АКТ (ф) СПб ГУТ

Отчет

По лабораторным работам №1-30

Системное программирование

Студент ИСПП-21 Миклякова С.А

Преподаватель Садовский Р.В

Архангельск 2025

# Лабораторная работа №1

## Изучение процесса разработки линейных алгоритмов на языке ассемблера

1. Цель работы
   1. Изучить процесс разработки линейных алгоритмов на языке ассемблера;
   2. Научиться выполнять вычисление математических выражений на языке ассемблера;
   3. Закрепить навык отладки приложений в MS Visual Studio
2. Ответы на контрольные вопросы
   1. Для чего применяется команда mov?

Команда mov используется для копирования данных из одного места в другое. Она переносит значение из источника в назначение, например, из регистра в память, из памяти в регистр или между регистрами. Это основная команда для передачи данных в ассемблере.

* 1. Какие арифметические команды применяются в языке ассемблера и какое у них назначение?

Основные арифметические команды включают:

add — сложение двух операндов.

sub — вычитание одного операнда из другого.

mul — умножение (обычно умножение на регистре или с использованием специальных команд).

div — деление.

inc — увеличение значения регистра или переменной на 1.

dec — уменьшение значения на 1.

Эти команды позволяют выполнять базовые арифметические операции для обработки данных.

* 1. Какие побитовые команды применяются для ускорения умножения и деления и когда они применимы?

Побитовые операции, такие как:

shl (сдвиг влево) — эквивалент умножения на 2^n, где n — число сдвигов.

shr (сдвиг вправо) — эквивалент деления на 2^n (для unsigned чисел).

Эти операции применимы, когда нужно быстро умножить или разделить целое число на степень двойки, что значительно быстрее обычных арифметических операций.

* 1. Каков размер в байтах регистров общего назначения EAX, AX, AH, AL?

EAX — 4 байта (32 бита).

AX — 2 байта (16 бит), часть EAX.

AH — 1 байт (старший байт AX).

AL — 1 байт (младший байт AX).

* 1. Какие способы обнуления регистров могут применяться в языке ассемблера?

Обнулить регистр можно разными способами:

Прямым присваиванием нуля (mov reg, 0).

Использованием команды xor reg, reg, которая сбрасывает регистр в ноль и часто является более быстрым способом.

* 1. Можно ли записать значение переменной типа int в регистр AX и почему?

Можно, если значение переменной типа int помещается по размеру в регистр AX (то есть не превышает 16 бит). В этом случае можно загрузить младшие 16 бит переменной в регистр AX с помощью команды типа mov ax, [variable]. Однако полностью переменную типа int, которая занимает 4 байта (32 бита), нельзя полностью поместить только в AX без дополнительных операций.

# Лабораторная работа №2

## Изучение процесса разработки разветвляющихся алгоритмов на языке ассемблера

1. Цель работы
   1. Изучить процесс разработки разветвляющихся алгоритмов на языке ассемблера;
   2. Закрепить навык отладки приложений в MS Visual Studio.
2. Ответы на контрольные вопросы
   1. Какие логические команды поддерживаются в языке asm?

В языке ассемблера поддерживаются следующие логические команды:

and — побитовое И.

or — побитовое ИЛИ.

xor — побитовое исключающее ИЛИ.

not — побитовое отрицание (инверсия).

Эти команды применяются для выполнения логических операций над битами данных.

* 1. Какая команда осуществляет сравнение двух операндов?

Команда cmp используется для сравнения двух операндов. Она выполняет вычитание одного операнда из другого, но без сохранения результата, только устанавливает флаги процессора, которые затем могут использоваться для условных переходов.

* 1. Какие операторы условного перехода имеются в языке asm?

В ассемблере есть множество условных переходов, основанных на флагах процессора, например:

je (jump if equal) — перейти, если равны (флаги указывают на равенство).

jne (jump if not equal) — перейти, если не равны.

jg (jump if greater) — перейти, если больше (для знаковых чисел).

jge — перейти, если больше или равно.

jl — перейти, если меньше.

jle — перейти, если меньше или равно.

ja, jae, jb, jbe и др., для беззнаковых сравнений.

* 1. Какой оператор безусловного перехода имеется в языке asm?

Безусловный переход реализуется командой:

jmp — прыжок без условий на указанную метку или адрес.

# Лабораторная работа №3

## Изучение процесса разработки циклов со счетчиком на языке ассемблера

1. Цель работы
   1. Изучить процесс разработки циклов со счетчиком на языке ассемблера;
   2. Закрепить навык отладки приложений в MS Visual Studio
2. Ответы на контрольные вопросы
   1. Какие команды asm требуются для организации цикла со счетчиком?

Для организации цикла со счетчиком в ассемблере обычно используют команды:

Инициализация счетчика (например, через MOV)

Команда сравнения (CMP) для проверки условия

Команда перехода (JMP, JE, JNE, JL, JG и т.п.) для повторения цикла или выхода из него

* 1. Как на языке ассемблера реализовать досрочный выход из цикла?

Для досрочного выхода из цикла в ассемблере используют безусловный переход (JMP) или условные переходы (JE, JNE, JL, JG и др.) к метке, расположенной после цикла.

* 1. Какие команды asm требуются, чтобы увеличить значение счетчик на 1 и на большее значение (для увеличения значения счетчика)?

Чтобы увеличить значение счетчика на 1, используется команда INC:

INC CX

Для увеличения на большее значение — команда ADD:

ADD CX, 5

* 1. Какие команды asm требуются, чтобы уменьшить значение счетчик на 1 и на большее значение (для уменьшения значения счетчика)?

Чтобы уменьшить значение счетчика на 1, используется команда DEC:

DEC CX

Для уменьшения на большее значение — команда SUB:

SUB CX, 3

# Лабораторная работа №4

## Изучение процесса разработки разветвляющихся алгоритмов на языке ассемблера

1. Цель работы
   1. Изучить процесс разработки циклов с предусловием и постусловием на языке ассемблера;
   2. Закрепить навык отладки приложений в MS Visual Studio.
2. Ответы на контрольные вопросы
   1. Какое минимальное количество раз выполняется цикл с предусловием?

Цикл с предусловием (например, while) может не выполниться вовсе, если условие изначально ложно. То есть, минимальное количество раз — 0.

* 1. Какое минимальное количество раз выполняется цикл с постусловием?

Цикл с постусловием (например, do-while) обязательно выполнится хотя бы один раз, потому что условие проверяется после выполнения тела цикла. Минимальное количество раз — 1.

* 1. Какова общая форма цикла с предусловием на языке ассемблера?

assembly

start:

; выполнение тела цикла

; например, какие-то инструкции

cmp condition\_register, 0 ; сравнение условия

jne start ; переход обратно, если условие истинно

То есть:

Метка начала цикла.

Выполнение тела.

Проверка условия (cmp и условный переход).

Если условие выполняется — переход к началу.

* 1. Какова общая форма цикла с постусловием на языке ассемблера?

Общая структура:

assembly

do\_start:

; выполнение тела цикла

; например, инструкции

cmp condition\_register, 0 ; проверка условия

jne do\_start ; переход обратно, если условие истинно

То есть:

Метка начала выполнения тела.

Выполнение тела цикла.

Проверка условия (cmp и условный переход).

Если условие истинно — повторить цикл.

# Лабораторная работа №5

## Изучение принципов работы математического сопроцессора

1. Цель работы
   1. Изучить принципы работы сопроцессора и методы его программирования средствами ассемблера.
   2. Закрепить навык отладки приложений в MS Visual Studio.
   3. Написать программу, вычисляющую значения функции 𝑓(𝑥) на языке ассемблера при вещественных значениях x из промежутка [x1, x2] с шагом 1. Значения x1 и x2 вводятся пользователем. Функции показаны на рисунке 1 (вариант = № ПК).
2. Ответы на контрольные вопросы
   1. Какая команда используется для инициализации сопроцессора?

Для инициализации сопроцессора обычно используется команда finit (или fninit в современных процессорах). Она сбрасывает состояние сопроцессора и подготавливает его к работе.

* 1. Какие команды сопроцессора используются для загрузки констант?

Для загрузки констант в регистры сопроцессора используют команды типа fld (load floating-point value), например:

fld1 — загрузить константу 1.0

fldz — загрузить 0.0 (ноль)

fldl2t, fldl2e, fldpi, и другие — для загрузки специальных констант.

* 1. Что такое ST(0)?

ST(0) — это регистр сопроцессора с плавающей точкой, являющийся текущим или верхним регистром стека. В архитектуре x87 FPU регистры организованы как стек, и ST(0) — это вершина этого стека, куда загружаются или из которой извлекаются данные.

* 1. Какие арифметические команды имеются в сопроцессоре?

Основные арифметические команды:

fadd — сложение

fsub — вычитание

fmul — умножение

fdiv — деление

Также есть их вариации с указанием конкретных регистров или операндов, например: faddp, fsubp, которые выполняют операцию и удаляют операнды из стека.

* 1. Какие команды сопроцессора используются для передачи данных?

Для передачи данных между основным процессором и сопроцессором используют команды:

fld — загрузка данных в регистр сопроцессора из памяти или регистра общего назначения.

fstp — сохранение данных из регистра сопроцессора в память и очистка регистра.

fxch — обмен значениями между регистрами сопроцессора (ST(i) и ST(0)).

# Лабораторная работа №6

## Изучение принципов работы цепочечных команд

1. Цель работы
   1. Изучить принципы обработки цепочек данных средствами ассемблера.
   2. Закрепить навык отладки приложений в MS Visual Studio.
2. Ответы на контрольные вопросы
   1. Для чего в операциях с цепочками используется флаг DF?

Флаг DF (Direction Flag) определяет направление обработки цепочки данных при выполнении строковых операций.

Если DF = 0, то указатели (например, SI/ESI для источника и DI/EDI для приемника) увеличиваются после каждой операции — обработка идет «вперед» (с увеличением адресов).

Если DF = 1, то указатели уменьшаются — обработка идет «назад» (с уменьшением адресов).

Это позволяет обрабатывать строки либо с начала к концу, либо с конца к началу.

* 1. Для чего используются префиксы в операциях с цепочками данных?

Префиксы в строковых операциях служат для расширения или изменения поведения команды:

Префиксы повторения (REP, REPE/REPZ, REPNE/REPNZ) позволяют повторять операцию над цепочкой данных заданное количество раз (в регистре CX/ECX), пока не выполнится условие (например, пока не встретится несовпадение или пока CX не станет 0).

Префиксы сегмента могут менять сегментный регистр, используемый для адресации источника или приемника.

Таким образом, префиксы позволяют эффективно обрабатывать большие объемы данных и управлять логикой повторения.

* 1. Какие команды используются для сравнения цепочек данных?

Основная команда для сравнения строк — это:

CMPS (Compare String) — сравнивает байт/слово/двойное слово по адресам, заданным SI/ESI (источник) и DI/EDI (приемник).

В сочетании с префиксами REPE/REPZ или REPNE/REPNZ позволяет сравнивать цепочку данных до первого несовпадения или до окончания заданного количества элементов.

* 1. Операнды каких размеров могут использоваться в операциях с цепочками данных?

В строковых операциях могут использоваться операнды следующих размеров:

8 бит (байт) — команды с суффиксом B (например, MOVSB)

16 бит (слово) — команды с суффиксом W (например, MOVSW)

32 бит (двойное слово) — команды с суффиксом D (например, MOVSD)

В современных 64-битных процессорах также возможна работа с 64-битными операндами, но классические строковые команды ограничены 8/16/32 битами.

* 1. Какие регистры предназначены для задания источника и приемника в операциях с цепочками данных?

Для адресации источника и приемника в строковых операциях используются:

Источник: регистр SI (16 бит) или ESI (32 бит)

Приемник: регистр DI (16 бит) или EDI (32 бит)

Эти регистры указывают на текущие адреса в сегментах данных/дополнительных сегментах, откуда берутся или куда записываются данные.

# Лабораторная работа №7

## Изучение процесса разработки модулей на языке ассемблера

1. Цель работы
   1. Изучить процесс разработки модулей с использованием ассемблера MASM.
2. Ответы на контрольные вопросы
   1. Из каких этапов состоит процесс разработки программ на языке ассемблера?

Процесс разработки программы на ассемблере обычно включает следующие этапы:

Написание исходного кода — создание текстового файла с инструкциями ассемблера.

Ассемблирование — преобразование исходного кода в объектный файл с машинным кодом (с помощью ассемблера, например MASM, NASM).

Компоновка (линковка) — объединение объектных файлов и библиотек в исполняемый файл.

Загрузка и выполнение — запуск программы на целевой машине.

Отладка и тестирование — проверка и исправление ошибок.

* 1. Из каких сегментов состоит программа на языке ассемблера и как объявляется каждый из сегментов?

Классическая программа на ассемблере состоит из нескольких сегментов:

Сегмент кода (Code Segment) — содержит машинные инструкции. Объявляется как:

.code

или в MASM:

.code

Сегмент данных (Data Segment) — содержит инициализированные данные (переменные). Объявляется как:

.data

Сегмент неинициализированных данных (BSS Segment) — содержит неинициализированные переменные. Объявляется как:

.bss

Сегмент стека (Stack Segment) — для организации стека. Объявляется как:

.stack

В MASM сегменты объявляются с помощью директив .code, .data, .stack и т.п.

* 1. Как в программе на языке ассемблера объявить целое число, строку, массив?

Для объявления целого числа используют директивы DB (байт), DW (слово, 16 бит), DWORD (двойное слово, 32 бита), например: myVar DWORD 1234. Строка объявляется как последовательность байтов с нулевым терминатором, например: myString DB 'Hello, world', 0. Массив объявляется перечислением значений, например: myArray DWORD 10, 20, 30, 40.

* 1. Для чего предназначен сегмент кода программы на языке ассемблера?

Сегмент кода предназначен для хранения исполняемых инструкций программы — команд процессору, реализующих логику приложения.

* 1. Что такое MASM?

MASM (Microsoft Macro Assembler) — макроассемблер компании Microsoft для архитектуры x86, поддерживающий макросы, структуры и удобные средства организации ассемблерного кода.

* 1. Каков синтаксис вызова процедур в программах на языке ассемблера?

Вызов процедуры осуществляется командой CALL с именем процедуры, возврат — командой RET. Процедура объявляется директивами PROC и ENDP. Пример: CALL MyProcedure ... MyProcedure PROC ... RET ... MyProcedure ENDP.

* 1. Какие способы передачи параметров между процедурами используются в программах на языке ассемблера?

Параметры между процедурами передаются через стек (помещаются в стек командой PUSH перед вызовом), через регистры (например, AX, BX и др.) или через глобальные переменные. Наиболее распространённый способ — передача параметров через стек.

# Лабораторная работа №8

## Дизассемблирование приложений

1. Цель работы
   1. Научиться применять дизассемблеры для изучения и модификации ПО.
2. Ответы на контрольные вопросы
   1. Что такое «дизассемблирование»?

Дизассемблирование — это процесс преобразования машинного кода (исполняемого файла) обратно в ассемблерный код, который более понятен человеку.

* 1. Для чего выполняется дизассемблирование программного кода?

Дизассемблирование выполняется для анализа, изучения или отладки программного кода, особенно когда исходный код недоступен. Это помогает понять логику работы программы, выявить ошибки или исследовать вредоносное ПО.

* 1. Что такое «дизассемблер»?

Дизассемблер — это программа, которая выполняет процесс дизассемблирования, то есть переводит машинный код в ассемблерный язык.

* 1. Какие существуют программы-дизассемблеры?

Существуют различные программы-дизассемблеры, например, IDA Pro, OllyDbg, Radare2, Ghidra, Hopper, x64dbg.

* 1. Как открыть окно дизассемблированного программного кода в Visual Studio?

В Visual Studio окно дизассемблированного кода можно открыть во время отладки, выбрав в меню Debug пункт Windows, затем Disassembly (Отладка — Окна — Дизассемблирование). Также можно вызвать дизассемблер через контекстное меню, если исходный код недоступен.

* 1. Для чего выполняется обфускация кода?

Обфускация кода выполняется для усложнения анализа и понимания программного кода третьими лицами, чтобы защитить интеллектуальную собственность, предотвратить реверс-инжиниринг и повысить безопасность программного обеспечения.

# Лабораторная работа №9

## Создание проекта в эмуляторе Arduino

1. Цель работы
   1. Научиться создавать имитационные модели проектов, работающих под управлением микроконтроллеров для Arduino;
   2. Получить навыки работы с платой Arduino Uno и макетной платой
2. Ответы на контрольные вопросы
   1. Что такое и для чего используется «Arduino Uno»?

Arduino Uno — это популярная микроконтроллерная плата на базе микроконтроллера ATmega328, используемая для создания электронных проектов и прототипирования. Она позволяет легко управлять датчиками, светодиодами, моторами и другими устройствами.

* 1. Что такое «макетная плата» и для чего она используется?

Макетная плата — это плата с множеством отверстий и соединительных дорожек, предназначенная для быстрого и удобного монтажа и тестирования электронных схем без пайки.

* 1. Для чего предназначен каждый из типов рельс на макетной плате?

На макетной плате обычно есть два типа рельсов: питание (power rails) и сигнальные рельсы. Рельсы питания служат для подачи напряжения (+ и –) по всей длине платы, а сигнальные рельсы используются для соединения компонентов внутри схемы.

* 1. Как соединить элементы на макетной плате последовательно?

Для последовательного соединения элементов на макетной плате нужно соединить выход одного элемента с входом следующего, используя соседние отверстия в одной линии или соединяя их проводниками таким образом, чтобы ток проходил через все элементы один за другим.

* 1. Как соединить элементы на макетной плате параллельно?

Для параллельного соединения элементы подключаются так, чтобы их входы были соединены вместе, а выходы — вместе, то есть все элементы получают одинаковое напряжение, а ток делится между ними. На макетной плате это достигается соединением соответствующих рельсов или линий.

* 1. В каком порядке должно выполняться подключение элементов на схеме?

Подключение элементов на схеме должно выполняться в логическом порядке: сначала подключается питание (источник напряжения), затем элементы управления и нагрузки, соблюдая правильную полярность и последовательность для корректной работы схемы. Важно также соблюдать правила безопасности и избегать коротких замыканий.

# Лабораторная работа №10

## Разработка скетчей для Arduino

1. Цель работы
   1. Научиться создавать скетчи (программное обеспечение на языке Си) для микроконтроллеров Arduino;
   2. Получить навыки работы с платой Arduino Uno и макетной платой
2. Ответы на контрольные вопросы
   1. Для чего предназначены и когда вызываются функции setup() и loop()?

Функция setup() предназначена для инициализации настроек и выполняется один раз при запуске программы на Arduino. Функция loop() содержит основной код, который выполняется циклично бесконечно после выполнения setup().

* 1. Для чего используется функция pinMode(...), какие параметры она принимает?

Функция pinMode(...) используется для настройки режима работы конкретного пина (вывода) Arduino. Она принимает два параметра: номер пина (целое число) и режим работы (например, INPUT, OUTPUT или INPUT\_PULLUP).

* 1. Для чего используется функция digitalWrite(...), какие параметры она принимает?

Функция digitalWrite(...) используется для установки логического уровня на цифровом пине. Она принимает два параметра: номер пина и значение уровня (HIGH — высокий уровень, LOW — низкий уровень).

* 1. Какие функции используются для реализации программной задержки, какие параметры принимают эти функции?

Для реализации программной задержки используется функция delay(), которая принимает один параметр — длительность задержки в миллисекундах (целое число). Также существует функция delayMicroseconds(), принимающая задержку в микросекундах.

* 1. Что такое «скетч»?

Скетч — это исходный код программы для Arduino, написанный на языке, основанном на C/C++, который загружается и выполняется на плате Arduino.

* 1. Как подключить библиотеки к скетчу?

Для подключения библиотек к скетчу используется директива #include, например: #include <LibraryName.h>. Также библиотеки можно подключать через меню среды Arduino IDE: Sketch → Include Library → Manage Libraries или выбрать из установленных.

# Лабораторная работа №11

## Разработка приложений для обработки файлов

1. Цель работы
   1. Научиться использовать файловые потоки в приложении на C#;
   2. Научиться применять классы для работы с файлами в приложениях на C#.
2. Ответы на контрольные вопросы
   1. Какое пространство имен содержит себе классы для работы с файловой системой и файловыми потоками?

Пространство имен, которое содержит классы для работы с файловой системой и файловыми потоками в C#, — это System.IO.

* 1. Для чего предназначены классы StreamReader и StreamWriter?

Классы StreamReader и StreamWriter предназначены для чтения и записи текстовых данных из/в потоки или файлы. StreamReader используется для последовательного чтения символов из текстового файла, а StreamWriter — для записи текста.

* 1. Для чего предназначены классы BinaryReader и BinaryWriter?

Классы BinaryReader и BinaryWriter используются для чтения и записи данных в бинарном формате. Они позволяют работать с примитивными типами данных (int, double, bool и др.) в бинарном виде, что полезно для эффективного хранения и передачи.

* 1. Какие классы предоставляют информацию о файлах?

Для получения информации о файлах используются классы FileInfo и DirectoryInfo из пространства имен System.IO. Они предоставляют свойства и методы для получения атрибутов файлов, размеров, дат создания и изменения, а также для управления файлами и каталогами.

# Лабораторная работа №12

## Разработка приложений для поиска файлов

1. Цель работы
   1. Научиться применять классы для работы с файлами и каталогами в приложениях на C#.
2. Ответы на контрольные вопросы
   1. В чем отличие между классами Directory и DirectoryInfo?

Отличие между классами Directory и DirectoryInfo:

Directory — статический класс, предоставляющий методы для работы с каталогами (создание, удаление, получение списка файлов и папок и т.д.) без необходимости создавать объект. Все методы вызываются напрямую через класс.

DirectoryInfo — класс экземпляра, который представляет конкретный каталог и позволяет получать информацию о нем, а также выполнять операции (создание, удаление и т.п.) через объект. Позволяет сохранять состояние и использовать свойства каталога.

* 1. В чем отличие между классами File и FileInfo?

Класс File — статический и содержит методы для работы с файлами без создания объектов, в то время как FileInfo — класс экземпляра, представляющий конкретный файл и предоставляющий информацию о нем и методы управления через объект.

* 1. Как получить список файлов и папок определенного каталога?

Для получения списка файлов и папок определенного каталога используют методы Directory.GetFiles(path) и Directory.GetDirectories(path), а также методы DirectoryInfo.GetFiles() и DirectoryInfo.GetDirectories().

* 1. Как задать шаблон поиска и опции поиска файлов и папок?

Для задания шаблона поиска и опций используют параметры searchPattern и searchOption в методах, например, Directory.GetFiles(path, ".txt", SearchOption.AllDirectories), где ".txt" — шаблон поиска, а SearchOption.AllDirectories — опция поиска во всех подкаталогах.

* 1. Какие свойства класса FileInfo позволяют получить информацию о файле?

Свойства класса FileInfo для получения информации о файле включают Name, FullName, Length, CreationTime, LastAccessTime, LastWriteTime, Extension и Attributes.

# Лабораторная работа №13

## **Разработка приложений для сортировки файлов**

1. Цель работы
   1. Научиться применять классы для работы с файлами и каталогами в приложениях на C#.
2. Ответы на контрольные вопросы
   1. Как проверить существование файла по его имени?

Для проверки существования файла по его имени используют метод File.Exists(путь\_к\_файлу), который возвращает true, если файл существует, и false — если нет.

* 1. Как проверить существование каталога по его имени?

Для проверки существования каталога по его имени используют метод Directory.Exists(путь\_к\_каталогу), возвращающий true, если каталог существует, и false — если нет.

* 1. Какие методы позволяют создавать, удалять, копировать и переносить каталоги?

Методы для работы с каталогами:

Создание: Directory.CreateDirectory(путь)

Удаление: Directory.Delete(путь) (с параметром true можно удалить каталог с содержимым)

Копирование: в стандартной библиотеке .NET нет прямого метода для копирования каталогов, обычно копирование реализуют рекурсивно самостоятельно.

Перемещение: Directory.Move(старый\_путь, новый\_путь)

* 1. Какие методы позволяют создавать, удалять, копировать и переносить файлы?

Методы для работы с файлами:

Создание: File.Create(путь) (возвращает FileStream), или запись через File.WriteAllText/WriteAllBytes

Удаление: File.Delete(путь)

Копирование: File.Copy(исходный\_путь, целевой\_путь, перезаписать\_если\_существует)

Перемещение: File.Move(старый\_путь, новый\_путь)

# Лабораторная работа №14

## Разработка утилиты для поиска дубликатов файлов

1. Цель работы
   1. Научиться получать и анализировать информацию о файлах в приложениях на C#.
2. Ответы на контрольные вопросы
   1. Какое свойство FileInfo возвращает имя файла?

Свойство FileInfo, возвращающее имя файла, называется Name.

* 1. Какое свойство FileInfo возвращает расширение файла?

Свойство FileInfo, возвращающее расширение файла, называется Extension.

* 1. Какое свойство FileInfo возвращает полное имя файла?

Свойство FileInfo, возвращающее полное имя файла (полный путь с именем), называется FullName.

* 1. Какое свойство FileInfo возвращает дату изменения файла?

Свойство FileInfo, возвращающее дату изменения файла, называется LastWriteTime.

# Лабораторная работа №15

## Разработка утилиты для просмотра изображений

1. Цель работы
   1. Научиться отображать файлы-изображения в приложениях на C#.
2. Ответы на контрольные вопросы
   1. Для чего предназначен класс BitmapImage?

Класс BitmapImage предназначен для загрузки и хранения растрового изображения в памяти, которое можно использовать в приложениях для отображения графики.

* 1. Для чего предназначен элемент управления Image?

Элемент управления Image предназначен для отображения изображений в пользовательском интерфейсе, позволяя показывать графические файлы или объекты BitmapImage.

* 1. Для чего предназначен элемент управления ScrollView?

Элемент управления ScrollView предназначен для создания области с прокруткой, позволяя размещать внутри него элементы, которые могут выходить за пределы видимой области, и обеспечивать их прокрутку пользователем.

* 1. Как получить высоту и ширину изображения BitmapImage?

Высоту и ширину изображения BitmapImage можно получить через свойства PixelHeight и PixelWidth соответственно.

# Лабораторная работа №16

## **Разработка утилиты Диспетчер задач**

1. Цель работы
   1. Научиться получать и отображать информацию о запущенных процессах в приложениях на C#.
2. Ответы на контрольные вопросы
   1. Какое пространство имен требуется подключить для работы с процессами?

Для работы с процессами требуется подключить пространство имен System.Diagnostics.

* 1. Какой метод запускает процесс?

Для запуска процесса используется метод Process.Start(). Его можно вызвать как статически с указанием имени файла или команды, так и через экземпляр класса Process.

* 1. Какой метод завершает процесс?

Для завершения процесса используется метод экземпляра Process.Kill().

* 1. Как получить список запущенных процессов?

Для получения списка запущенных процессов применяется статический метод Process.GetProcesses(), который возвращает массив объектов Process.

* 1. Как получить процесс по его идентификатору?

Для получения процесса по его идентификатору (PID) используется метод Process.GetProcessById(int processId).

# Лабораторная работа №17

## Разработка утилиты Файловый менеджер

1. Цель работы
   1. Научиться применять элементы управления для отображения файлов и папок в приложениях на C#.
2. Ответы на контрольные вопросы
   1. Как получить список дисков?

Для получения списка дисков используют метод DriveInfo.GetDrives() из пространства имен System.IO. Он возвращает массив объектов DriveInfo, представляющих доступные диски.

* 1. Как получить список файлов?

Для получения списка файлов в каталоге применяют метод Directory.GetFiles(путь) или DirectoryInfo.GetFiles(); оба возвращают список файлов в указанной директории.

* 1. Для чего предназначен элемент управления TreeView?

Элемент управления TreeView предназначен для отображения иерархической структуры данных в виде дерева, позволяя пользователю разворачивать и сворачивать узлы для удобной навигации по вложенным элементам.

# Лабораторная работа №18

## Разработка утилиты для анализа дисков

1. Цель работы
   1. Научиться получать и отображать статистическую информацию о дисковом пространстве в приложениях на C#.
2. Ответы на контрольные вопросы
   1. Какой класс предоставляет информацию о дисках?

Класс, предоставляющий информацию о дисках — это DriveInfo (пространство имен System.IO).

* 1. Какой класс предоставляет информацию о файлах?

Класс, предоставляющий информацию о файлах — это FileInfo (пространство имен System.IO).

* 1. Какой метод позволяет просуммировать значения в списке?

Для суммирования значений в списке можно использовать метод расширения LINQ Sum(), например: list.Sum().

* 1. Какие методы позволяют отсортировать список по возрастанию и убыванию?

Для сортировки списка по возрастанию и убыванию применяют методы расширения LINQ:

По возрастанию: OrderBy()

По убыванию: OrderByDescending()

# Лабораторная работа №19

## Разработка утилиты Архиватор

1. Цель работы
   1. Научиться применять классы для работы с архивами в приложениях на C#.
2. Ответы на контрольные вопросы
   1. Какие пространства имен нужно подключить для работы с архивами?

Для работы с архивами в .NET обычно подключают пространство имен System.IO.Compression и System.IO.Compression.FileSystem.

* 1. Какие ссылки нужно подключить для работы с архивами?

Для работы с архивами необходимо добавить ссылку на сборку System.IO.Compression.dll и System.IO.Compression.FileSystem.dll (в некоторых проектах, например, в .NET Framework).

* 1. Для чего предназначен класс ZipArchive?

Класс ZipArchive предназначен для представления zip-архива в памяти, позволяя читать, создавать и изменять содержимое архива.

* 1. Для чего предназначен класс ZipFile?

Класс ZipFile предоставляет статические методы для удобного создания, извлечения и управления zip-архивами на основе файловой системы.

* 1. Для чего предназначен класс ZipArchiveEntry?

Класс ZipArchiveEntry представляет отдельный файл или папку внутри zip-архива, предоставляя доступ к его имени, содержимому и другим атрибутам.

# Лабораторная работа №20

## Разработка утилиты скринсейвер

1. Цель работы
   1. Научиться разрабатывать оконные приложения-заставки, используя таймер.
2. Ответы на контрольные вопросы
   1. Что такое «утилита»?

«Утилита» — это небольшая специализированная программа, предназначенная для выполнения определённых вспомогательных задач по обслуживанию, настройке или диагностике компьютерной системы.

* 1. Что такое «скринсейвер»?

«Скринсейвер» — это программа-заставка, которая автоматически запускается при бездействии пользователя и отображает анимацию или изображение на экране, чтобы предотвратить выгорание экрана и обеспечить эстетический вид.

* 1. Какое расширение должно быть у файлов-заставок в Windows?

Файлы-заставки в Windows должны иметь расширение .scr.

# Лабораторная работа №21

## Разработка утилиты для вычисления хэш-суммы файлов

1. Цель работы
   1. Научиться вычислять хэш-суммы файлов в приложениях на C#, используя встроенные алгоритмы шифрования.
2. Ответы на контрольные вопросы
   1. Что такое хэш-сумма?

Хэш-сумма — это фиксированного размера строка (или число), получаемая в результате применения хэш-функции к данным. Она служит уникальным цифровым «отпечатком» исходного содержимого.

* 1. Для чего применяются хэш-суммы файлов?

Хэш-суммы файлов применяются для проверки целостности и подлинности данных, позволяя обнаружить изменения или повреждения файла при передаче или хранении.

* 1. Какие алгоритмы применяются для вычисления хэш-суммы файлов?

Для вычисления хэш-суммы файлов часто используют криптографические алгоритмы, такие как MD5, SHA-1, SHA-256 и другие семейства SHA (Secure Hash Algorithm).

# Лабораторная работа №22

## Разработка утилиты Менеджер паролей

1. Цель работы
   1. Научиться выполнять шифрование и дешифрование данных в приложениях на C#, используя встроенные алгоритмы шифрования.
2. Ответы на контрольные вопросы
   1. Что такое «шифрование»?

«Шифрование» — это процесс преобразования исходных данных (открытого текста) в зашифрованный формат (шифротекст) с целью защиты информации от несанкционированного доступа.

* 1. Что такое «дешифрование»?

«Дешифрование» — это обратный процесс преобразования зашифрованных данных (шифротекста) обратно в исходный читаемый формат (открытый текст).

* 1. Что такое «AES»?

«AES» (Advanced Encryption Standard) — это симметричный алгоритм блочного шифрования, широко используемый для защиты данных благодаря высокой безопасности и эффективности.

* 1. Какими могут быть размеры ключа в алгоритме AES?

В алгоритме AES размеры ключа могут быть 128, 192 или 256 бит.

* 1. Какое пространство имен требуется подключить для применения стандартных алгоритмов шифрования?

Для применения стандартных алгоритмов шифрования в .NET требуется подключить пространство имен System.Security.Cryptography.

# Лабораторная работа №23

## Изучение процесса разработки DLL

1. Цель работы
   1. Изучить процесс создания, применения, тестирования и документирования библиотеки классов в Microsoft Visual Studio.
2. Ответы на контрольные вопросы
   1. Что такое «библиотека классов»?

«Библиотека классов» — это набор связанных классов, интерфейсов и других типов, упакованных в один компонент, предназначенный для повторного использования в различных приложениях.

* 1. Какое расширение у файлов библиотек?

Расширение файлов библиотек в .NET — .dll (Dynamic Link Library).

* 1. Как на C# создать библиотеку классов в Visual Studio?

В Visual Studio для создания библиотеки классов на C# нужно:

Создать новый проект и выбрать шаблон «Class Library» (или «Библиотека классов»).

Написать необходимые классы и методы.

Собрать проект — будет сгенерирован файл .dll.

* 1. Как подключить библиотеку к проекту?

Для подключения библиотеки к проекту в Visual Studio:

Щёлкнуть правой кнопкой по проекту → «Добавить» → «Ссылка» (Add Reference).

В открывшемся окне выбрать нужный .dll файл или проект из решения.

Подтвердить добавление ссылки.

* 1. Что такое «XML-документация»?

«XML-документация» — это специальный формат комментариев в исходном коде C#, который позволяет автоматически генерировать документацию по классам, методам и другим элементам кода.

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

Чтобы XML-документация была видна при использовании библиотеки в стороннем проекте, необходимо:

В настройках проекта библиотеки включить генерацию XML-документации (в свойствах проекта → вкладка «Сборка» → отметить «XML-документ»).

При сборке проекта будет создан файл .xml с документацией рядом с .dll.

При подключении библиотеки в стороннем решении убедиться, что файл .xml находится в той же папке, что и .dll — тогда среда разработки сможет отображать подсказки и документацию.

# Лабораторная работа №24

## Использование потоков

1. Цель работы
   1. Научиться разрабатывать многопоточные приложения на C#;
   2. Научиться создавать и применять фоновые и основные потоки и выполнять обмен данных между ними в программах на C#.
2. Ответы на контрольные вопросы
   1. В чем отличие потока от процесса?

Отличие потока от процесса:

Процесс — это изолированная единица выполнения, содержащая собственное адресное пространство, ресурсы и минимум один поток.

Поток (thread) — это наименьшая единица выполнения внутри процесса, которая разделяет с другими потоками процесса память и ресурсы, но имеет собственный стек и регистры.

* 1. Какие преимущества дает многопоточная архитектура?

Преимущества многопоточной архитектуры:

Повышение производительности за счёт параллельного выполнения задач.

Улучшение отзывчивости приложений (например, интерфейс остаётся активным при выполнении фоновых операций).

Более эффективное использование многоядерных процессоров.

Возможность разделения сложных задач на более простые параллельные.

* 1. Какие существуют основные средства синхронизации потоков?

Основные средства синхронизации потоков:

Мьютексы (Mutex)

Семафоры (Semaphore)

Мониторы (Monitor, ключевое слово lock в C#)

События (EventWaitHandle, AutoResetEvent, ManualResetEvent)

Барьеры (Barrier)

* 1. Каким образом на однопроцессорных компьютерах исполняются многопоточные приложения?

На однопроцессорных компьютерах многопоточные приложения исполняются посредством псевдопараллелизма: операционная система быстро переключает выполнение между потоками (контекстное переключение), создавая иллюзию одновременной работы.

* 1. Для чего в С# используется класс Thread?

В C# класс Thread используется для создания, управления и контроля отдельных потоков исполнения, позволяя запускать код параллельно, останавливать, приостанавливать и возобновлять потоки.

# Лабораторная работа №25

## Разработка и вызов асинхронных методов

1. Цель работы
   1. Научиться реализовывать и запускать асинхронные операции на C#;
   2. Научиться выполнять вычисления, используя асинхронные операции на C#.
2. Ответы на контрольные вопросы
   1. Какие ключевые слова используются в C# для работы с асинхронными вызовами?

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

async и await

lock и volatile

try и catch

static и virtual

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

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

void — для методов без возвращаемого значения, Task — для асинхронных операций без результата, Task<T> — для асинхронных операций с результатом типа T

int — для целочисленных значений, string — для строк, bool — для логических значений

async — для асинхронных методов, await — для ожидания результата

public — для публичных методов, private — для приватных

* 1. Как вызвать метод в асинхронном режиме?

Использовать await перед вызовом метода внутри async метода

Вызвать метод напрямую без await

Использовать ключевое слово lock

Объявить метод как static

* 1. Как указать, что в методе могут быть асинхронные вызовы?

Добавить ключевое слово async в объявление метода

Использовать ключевое слово static

Обернуть метод в try-catch

Использовать ключевое слово override

* 1. Как обработать исключения, возникшие в асинхронных вызовах?

Использовать блок try-catch вокруг await

Игнорировать исключения

Использовать lock для синхронизации

Объявить метод как void

# Лабораторная работа №26

## Обмен данными

1. Цель работы
   1. Научиться реализовывать и запускать асинхронные операции на C#;
   2. Научиться выполнять HTTP-запросы, используя асинхронные операции на C#.
2. Ответы на контрольные вопросы
   1. Для чего используется класс WebClient?

Класс WebClient используется для упрощения взаимодействия с ресурсами в интернете. Он предоставляет удобные методы для загрузки и отправки данных по протоколам HTTP и FTP, позволяя скачивать файлы, получать строки или загружать данные на сервер.

* 1. Какие методы для загрузки данных поддерживаются в классе WebClient?

В классе WebClient поддерживаются следующие методы для загрузки данных:

DownloadString — загружает данные в виде строки;

DownloadData — загружает данные в виде массива байт;

DownloadFile — загружает файл и сохраняет его на диск;

Async-версии этих методов (например, DownloadStringAsync) для асинхронной загрузки.

* 1. Какое пространство имен требуется подключить для работы с WebClient?

Для работы с классом WebClient необходимо подключить пространство имен System.Net.

# Лабораторная работа №27

## Сетевое программирование сокетов

1. Цель работы
   1. Научиться применять сокеты в клиент-серверных приложениях на C#.
2. Ответы на контрольные вопросы
   1. Что такое «сокет»?

Сокет — это программный интерфейс для организации сетевого взаимодействия между компьютерами. Он представляет собой конечную точку для обмена данными по сети и служит для установления соединений и передачи данных по протоколам TCP или UDP.

* 1. Каков алгоритм работы сервера, использующего сокеты?

Алгоритм работы сервера, использующего сокеты:

Создать сокет с указанием параметров (семейство адресов, тип сокета, протокол);

Привязать сокет к локальному IP-адресу и порту (bind);

Начать прослушивание входящих соединений (listen);

Принять входящее соединение от клиента (accept);

Обмениваться данными с клиентом (send, receive);

Закрыть соединение по окончании работы.

* 1. Каков алгоритм работы клиента, использующего сокеты?

Алгоритм работы клиента, использующего сокеты:

Создать сокет с необходимыми параметрами;

Установить соединение с сервером по IP-адресу и порту (connect);

Отправлять и получать данные (send, receive);

Закрыть соединение после завершения обмена.

* 1. Какие пространства имен требуется подключить для работы с сокетами?

Для работы с сокетами в C# необходимо подключить пространства имен:

System.Net;

System.Net.Sockets.

* 1. Какие параметры требуется указать при создании сокета?

При создании сокета требуется указать следующие параметры:

Семейство адресов (например, AddressFamily.InterNetwork для IPv4);

Тип сокета (SocketType.Stream для TCP или SocketType.Dgram для UDP);

Протокол (ProtocolType.Tcp или ProtocolType.Udp).

* 1. Как выполнить получение данных с использованием сокетов?

Для получения данных с использованием сокетов выполняются следующие действия:

Вызвать метод Receive, передав буфер для приема данных;

Метод блокирует выполнение до получения данных или возвращает количество принятых байт;

Обработать полученные данные из буфера;

При необходимости повторять прием для получения полного сообщения.

# Лабораторная работа №28

## Разработка приложения для загрузки и отправки данных по сети

1. Цель работы
   1. Научиться разрабатывать RESTfulAPI для доступа к БД из приложения C#.
2. Ответы на контрольные вопросы
   1. Что такое REST-запрос?

REST-запрос — это HTTP-запрос, который следует принципам REST (Representational State Transfer) архитектуры, предназначенный для взаимодействия с ресурсами веб-сервиса через стандартные HTTP-методы (GET, POST, PUT, DELETE и др.).

* 1. Что такое RESTful?

RESTful — это термин, описывающий веб-сервисы или API, которые реализуют архитектурный стиль REST, соблюдая его принципы, такие как использование стандартных HTTP-методов, адресация ресурсов через URI, отсутствие состояния на сервере и использование представлений ресурсов.

* 1. Для чего в контроллере используется метод GET?

Метод GET в контроллере используется для получения данных с сервера. Он запрашивает ресурс или коллекцию ресурсов и не изменяет состояние сервера.

* 1. Для чего в контроллере используется метод POST?

Метод POST в контроллере используется для создания нового ресурса на сервере или отправки данных, которые должны быть обработаны сервером.

* 1. Для чего в контроллере используется метод PUT?

Метод PUT в контроллере используется для обновления существующего ресурса на сервере или создания ресурса по указанному URI, если он отсутствует.

* 1. Для чего в контроллере используется метод DELETE?

Метод DELETE в контроллере используется для удаления указанного ресурса с сервера.

# Лабораторная работа №29

## Вызов сервисов

1. Цель работы
   1. Научиться проверять работоспособность RESTful API в клиентском приложении.
   2. Научиться выполнять тестирование RESTful API методом черного ящика.
2. Ответы на контрольные вопросы
   1. Как указать у объекта HttpClient ссылку на веб-страницу?

Чтобы указать у объекта HttpClient ссылку на веб-страницу, используют метод вызова с URL в виде строки, например:

HttpResponseMessage response = await httpClient.GetAsync("https://example.com/page");

* 1. Как указать, что объект HttpClient принимает данные в формате JSON?

Чтобы указать, что объект HttpClient принимает данные в формате JSON, в заголовках запроса устанавливают заголовок Accept:

httpClient.DefaultRequestHeaders.Accept.Clear();

httpClient.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));

* 1. Для чего в объекте HttpClient используется метод GetAsync?

Метод GetAsync в объекте HttpClient используется для отправки HTTP GET-запроса асинхронно, то есть для получения данных с указанного URI без изменения состояния сервера.

* 1. Для чего в объекте HttpClient используется метод DeleteAsync?

Метод DeleteAsync используется для отправки HTTP DELETE-запроса асинхронно, чтобы запросить удаление ресурса по указанному URI.

* 1. Для чего в объекте HttpClient используется метод PostAsync?

Метод PostAsync применяется для отправки HTTP POST-запроса асинхронно, обычно для создания нового ресурса или передачи данных на сервер.

* 1. Для чего в объекте HttpClient используется метод PutAsync?

Метод PutAsync используется для отправки HTTP PUT-запроса асинхронно, чтобы обновить существующий ресурс или создать ресурс по указанному URI.

# Лабораторная работа №30

## Работа с буфером экрана

1. Цель работы
   1. Научиться настраивать интерфейс консольных приложений на C#;
   2. Научиться применять параметры командной строки в приложениях на C#
2. Ответы на контрольные вопросы
   1. Как изменить цвет фона консольного приложения?

Чтобы изменить цвет фона консольного приложения в C#, используйте свойство Console.BackgroundColor. Например:

Console.BackgroundColor = ConsoleColor.Blue;

Console.Clear(); // Применить изменение ко всему окну

* 1. Как изменить цвет текста консольного приложения?

Чтобы изменить цвет текста в консольном приложении, используйте свойство Console.ForegroundColor:

Console.ForegroundColor = ConsoleColor.Yellow;

* 1. Как изменить стандартный потоковый ввод, перенаправив его на запись в файл?

Чтобы перенаправить стандартный поток ввода (Console.In) на чтение из файла, используйте метод Console.SetIn:

var fileReader = new StreamReader("input.txt");

Console.SetIn(fileReader);

* 1. Как перехватить нажатие произвольной клавиши в консольном приложении C#?

Чтобы перехватить нажатие произвольной клавиши, используйте метод Console.ReadKey:

ConsoleKeyInfo keyInfo = Console.ReadKey(intercept: true); // intercept=true чтобы не выводить символ на экран

// Обработка нажатой клавиши через keyInfo.Key или keyInfo.KeyChar

* 1. Как изменить заголовок окна в консольном приложении C#?

Чтобы изменить заголовок окна консольного приложения, используйте свойство Console.Title:

Console.Title = "Мой заголовок окна";