByDescending?

Для сортировки по возрастанию, убыванию

2.6 Для чего используются методы ThenBy, ThenByDescending?

ThenBy используется после OrderBy, чтобы отсортировать коллекцию по другому полю в восходящем порядке.

ThenByDescending применяется для вторичной сортировки в порядке убывания.

3 Вывод

3.1 Изучить процесс обработки списков стандартными методами на языке C#

Лабораторная работа №18
Проектирование диаграммы классов

1 Цель работы

1.1 Изучить процесс описания типов данных с применением диаграммы классов

2 Контрольные вопросы

8.1 Для чего используется диаграмма классов?

Эта диаграмма, наиболее распространенная при разработке ПО, используется для изображения логической и физической структуры системы и показывает ее классы.

8.2 Из каких элементов состоит описание типа в диаграмме классов?

Основные элементы диаграммы классов:
класс — сущность, фигурирующая в системе;
атрибуты класса — свойства объектов класса;
операция — функция над классом или его преобразование;
связи: ассоциация, агрегация, наследование.

8.3 На какие группы делятся взаимосвязи в диаграмме классов?

Существует шесть основных типов отношений между классами: наследование, реализация/реализация, композиция, агрегация, ассоциация и зависимость.

8.4 В чем отличие между взаимосвязью «наследование» и «реализация»?

Реализация предполагает определение интерфейса и его реализация в классах.

Наследование позволяет одному классу (наследнику) унаследовать функционал другого класса (родительского).

8.5 В чем отличие между взаимосвязью «агрегация» и «композиция»?

В случае композиции целое явно контролирует время жизни своей составной части (часть не существует без целого), а в случае агрегации целое хоть и содержит свою составную часть, время их жизни не связано (например, составная часть передается через параметры конструктора).

8.6 Что показывает взаимосвязь «ассоциация»?

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

8.7 Какие виды мощности отношений могут быть указаны в диаграмме классов?

Зависимость. Это связь между двумя или более классами, в которой изменение одного может вызвать изменения в другом.

Обобщение. Помогает соединить подкласс с его суперклассом. Подкласс наследуется от своего суперкласса.

Ассоциация. Представляет собой статические отношения между классами A и B.

3 Вывод

3.1 Изучен процесс описания типов данных с применением диаграммы классов

Лабораторная работа №19
Использование шаблонов проектирования

1 Цель работы

1.1 Научиться применять паттерны проектирования в разработке программ

2 Контрольные вопросы

2.1 Для чего используются порождающие паттерны?

Порождающие паттерны - это те паттерны, которые отвечают за создание тех или иных объектов. Они помогают создавать объекты так, чтобы они эффективно общались с другими, а также управлять их работой.

2.2 Какие паттерны относятся к порождающим?

К порождающим шаблонам можно отнести шесть:
Прототип (Prototype), Абстрактная фабрика (Abstract Factory), Фабричный метод (Factory Method), Строитель (Builder), Одиночка (Singleton), Ленивая инициализация (Lazy initialization).

2.3 Для чего используются структурные паттерны?

Структурные паттерны - это группа шаблонов проектирования, которая решает задачи, связанные с организацией классов и объектов в более крупные структуры.

2.4 Какие паттерны относятся к структурным?

Адаптер (Adapter), Мост (Bridge), Компоновщик (Composite), Декоратор (Decorator), Фасад (Facade), Приспособленец (Flyweight), Заместитель (Proxy).

2.5 Для чего используются поведенческие паттерны?

Поведенческие паттерны - это шаблоны проектирования, которые определяют взаимодействие и коммуникацию между объектами в программе. Управляют их поведением, а также обеспечивают более гибкую и переиспользуемую архитектуру.

2.6 Какие паттерны относятся к поведенческим?

Цепочка обязанностей (Chain of Responsibility), Команда (Command), Итератор (Iterator), Посредник (Mediator), Снимок (Memento), Наблюдатель (Observer), Состояние (State), Стратегия (Strategy), Шаблонный метод (Template method), Посетитель (Visitor)

3 Вывод

3.1 Научилась применять паттерны проектирования в разработке программ

Лабораторная работа №20
Проектирование API

1 Цель работы

1.1 Научиться проектировать API

2 Контрольные вопросы

2.1 Что такое REST?

Архитектурный стиль или набор принципов взаимодействия компонентов различных систем в интернете.

2.2 Для чего используется метод GET?

GET для получения данных

2.3 Для чего используется метод POST?

POST для публикации данных

2.4 Для чего используется метод PUT?

PUT для обновления данных

2.5 Для чего используется метод DELETE?

DELETE для удаления данных

3 Вывод

3.1 Научилась проектировать API

Лабораторная работа №21
Оптимизация кода

1 Цель работы

1.1 Изучить методы оптимизации программного кода

2 Контрольные вопросы

2.1 Что такое «оптимизация программного кода»?

Оптимизация программного кода — это модификация программ, выполняемая оптимизирующим компилятором или интерпретатором с целью улучшения их характеристик, таких как производительности или компактности, — без изменения функциональности.

2.2 Какова цель оптимизации программного кода?

Улучшение производительности, уменьшение времени отладки кода

2.3 Какие методы оптимизации программного кода применяются?

Оптимизация циклов. Существует большое количество оптимизаций, применяемых к циклам.

Межпроцедурная оптимизация. Анализирует сразу весь исходный код программы.

Внутрипроцедурная оптимизация. Выполняется целиком в рамках функции или процедуры.

3 Вывод

3.1 Изучены методы оптимизации программного кода

Лабораторная работа №22
Асинхронное программирование

1 Цель работы

1.1 Научиться реализовывать и запускать асинхронные операции на C#.

1.2 Научиться выполнять вычисления, используя асинхронные операции.

1.3 Научиться выполнять ввод и вывод данных, используя асинхронные операции

2 Контрольные вопросы

2.1 Какие ключевые слова используются в C# для работы с асинхронными вызовами?

Два оператора: async и await

2.2 Какие типы возврата могут быть у асинхронных методов и для чего предназначен каждый из типов?

При использовании ключевого слова void асинхронный метод ничего не возвращает

Task — возвращение объекта типа Task

Task<T> — метод может возвращать некоторое значение. Тогда возвращаемое значение оборачивается в объект Task, а возвращаемым типом является Task<T>

2.3 Как вызвать метод в асинхронном режиме?

Чтобы вызвать метод в асинхронном режиме нужно:

Использовать ключевое слово «await» перед вызовом метода.

Обернуть код внутри блока «try-catch» для обработки возможных исключений.

2.4 Как указать, что в методе могут быть асинхронные вызовы?

Ключевое слово async

2.5 Как обработать исключения, возникшие в асинхронных вызовах?

Для обработки ошибок в асинхронных методах, использующих ключевые слова async и await, выражение await помещается в блок try.

3 Вывод

3.1 Научилась реализовывать и запускать асинхронные операции на C#.

3.2 Научилась выполнять вычисления, используя асинхронные операции.

3.3 Научилась выполнять ввод и вывод данных, используя асинхронные операции

Лабораторная работа №23
Документирование кода

1 Цель работы

1.1 Изучить процесс документирования программного кода

2 Контрольные вопросы

2.1 Что такое «XML-документация»?

XML документация — это специальные теги XML, которые содержатся в комментариях и описывают свойства или методы в конкретном файле.

2.2 Как сгенерировать XML-комментарий?

Написать в коде тройной слэш

2.3 Какие действия нужно выполнить, чтобы XML-документация была видна при подключении библиотеки в стороннем решении?

Поместить файл документации в ту же папку, что и dll-файл

2.4 Что пишется в разделе summary?

Предоставление краткого обзора элемента, который он документирует.

2.5 Что пишется в разделе param?

Используется для описания параметров метода, конструктора или другого члена.

2.6 Что пишется в разделе returns?

Описание возвращаемого значения

3 Вывод

3.1 Изучен процесс документирования программного кода

Лабораторная работа №24
Рефакторинг кода

1 Цель работы

1.1 Изучить техники рефакторинга программного кода.

2 Контрольные вопросы

2.1 Что такое «рефакторинг»?

Рефакторинг может привести к постепенному эволюционному улучшению структуры программы, сделать ее более гибкой и адаптируемой к новым условиям. Оптимизация производительности. В некоторых случаях рефакторинг направлен на устранение узких мест в коде, что может положительно отразиться на скорости работы программы.

2.2 Какие группы техник рефакторинга существуют?

Выделение методов в отдельные методы. Это позволяет разбить сложный метод на разные части, сохранив при этом его функциональность.

Удаление неиспользуемых директив. Это поможет сделать код более чистым.

Выделение интерфейсов. Это поможет создать слабосвязанный код.

Переименование переменных. Это позволит сделать код более читаемым.

2.3 Как выполнить рефакторинг в Visual Studio?

С помощью вкладки Быстрые действия и рефакторинг

3 Вывод

3.1 Изучены техники рефакторинга программного кода.

Лабораторная работа №25
Работа с системой контроля версий

1 Цель работы

1.1 Научиться применять систему контроля версий git в процессе разработки программного обеспечения.

2 Контрольные вопросы

2.1 Что такое «репозиторий»?

Репозиторий Git — это все файлы, находящиеся под контролем версий, вместе с историей их изменения и другой служебной информацией.

2.2 Что указывается в файле readme.md?

что делает проект;
почему проект полезен;
как пользователи могут приступить к работе с проектом;
где пользователи могут получить помощь по проекту;
кто поддерживает проект и вносит вклад в проект.

2.3 Что указывается в файле .gitignore?

Git сопоставляет .gitignore шаблоны поиска с файлами в проекте, чтобы определить, какие файлы следует игнорировать.

2.4 Какое программное обеспечение может применяться для управления git-репозиторием?

2.5 Где может располагаться репозиторий?

Репозитории могут располагаться:

Локально — на машине пользователя, в рабочей папке проекта.

Удаленно — в облаке, на сторонних сервисах, специально созданных для работы с проектами git.

3 Вывод

3.1 Научилась применять систему контроля версий git в процессе разработки программного обеспечения.

Лабораторная работа №26
Разработка интерфейса пользователя: компоновка элементов

1 Цель работы

1.1 Изучить элементы-контейнеры, применяющиеся в приложениях WPF для компоновки.

2 Контрольные вопросы

2.1 Что такое «компоновка» в WPF?

Это процесс размещения визуальных элементов на поверхности родительского элемента.

2.2 Какой класс является родительским для всех элементов-контейнеров в WPF?

Класс Panel

2.3 Как выровнять элементы внутри контейнера по высоте и по ширине?

Ширину с помощью свойства Width и высоту с помощью свойства Height.

2.4 В чем особенность компоновки с использованием следующих элементов-контейнеров: Grid и Canvas?

Grid – один из наиболее гибких и широко используемых контейнеров компоновки. Grid организует пространство как таблицу с настраиваемыми столбцами и строками.

Canvas в WPF позволяет размещать элементы с использованием точных координат. Это делает его ценным инструментом при построении других элементов, например, поверхностей рисования для инструментов построения диаграмм.

2.5 Чем отличается компоновка с использованием StackPanel, DockPanel, WrapPanel?

StackPanel — размещает элементы в горизонтальные и вертикальные стопки. Этот контейнер часто используется для организации небольших участков более крупного и сложного окна.

WrapPanel — размещает элементы управления в доступном пространстве, по одной строке или колонке.

DockPanel — размещает элементы управления относительно одного из своих внешних краёв.

2.6 В каких единицах измерения могут задаваться размеры элементов в приложениях WPF?

сантиметры (cm), точки (pt), дюймы (in) и пиксели (px)

3 Вывод

3.1 Изучены элементы-контейнеры, применяющиеся в приложениях WPF для компоновки.

Лабораторная работа №27
Организация интерфейса пользователя

1 Цель работы

1.1 Изучить процесс настройки интерфейса и организации переходов в приложениях WPF.

2 Контрольные вопросы

2.1 Для чего используется элемент управления Frame?

Frame является элементом управления содержимым, который предоставляет возможность перехода к содержимому и его отображения

2.2 Для чего используется элемент управления Page?

Класс Page — это элемент управления, который выступает в качестве пользовательского интерфейса веб-приложения

2.3 Как перейти к определенной странице, используя фрейм?

Через Navigate(), в параметрах указать нужную страницу

2.4 Как проверить, что во фрейме можно вернуться к предыдущей странице?

С помощью свойства CanGoBack

2.5 Как перейти к предыдущей странице, используя фрейм?

GoBack()

2.6 Какие элементы позволяют сгруппировать содержимое?

Expander, TabItem

3 Вывод

3.1 Изучен процесс настройки интерфейса и организации переходов в приложениях WPF.

Лабораторная работа №28
Разработка интерфейса пользователя: настройка стилей

1 Цель работы

1.1 Изучить процесс настройки интерфейса с использованием стилей в приложениях WPF.

2 Контрольные вопросы

2.1 Для чего используются стили в приложениях WPF?

Стили WPF позволяют определять общий набор характеристик форматирования и применять его повсюду в приложении для обеспечения согласованного вида

2.2 Какова общая форма локального определения стиля элемента управления?

<Элемент>
<Элемент.Style>
<StyleTargetType = «Элемент»>
<Setter Property= «Свойство» Value= «Значение» >

Закрывающиеся тэги

2.3 Какова общая форма определения стиля приложения?

<StyleTargetType = «Control»>
<Setter Property= «Свойство» Value= «Значение» >

2.4 Как указать явное использование стилей?

В разметке в Style указать название используемого стиля

2.5 Как указать наследование стиля?

Через BaseOn

2.6 Как добавить новую тему в приложение?

Создать словарь ресурсов и настроить его

2.7 Как выполнить переключение между темами?

Добавить несколько тем и по кнопкам привязать их изменение

3 Вывод

3.1 Изучен процесс настройки интерфейса с использованием стилей в приложениях WPF.

Лабораторная работа №29
Разработка интерфейса пользователя: применение триггеров

1 Цель работы

1.1 Изучить процесс применения триггеров в приложениях WPF.

1.2 Закрепить навык применения стилей в приложениях на WPF

2 Контрольные вопросы

2.1 Что позволяют делать триггеры в приложениях WPF?

Тригерры позволяют декларативно задать некоторые действия, которые выполняются при изменении свойств стиля.

2.2 Какие виды триггеров можно разработать в приложениях WPF?

Trigger — простейшая форма триггера. Он следит за изменением в свойстве зависимости и затем использует средство установки для изменения стиля.

MultiTrigger — похож на Trigger, но поддерживает проверку множества условий. Этот триггер вступает в действие, только если удовлетворены все заданные условия.

DataTrigger — работает с привязкой данных. Он похож на Trigger, но следит за изменением в любых связанных данных.

MultiDataTrigger — объединяет множество триггеров данных.

EventTrigger — наиболее сложный триггер. Он применяет анимацию, когда возникает соответствующее событие.

2.3 Для чего используется и когда срабатывает триггер свойств?

Триггер свойства — это элемент в разметке, который ждет, пока выбранное свойство элемента управления не изменится согласно установленным правилам. После совпадения условий он модифицирует необходимые свойства.

2.4 Для чего используется и когда срабатывает триггер данных?

Вызываются в ответ на изменения значений любых свойств (они необязательно должны быть свойствами зависимостей)

2.5 Для чего используется и когда срабатывает триггер событий?

Триггер события в WPF используется для применения анимации. Он ожидает возникновения конкретного события и требует предоставления последовательности действий, модифицирующих элемент управления.

3 Вывод

3.1 Изучен процесс применения триггеров в приложениях WPF.

3.2 Закреплен навык применения стилей в приложениях на WPF

Лабораторная работа №30
Изучение особенностей элементов выбора в приложениях WPF

1 Цель работы

1.1 Изучить свойства и процесс обработки событий элементов выбора в приложениях WPF

2 Контрольные вопросы

2.1 Что такое CheckBox и для чего он используется?

CheckBox — это элемент управления, который представляет собой обычный флажок. Он является производным от класса ToggleButton и может принимать три состояния: Checked, Unchecked и Intermediate.

2.2 Что такое RadioButton и для чего он используется?

Radiobutton - переключатель - это элемент управления, который позволяет пользователю выбрать одну опцию из группы опций.

2.3 Как проверить, что флажок или переключатель выбран?

Свойство IsChecked

2.4 Какое событие срабатывает при выборе флажка или переключателя?

Checked

2.5 Какое событие срабатывает при снятии выбора флажка или переключателя?

Unchecked

2.6 Какие значения могут принимать флажки и переключатели?

Переключатели позволяют сделать выбор между двумя прямо противоположными вариантами. Как правило, переключатели используются для включения и отключения какого-либо действия (запустить или остановить что-то).

3 Вывод

3.1 Изучены свойства и процесс обработки событий элементов выбора в приложениях WPF

Лабораторная работа №31
Разработка приложения с использованием текстовых компонентов

1 Цель работы

1.1 Изучить различные типы полей ввода, применяющихся в приложениях WPF;

1.2 Изучить свойства полей ввода и процесс обработки событий полей ввода.

2 Контрольные вопросы

2.1 Как задать имя элементам управления в WPF?

В свойстве x:Name

2.2 Как создать обработчик события в WPF?

В разметке выбрать нужное событие и добавить его или во вкладке события по двойному щелчку добавить событие

2.3 Для чего используется Slider в WPF?

Класс Slider (ползунок) является специализированным элементом управления, который часто бывает очень полезным. Им можно воспользоваться для задания числовых значений в тех ситуациях, когда само число не особенно важно.

2.4 Для чего используется TextBox в WPF?

Чаще всего TextBox используется для редактирования текста без форматирования в форме. Например, форма, запрашивающая имя пользователя, номер телефона и т. д., будет использовать элементы управления TextBox для ввода текста

2.5 Для чего используется TextBlock в WPF?

Он позволяет разместить текст на экране, как Label, но при этом прост в использовании и не требователен к ресурсам. В общем восприятии, Label это короткий однострочный текст (но может включать в себя, например, изображение), в то время как TextBlock отлично справляется еще и с многострочным текстом, но содержать в себе может исключительно текст

2.6 Для чего используется Calendar в WPF?

Элемент Calendar выводит календарь, похожий на тот, который использует операционная система Windows (например, при настройке системной даты).

2.7 Для чего используется DatePicker в WPF?

Это элемент управления, который позволяет пользователю выбирать значение даты. Пользователь выбирает дату с помощью выпадающего списка для значений месяца, дня и года.

2.8 Для чего используется PasswordBox в WPF?

Элемент предназначен для ввода парольной информации. По сути это тоже текстовое поле, только для ввода символов используется маска

3 Вывод

3.1 Изучены различные типы полей ввода, применяющихся в приложениях WPF;

3.2 Изучены свойства полей ввода и процесс обработки событий полей ввода.

Лабораторная работа №32
Разработка приложения с использованием элементов отображения списков

1 Цель работы

1.1 Изучить свойства и процесс обработки событий элементов отображения списков в приложениях WPF

2 Контрольные вопросы

2.1 Что такое ComboBox и для чего он используется?

Он представляет собой комбинацию расширяемого списка опций, из которых пользователь может сделать выбор, и текстового поля, где можно ввести текст.

2.2 Что такое ListBox и для чего он используется?

Класс ListBox представляет распространенный компонент среды Windows — списки переменной длины, которые позволяют пользователю выбрать один из элементов.

2.3 Какое событие срабатывает при выборе элемента в селекторе?

SelectionChanged

2.4 В каком свойстве хранятся элементы селекторов?

В Items

2.5 Какого типа элементы могут быть в селекторе?

Коллеция с разными типами элементов

2.6 Какое свойство позволяет привязать селектор к набору данных?

ItemsSource

2.7 Для чего используется свойство DisplayMemberPath в селекторе?

Идентифицирует свойство, которое будет применяться для создания отображаемого текста каждого элемента коллекции. ItemTemplate.

3 Вывод

3.1 Изучены свойства и процесс обработки событий элементов отображения списков в приложениях WPF

Лабораторная работа №33
Разработка приложения для отображения данных в табличном виде

1 Цель работы

1.1 Изучить свойства и процесс настройки внешнего вида элемента DataGrid в приложениях WPF.

2 Контрольные вопросы

2.1 Что такое DataGrid и для чего он используется?

DataGrid представляет собой элемент управления отображением данных, который извлекает информацию из коллекции объектов и визуализирует ее в сетке со строками и ячейками.

2.2 Какие типы столбцов поддерживаются в DataGrid?

DataGridTextColumn — стандартный выбор для большинства типов данных. Значение преобразуется в текст и отображается в элементе управления TextBox.

DataGridCheckBoxColumn — отображает флажок и чаще всего автоматически используется для булевых значений.

DataGridHyperlinkColumn — отображает ссылку, по которой можно щёлкать.

DataGridComboBox — первоначально выглядит как DataGridTextColumn, но в режиме редактирования превращается в раскрывающийся список ComboBox.

DataGridTemplateColumn — позволяет определять шаблон данных для отображения значений столбцов.

2.3 Как добавить кнопку в строки DataGrid?

Настроить DataGridTemplateColumn и добавить туда кнопку

2.4 Как указать источник данных для DataGrid?

С помощью свойства ItemsSource

2.5 Как указать источник данных для выпадающего списка DataGrid?

SelectedItemBinding, SelectedValueBinding

3 Вывод

3.1 Изучены свойства и процесс настройки внешнего вида элемента DataGrid в приложениях WPF.

Лабораторная работа №34
Разработка приложения с меню и панелью инструментов

1 Цель работы

1.1 Изучить свойства и процесс настройки внешнего вида меню и панели инструментов в приложениях WPF.

2 Контрольные вопросы

2.1 Что такое Menu и для чего он используется?

Элемент Menu в WPF используется для создания стандартных меню.

Он включает набор элементов MenuItem, которые являются элементами управления содержимым и могут включать другие элементы MenuItem и не только.

2.2 Что такое ContextMenu и для чего он используется?

ContextMenu - это всплывающее меню, которое позволяет элементу управления предоставлять функциональность, специфичную для контекста элемента управления.

2.3 Что такое ToolBar и для чего он используется?

Элемент управления WPF ToolBar используется для создания интерфейсов с панелями инструментов.

2.4 Что такое StatusBar и для чего он используется?

StatusBar — это элемент управления в WPF, который используется для отображения текста и графических индикаторов.

3 Вывод

3.1 Изучены свойства и процесс настройки внешнего вида меню и панели инструментов в приложениях WPF.

Лабораторная работа №35
Создание проекта с использованием компонентов стандартных диалогов

1 Цель работы

1.1 Изучить процесс создания и применения стандартных диалоговых окон в приложениях WPF.

2 Контрольные вопросы

2.1 Что такое диалоговое окно?

Диалоговое окно — это временное окно, создаваемое приложением для получения введенных пользователем данных.

2.2 Для чего используется OpenFileDialog?

OpenFileDialog — это класс, который реализует общее диалоговое окно открытия файла

2.3 Для чего используется SaveFileDialog?

SaveFileDialog — это диалог в WPF, который помогает выбрать местоположение и имя файла для сохранения

2.4 Что такое MessageBox и какие настройки можно ему задать?

С помощью функции MessageBox можно вывести на экран окно сообщения с заданными заголовком и текстом. Окно можно дополнить иконкой из специально предназначенного для этого стандартного набора

2.5 Для чего используется PrintDialog?

Он только позволяет пользователю задать атрибуты печати

3 Вывод

3.1 Изучить процесс создания и применения стандартных диалоговых окон в приложениях WPF.

Лабораторная работа №36
Разработка приложения с несколькими формами

1 Цель работы

1.1 Изучить процесс создания и применения пользовательских окон в приложениях WPF.

2 Контрольные вопросы

2.1 Как сделать доступными данные пользовательского диалогового окна вызывающим его окнам?

Создать открытое свойство на чтение данных

2.2 Какие значения может принимать переменная DialogResult?

значения true, false и null

2.3 Как открыть окно в диалоговом режиме?

Для открытия окна в диалоговом режиме в WPF можно использовать метод Show().

2.4 Как открыть окно в недиалоговом режиме?

Метод ShowDialog()

2.5 В чем отличие между диалоговым и недиалоговым режимами работы?

Диалоговое окно должны быть закрыты или скрыты, прежде чем вы сможете продолжить работу с остальной частью приложения

3 Вывод

3.1 Изучен процесс создания и применения пользовательских окон в приложениях WPF.

Лабораторная работа №37
Реализация фильтрации данных

1 Цель работы

1.1 Научиться применять LINQ-запросы для фильтрации данных по одному критерию и набору критериев.

2 Контрольные вопросы

2.1 Для чего используется метод Where?

Предложение where используется в выражении запроса для того, чтобы указать, какие элементы из источника данных будут возвращаться в выражении запроса.

2.2 Какие логические операторы могут применяться при составлении условий?

Включают операторы AND (&&), OR (||) и NOT (!)

2.3 Как выполнить регистронезависимый поиск?

Для выполнения регистронезависимого поиска можно использовать метод String.Equals с параметром StringComparison.OrdinalIgnoreCase, который игнорирует регистр символов.

2.4 Как проверить, что строка начинается с определенного текста?

С помощью метода StartsWith();

2.5 Как проверить, что строка содержит определенный текст?

С помощью метода Contains()

2.6 Как составить LINQ-запрос для фильтрации по нескольким критериям, указываемым пользователем?

Можно использовать условные операторы для добавления дополнительных фильтров в запрос.

3 Вывод

3.1 Научилась применять LINQ-запросы для фильтрации данных по одному критерию и набору критериев.

Лабораторная работа №38
Реализация постраничного вывода информации

1 Цель работы

1.1 Научиться применять LINQ-запросы для постраничного вывода данных.

2 Контрольные вопросы

2.1 Почему может потребоваться выводить данные постранично?

Ускорить загрузку страниц;

Упростить навигацию по сайту;

Сделать просмотр информации удобным для пользователей;

Привести дизайн страниц к привычному для пользователей виду.

2.2 Что такое «пагинация»?

Это нумерация страниц на сайте. Обычно она выглядит как перечень номеров или букв. Каждый номер — это ссылка, которая ведет на конкретную страницу.

2.3 Для чего используется метод Take?

Метод Take используется для возврата указанного количества элементов из входной последовательности, начиная с её начала.

2.4 Для чего используется метод Skip?

Метод Skip используется для пропуска указанного количества элементов из входной последовательности, начиная с её начала, и вывода остальных элементов.

2.5 Для чего используется метод TakeWhile?

Метод TakeWhile возвращает набор элементов последовательности, до тех пор, пока заданное условие истинно

2.6 Для чего используется метод SkipWhile?

SkipWhile — пропускает элементы в последовательности, пока они удовлетворяют заданному условию, и затем возвращает оставшиеся элементы.

2.7 Каким должен быть набор данных, чтобы можно было использовать Take и Skip?

Методы Take и Skip можно использовать, когда необходимо получать элементы из большого набора данных «постранично», например, последовательно по десять элементов.

3 Вывод

3.1 Научилась применять LINQ-запросы для постраничного вывода данных.

Лабораторная работа №39
Реализация группировки и соединения данных

1 Цель работы

1.1 Научиться применять LINQ-запросы для группировки и соединения данных.

2 Контрольные вопросы

2.1 Какие агрегатные функции поддерживаются в LINQ?

В LINQ поддерживаются следующие агрегатные функции:

Sum() — возвращает сумму всех чисел, предоставленных в качестве входных данных.

Max() — возвращает максимальное значение всех чисел, предоставленных в качестве входных данных.

Min() — возвращает минимальное значение всех чисел, предоставленных в качестве входных данных.

Count() — возвращает общее количество всех чисел, предоставленных в качестве входных данных.

Average() — возвращает среднее значение чисел, предоставленных в качестве входных данных.

Aggregate() — возвращает накопленную агрегацию всех чисел, предоставленных в качестве входных данных.

2.2 Что возвращает метод Distinct?

Неупорядоченную последовательность

2.3 Для чего используется метод GroupBy?

Используется для группировки элементов в коллекции на основе какого-либо свойства, которое является общим для всех элементов.

2.4 Для чего используется метод Join?

Соединение в LINQ используется для объединения двух разнотипных наборов в один.

2.5 Для чего используется метод GroupJoin?

Используется для выполнения операции группового объединения между двумя последовательностями (коллекциями) на основе совпадающих ключей, извлечённых из элементов каждой последовательности.

2.6 В чем отличие результатов, полученных при вызове LINQ-методов Concat, Union, Except, Intersect?

Union — объединяет две последовательности и возвращает новую последовательность, которая содержит уникальные элементы из обеих.

Intersect — возвращает новую последовательность, которая включает элементы, существующие в обеих исходных последовательностях.

Except — возвращает элементы из первой последовательности, которые не существуют во второй.

Concat — используется для добавления одной последовательности к другой.

3 Вывод

3.1 Научилась применять LINQ-запросы для группировки и соединения данных.

Лабораторная работа №40
Разработка приложения для работы с графикой

1 Цель работы

1.1 Изучить процесс рисования и трансформации объектов в приложениях WPF.

2 Контрольные вопросы

2.1 Какие графические примитивы доступны в приложениях WPF?

Ellipse, Rectangle, Line, Polygon, Polyline

2.2 Для чего используется Path?

Класс Path в WPF используется для создания сложных и комплексных рисунков и дизайна.

2.3 Какие виды трансформаций объектов доступны в приложениях WPF?

В приложениях WPF доступны следующие виды трансформаций объектов:

TranslateTransform — сдвигает элементы по горизонтали и вертикали.

RotateTransform — вращает элемент.

ScaleTransform — выполняет операции масштабирования.

SkewTransform — изменяет позицию элемента путём наклона на определённое количество градусов.

MatrixTransform — изменяет координатную систему в соответствии с определённой матрицей.

TransformGroup — представляет группу трансформаций.

2.4 Как указать заливку и контур графических объектов в WPF?

Заливка — свойство Fill, StrokeThickness — описывает толщину линий контура фигуры.

3 Вывод

3.1 Изучен процесс рисования и трансформации объектов в приложениях WPF.

Лабораторная работа №41
Разработка приложения с анимацией

1 Цель работы

1.1 Изучить процесс анимации объектов в приложениях WPF

2 Контрольные вопросы

2.1 Как задать анимацию размеров в приложениях WPF?

Создать триггер событий, в тэге <BeginStoryboard> задать изменение размеров элемента управления.

2.2 Как задать анимацию цвета в приложениях WPF?

Можно использовать объекты Storyboard и ColorAnimation, которые позволяют анимировать изменение цвета элемента интерфейса.

2.3 Чем отличается покадровая анимация от плавной анимации в WPF?

Покадровая анимация в WPF изменяет свойства объекта постепенно, а плавная анимация обеспечивает более плавное и непрерывное изменение свойств

2.4 Какие свойства позволяют управлять анимацией в WPF и для чего предназначено каждое свойство?

Свойства: Duration (продолжительность анимации), RepeatBehavior (повторение анимации), AutoReverse (автоматическое возвращение к начальным значениям), и FillBehavior (поведение после завершения анимации).

3 Вывод

3.1 Изучен процесс анимации объектов в приложениях WPF

Лабораторная работа №42
Разработка мультимедиа-приложения

1 Цель работы

1.1 Изучить процесс создания мультимедиа-приложений

2 Контрольные вопросы

2.1 Что такое «мультимедиа»?

Это данные или содержание, которые представляются одновременно в разных формах: звук, анимированная компьютерная графика, видеоряд, изображения.

2.2 Какие элементы позволяют отображать изображения в приложениях WPF?
Image

2.3 Какие элементы позволяют воспроизводить аудио в приложениях WPF?
MediaElement и MediaPlayer.

2.4 Какие элементы позволяют воспроизводить видео в приложениях WPF?
MediaElement и MediaPlayer.

3 Вывод

3.1 Изучен процесс создания мультимедиа-приложений

Лабораторная работа №43
Разработка игрового приложения

1 Цель работы

1.1 Изучить процесс разработки игровых приложений WPF, использующих графику, обработчики событий и таймеры.

2 Контрольные вопросы

2.1 Как подключить таймер к приложению на WPF?

Подключить пространство System.Windows.Threading
DispatcherTimer timer = new();

2.2 Как создать обработчик события для таймера в приложении WPF?

timer.Tick += Timer_Tick

2.3 Как изменить интервал таймера в приложении WPF?

timer.Interval = TimerSpan. ...;
Выбрать метод с нужной единицей измерения и указать значение в параметре

3 Вывод

3.1 Изучен процесс разработки игровых приложений WPF, использующих графику, обработчики событий и таймеры.

Лабораторная работа №44
Создание БД

1 Цель работы

1.1 Изучить процесс создания таблиц и связей между ними в реляционной СУБД.

2 Контрольные вопросы

2.1 Что такое «система управления базами данных»?

Система управления базами данных (сокращенно СУБД) – это программное обеспечение для создания и работы с базами данных. Главная функция СУБД – это управление данными

2.2 Что такое «база данных»?

База данных — это упорядоченный набор структурированной информации или данных, которые обычно хранятся в электронном виде в компьютерной системе. База данных обычно управляется системой управления базами данных (СУБД).

2.3 Какие СУБД являются реляционными (привести примеры ПО)?

Наиболее известными реляционными базами данных являются Open Source проекты PostgreSQL, MySQL и SQLite, а также проприетарные решения Oracle, Microsoft SQL Server

3 Вывод

3.1 Изучен процесс создания таблиц и связей между ними в реляционной СУБД.

Лабораторная работа №45
Создание приложения с БД для чтения данных

1 Цель работы

1.1 Научиться создавать приложения для чтения данных из БД

2 Контрольные вопросы

2.1 Каково назначение элемента SqlConnection?

Объект SqlConnection представляет уникальный доступ в источнику данных SQL Server. С клиент-серверной системой базы данных, SqlConnection эквивалентен сетевому подключению к серверу. SqlConnection как правило используется с SqlDataAdapter и SqlCommand что б улучшить производительность, подключаясь к базе данных Microsoft SQL Server.

2.2 Как считать из БД одно значение?

2.3 Каково назначение элемента SqlDataReader?

Класс SqlDataReader пространства имён Microsoft.Data.SqlClient предназначен для получения данных из базы. Как следует из названия, объекты данного класса только читают информацию, но не изменяют данные в базе.

2.4 Как получить данные из БД, используя SqlDataAdapter?

Для получения данных из БД с помощью SqlDataAdapter необходимо организовать подключение к БД и выполнить команду SELECT.

2.5 Какие пространства имен требуется подключить для реализации подключения к СУБД MS SQL Server?

Microsoft.Data.SqlClient

3 Вывод

3.1 Научилась создавать приложения для чтения данных из БД

Лабораторная работа №46
Создание приложения с БД для записи данных

1 Цель работы

1.1 Научиться создавать приложения для записи данных в БД

2 Контрольные вопросы

2.1 Для чего применяется компонент SqlDataAdapter?

SqlDataAdapter предоставляет богатый интерфейс для автоматического заполнения DataTable и DataSet, а также предоставляет средства синхронизации данных между DataSet и БД

2.2 Для чего применяется компонент SqlCommandBuilder?

SqlCommandBuilder — это компонент, который используется для построения и выполнения SQL-запросов на основе предоставленной команды выбора.

2.3 Как изменить данные в БД, используя SqlCommand?

Создать объект типа SqlCommand, создать запрос с изменением данных и вызвать запрос

2.4 Как изменить данные в БД, используя SqlDataAdapter?

Создаете экземпляр класса SqlCommandBuilder, который автоматически создает и настраивает команды Insert, Update и Delete для SqlDataAdapter на основе структуры таблицы. Затем вызываете методы Fill и Update SqlDataAdapter для выполнения операций с данными.

2.5 Как связать SqlCommandBuilder и SqlDataAdapter?

Для связывания SqlCommandBuilder и SqlDataAdapter достаточно передать SqlDataAdapter в конструктор SqlCommandBuilder.

3 Вывод

3.1 Научилась создавать приложения для записи данных в БД

Лабораторная работа №47
Создание запросов к БД

1 Цель работы

1.1 Научиться выполнять запросы к БД из клиентского приложения,

1.2 Научиться передавать параметры в запросы

2 Контрольные вопросы

2.1 Для чего используются параметры в командах?

Параметры избавляют от некорректного ввода данных, необходимости форматировать значения и реализовывать защиту от sql-инъекций

2.2 Как добавить параметр в команду, используя Add?

В параметрах метода Add передать название параметра и какого типа он должен быть

С помощью свойства Value — изменять значение параметра

2.3 Как добавить параметр в команду, используя AddWithValue?

В параметрах метода передать название параметра и его значение

2.4 Какие свойства можно указать у параметра?

Значение, входной/выходной

3 Вывод

3.1 Научилась выполнять запросы к БД из клиентского приложения,

3.2 Научилась передавать параметры в запросы

Лабораторная работа №48
Создание хранимых процедур

1 Цель работы

1.1 Научиться создавать и использовать хранимые процедуры в MS SQL Server;

1.2 Закрепить навык создания запросов на выборку и модификацию данных в MS SQL Server.

2 Контрольные вопросы

2.1 Что такое хранимые процедуры и для чего они применяются?

Хранимая процедура — это предварительно скомпилированный набор операторов SQL, который хранится в базе данных и может быть выполнен одной командой

2.2 Как вызвать выполнение хранимой процедуры в клиентском приложении?

Создать команду sql-запроса, вызвать её нужным методом, ExecuteNonQuery(), ExecuteScalar()

2.3 Как задать входные параметры хранимой процедуры?

С помощью параметров

3 Вывод

3.1 Научилась создавать и использовать хранимые процедуры в MS SQL Server;

3.2 Закреплен навык создания запросов на выборку и модификацию данных в MS SQL Server.

Практическая работа №1
Разработка приложений для обработки файлов

1 Цель работы:

1.1 Научиться применять классы для работы с файлами в приложениях на C#.

2 Контрольные вопросы

2.1 В чем отличие между классами Directory и DirectoryInfo?

Directory это статический класс который предоставляет статические методы для работы с директориями. DirectoryInfo - это класс, который предоставляет информацию про определённую директорию.

2.2 В чем отличие между классами File и FileInfo?

Классы File и FileInfo предназначены для обработки файлов как элементов файловой системы, без доступа к их содержимому. Класс File содержит только классовые методы; при этом первым параметром любого метода является имя name обрабатываемого файла. Класс FileInfo содержит свойства и экземплярные методы, для доступа к которым необходимо создать объект данного класса, указав в его конструкторе имя обрабатываемого файла.

2.3 Как получить список файлов и папок определенного каталога?

GetFiles(): получает список файлов в папке в виде массива FileInfo.

2.4 Какие свойства класса FileInfo позволяют получить информацию о файле?

Вот некоторые свойства класса FileInfo, которые позволяют получить информацию о файле:

Directory — возвращает экземпляр родительского каталога файла.

DirectoryName — возвращает полный путь родительского каталога файла.

Exists — возвращает значение, указывающее, существует ли файл.

IsReadOnly — определяет, может ли текущий файл быть изменён.

Length — возвращает размер текущего файла в байтах.

Name — возвращает имя файла.

Extension — возвращает часть расширения файла.

CreationTime — позволяет получить или установить время создания текущего файла или каталога.

LastAccessTime — позволяет получить или установить время последнего доступа к текущему файлу или каталогу.

LastWriteTime — позволяет получить или установить время последней записи в текущий файл или каталог.

3 Выводы

3.1 Научилась применять классы для работы с файлами в приложениях на C#.

Практическая работа №2
Работа с табличными файлами

1 Цель работы:

1.1 Научиться выполнять создание и редактирование табличных документов на C#

2 Контрольные вопросы

2.1 Какое пространство имен требуется подключить для работы с Excel?
Microsoft.Office.Interop.Excel

2.2 Как создать объект типа «приложение Excel» в программе на C#?
Включить пространство имен и задать псевдонимы им
var excelApp = new Excel.Application();
excelApp.Visible = true;

2.3 Что такое Workbooks?
Книга

2.4 Что такое Worksheets?
Листы

2.5 Что такое Range?
Диапазон

2.6 Что такое Cells?
Ячейка

2.7 Как получить доступ к значению ячейки и диапазона?
К ячейке — через указание в [] [] столбца и строки
К диапазону — задать диапазону ячейку начала и конца, задавать значение через свойство Value

3 Выводы

3.1 Научилась выполнять создание и редактирование табличных документов на C#

Практическая работа №3
Работа с текстовыми файлами

1 Цель работы:

1.1 Научиться выполнять создание и редактирование текстовых документов на C#

2 Контрольные вопросы

2.1 Какое пространство имен требуется подключить для работы с Word?
Microsoft.Office.Interop.Word

2.2 Как создать объект типа «приложение Word» в программе на C#?
Включить пространство имен и задать псевдонимы им
var wordApp = new Word.Application();
wordApp.Visible = true;

2.3 Что такое Documents?
Коллекция всех Document объектов, открытых в Word.

2.4 Что такое Range?
Интервал между двумя позициями

2.5 Что такое Selection?
Выделенная область

2.6 Что такое Paragraphes?
Paragraphs – параграфы

2.7 Что такое Tables?
Таблицы

3 Выводы

3.1 Научилась выполнять создание и редактирование текстовых документов на C#

Практическая работа №4
Сохранение настроек приложения

1 Цель работы:

1.1 Научиться сохранять настройки в клиентском приложении на C#

2 Контрольные вопросы

2.1 Как добавить настройки в приложение на C#?
Программно или вручную в параметрах — создать или открыть

2.2 Как программно считать значение параметра из настроек?
Properties.Settings.Default.ИмяПараметра

2.3 Как программно изменить значение параметра в настройках?
Обратиться к значению параметра и изменить его

2.4 Как выполнить сохранение значений параметров в настройках?
Properties.Settings.Default.Save()

3 Выводы

3.1 Научилась сохранять настройки в клиентском приложении на C#

Практическая работа №5
Создание пользовательских элементов управления

1 Цель работы:

1.1 Научиться создавать пользовательские элементы управления в приложении WPF.

2 Контрольные вопросы

2.1 Для чего применяется ControlTemplate?

ControlTemplate используется в качестве значения свойства Control.Template, которое определяет визуальные элементы элемента управления путем применения шаблона.

2.2 Где может быть описан ControlTemplate?

Чаще всего ControlTemplate объявляется как ресурс в разделе Resources файла XAML.

2.3 Для чего применяется UserControl?

UserControl применяется для создания элементов управления, которые можно использовать в нескольких местах в приложении или организации.

2.4 Каков алгоритм создания пользовательского элемента управления?

ПКМ по проекту — Добавить — Пользовательский ЭУ

Разместить ЭУ и настроить внешний вид

Пересобрать проект

2.5 Как программно создать обработчик события?

Создать перехватчик public event RoutedEventHandler ИмяПерехватчика

2.6 Для чего используется свойство зависимости?

Все свойства зависимостей представляют статические публичные поля (public static) с суффиксом Property.

3 Выводы

3.1 Научилась создавать пользовательские элементы управления в приложении WPF.

Практическая работа №6
Привязка данных

1 Цель работы:

1.1 Научиться выполнять привязку данных в приложении WPF.

2 Контрольные вопросы

2.1 Что такое «привязка данных»?

Привязка данных — это процесс установки соединения между пользовательским интерфейсом и отображаемыми данными.

2.2 Каков шаблон настройки привязки данных?

{Binding ElementName=Имя_объекта-источника, Path=Свойство_объекта-источника}

2.3 Какой интерфейс надо реализовать для создания конвертера значений?

IValueConverter

2.4 Какой интерфейс надо реализовать для валидации данных при привязке?

IDataErrorInfo

3 Выводы

3.1 Научилась выполнять привязку данных в приложении WPF.