МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»
(БГТУ им. В.Г. Шухова)

УТВЕРЖДАЮ
Директор института
_____ Константинов И. С.
«27» 05 2025 г.

РАБОЧАЯ ПРОГРАММА
дисциплины

Архитектура вычислительных систем

направление подготовки:

09.03.01 «Информатика и вычислительная техника»

Направленность программы (профиль):

Интеллектуальные системы

Квалификация

Бакалавр

Форма обучения

Очная

Институт информационных технологий и управляющих систем

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Белгород 2025

Рабочая программа составлена на основании требований:

- Федерального государственного образовательного стандарта высшего образования – бакалавриат по направлению подготовки 09.03.01 «Информатика и вычислительная техника», утвержденного приказа Минобрнауки России от 19.09.2017 № 929
- учебного плана, утвержденного ученым советом БГТУ им. В.Г. Шухова в 2023 году.

Составитель : к.ф.-м.н., доцент _____ (О.В. Осипов)
(ученая степень и звание, подпись) (инициалы, фамилия)

Рабочая программа обсуждена на заседании кафедры

«20» 05 2025 г., протокол № 11

Заведующий кафедрой: к.т.н., доцент _____ (В.М. Поляков)
(ученая степень и звание, подпись) (инициалы, фамилия)

Рабочая программа согласована с выпускающей кафедрой программного обеспечения вычислительной техники и автоматизированных систем
(наименование кафедры/кафедр)

Заведующий кафедрой: к.т.н., доцент _____ (В.М. Поляков)
(ученая степень и звание, подпись) (инициалы, фамилия)

«20» 05 2025 г.

Рабочая программа одобрена методической комиссией института

«27» 05 2025 г., протокол № 9

Председатель доцент _____ (Ю.Д. Рязанов)
(ученая степень и звание, подпись) (инициалы, фамилия)

1. ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ ПО ДИСЦИПЛИНЕ

Категория (группа) компетенций	Код и наименование компетенции	Код и наименование индикатора достижения компетенции	Наименование показателя оценивания результата обучения по дисциплине
	ПК-1 Способен разрабатывать требования и проектировать программное обеспечение	ПК-1.1 Анализирует требования к программному обеспечению	Знания, умения, навыки.
		ПК-1.2 Разрабатывает технические спецификации на программные компоненты и их взаимодействие	Знания, умения, навыки.
		ПК-1.3 Проектирует программное обеспечение, в том числе для беспилотных авиационных систем	Знания, умения.

2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

1. Компетенция ПК-1 Способен разрабатывать требования и проектировать программное обеспечение

Данная компетенция формируется следующими дисциплинами.

Стадия	Наименования дисциплины
1.	Алгоритмы и структуры данных
2.	Объектно-ориентированное программирование
3.	Интеллектуальные системы реального времени
4.	Программирование мобильных устройств
5.	Тестирование программных систем
6.	Моделирование систем
7.	Архитектура вычислительных систем
8.	Программирование распределённых систем
9.	Программирование микроконтроллеров
10.	Микропроцессорные системы
11.	Технологии Web-программирования
12.	Программирование мобильной робототехники
13.	Производственная преддипломная практика

3. ОБЪЁМ ДИСЦИПЛИНЫ

Общая трудоёмкость дисциплины составляет 4 зач. единицы, 144 часа.

Форма промежуточной аттестации: дифференцированный зачёт.

Вид учебной работы	Всего часов	Семестр № 5
Общая трудоемкость дисциплины, час	144	144
Контактная работа (аудиторные занятия), в т.ч.:	71	71
лекции	34	34
лабораторные	34	34
практические		
групповые консультации в период теоретического обучения и промежуточной аттестации	3	3
Самостоятельная работа студентов, включая индивидуальные и групповые консультации, в том числе:	73	73
Курсовой проект		
Курсовая работа		
Расчетно-графическое задание	18	18
Индивидуальное домашнее задание		
Самостоятельная работа на подготовку к аудиторным занятиям (лекции, практические занятия, лабораторные занятия)	55	55
Экзамен		

4. СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

4.1. Наименование тем, их содержание и объём
Курс 3 Семестр 5

№ п/п	Наименование раздела (краткое содержание)	Объём на тематический раздел по видам учебной нагрузки, час			
		Лекции	Практические занятия	Лабораторные занятия	Самостоятельная работа на подготовку к аудиторным
1.	Становление и основные тенденции развития вычислительной техники				
	Исторические этапы развития вычислительной техники. ЭВМ на базе электронно-вакуумных ламп. ЭВМ на базе интегральных схем. Становление и развитие вычислительной техники в СССР и за рубежом. Вычислительная техника будущего. Закон Мура. Технологический процесс. Текущее состояние вычислительной техники. Ведущие производители процессоров и графического оборудования.	4	–	2	3
2.	Виды и классификация вычислительных систем				
	Классификация вычислительных систем по Флинну. Классы параллельных вычислительных систем (SMP, MPP, NUMA). Архитектура с расширенным набором команд. Архитектура с сокращённым набором команд. Архитектура x86. SPARC-архитектура. VLIW-архитектура. Производительность вычислительных систем. Тесты для оценки производительности.	6	–	2	3
3.	Архитектура Джона фон Неймана				
	Основные принципы фон Неймановской архитектуры. Тактовый генератор. Цикл выполнения команды. Представление команд в памяти. Принцип хранимой в памяти программы.	4	–	2	4
4.	Устройство процессора				
	Характеристики процессора. Арифметико-логическое устройство. Устройство управления. Регистры и флаги процессора. Арифметические и логические команды процессора для работы с целочисленной арифметикой. Команды передачи управления. Команды для работы со стеком. Цепочечные команды. Команды пересылки данных. 64-разрядная архитектура.	4	–	10	17
5.	Организация шин				
	Характеристики и виды шин. Последовательные и параллельные шины. Централизованный и децентрализованный арбитраж шин. Алгоритмы динамического изменения приоритетов при организации арбитража. Системы ввода/вывода.	4	–	4	3
6.	Память				

	Иерархия памяти компьютера. Оперативная память. Кэш-память. Характеристики и виды оперативной памяти. Сплошная и сегментированная модели памяти. Внешняя память компьютера. Жёсткий диск. Твердотельный накопитель. Кластеризация.	6	–	4	5
7.	Программирование на ассемблере				
	Структура ассемблерной программы. Компиляция и отладка ассемблерных программ с использованием пакета masm32 и отладчика OllyDbg. Представление в сегменте данных чисел, строк, массивов. Директивы объявления данных. Назначение стека. Вызов подпрограмм. Устройство сопроцессора. Стили вызова stdcall, cdecl, fastcall, thiscall, pascal. Разработка dll-библиотек на ассемблере. Статический и динамический способы подключения dll-библиотек.	6	–	10	20
	ВСЕГО	34		34	55

4.2. Содержание практических (семинарских) занятий

Не предусмотрено учебным планом

4.3. Содержание лабораторных занятий

№ п/п	Наименование раздела дисциплины	Тема лабораторного занятия	К-во часов	Самостоятельная работа на подготовку к лабораторным занятиям
семестр №5				
1	Становление и основные тенденции развития вычислительной техники	Разработка программ на ассемблере. Работа с отладчиком OllyDbg и пакетом masm32. Структура программы: сегмент данных, сегмент кода, сегмент стека. Директивы объявления данных. Представление переменных и массивов в сегменте данных. Трассировка программы.	4	4
2	Архитектура Джона фон Неймана, Устройство процессора	Структура команд процессора. Код операции. Представление команд в памяти ЭВМ. Режимы адресации. Построение машинного кода команды по её названию.	4	4
3	Устройство процессора	Арифметические команды центрального процессора. Команды процессора для выполнения сложения, вычитания, умножения и деления целых чисел. Представление в памяти ЭВМ целых знаковых и беззнаковых чисел. Команды изменения размерности и знака числа. Команды пересылки данных.	4	4
4	Организация шин, Программирование на ассемблере	Команды передачи управления. Условный и безусловный переход. Команды для организации циклов. Флаги процессора. Вызов подпрограмм. Соглашение о вызове подпрограмм. Консольный ввод и вывод. Назначение стека. Обработка массивов.	6	6

5	Программирование на ассемблере	Команды сопроцессора для работы с целыми и вещественными числами. Регистры сопроцессора. Алгоритм возведения числа *x* в степень *y* с использованием команд сопроцессора.	2	2
6	Программирование на ассемблере	Логические команды и команды сдвига центрального процессора для работы с целыми числами.	3	3
7	Программирование на ассемблере	Цепочечные команды центрального процессора. Обработка массивов и строк с использованием цепочечных команд.	3	3
8	Виды и классификация вычислительных систем, Память	Способы вызова ассемблерных подпрограмм в языках высокого уровня. Стили вызова подпрограмм fastcall, stdcall, pascal, cdecl. Ассемблерная вставка. Разработка dll-библиотек на ассемблере. Сравнение производительности кода, написанного на ассемблере и кода, полученного компиляторами C++, C#, Pascal.	4	4
9	Память	Принципы формирования объектно-ориентированного кода. Стиль вызова thiscall. Таблица виртуальных методов.	4	4
		ИТОГО:	34	34

4.3. Содержание курсового проекта/работы

Выполнение курсового проекта/работы не предусмотрено учебным планом.

4.4. Содержание расчетно-графического задания, индивидуальных домашних заданий

Учебным планом предусмотрено одно расчётно-графическое задание, для выполнения которого предусмотрено 18 часов самостоятельной работы студента.

Цель РГЗ: разработка оконного графического Windows-приложения на ассемблере с использованием пакета masm32 и отладчика OllyDbg. Разрабатываемая программа должна решать задачу анимации процесса движения графических примитивов внутри окна с использованием API-функций ОС Windows для работы с GDI-графикой.

Типовые задания РГЗ:

1. Создать анимацию движения абсолютно упругих мячей в невесомости, которые взаимодействуют с границей окна, с другими графическими примитивами и друг с другом.
2. Создать анимацию движения упругих мячей в поле силы тяжести, параметры которого задаются в программе.

РГЗ включает в себя следующие разделы: цель задачи, описание программы в виде блок-схем, исходный код программы, результаты выполнения программы. Оценка РГЗ производится по результатам проверки пояснительной записки и работоспособности написанной программы, а также по результатам защиты, которая проходит в виде устной беседы с преподавателем.

5. ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ДЛЯ ПРОВЕДЕНИЯ ТЕКУЩЕГО КОНТРОЛЯ, ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ

5.1. Реализация компетенций

Компетенция ПК-1 Способен разрабатывать требования и проектировать программное обеспечение

Наименование индикатора достижения компетенции	Используемые средства оценивания
ПК-1.1 Анализирует требования к программному обеспечению	Защита лабораторных работ, дифференцированный зачёт
ПК-1.2 Разрабатывает технические спецификации на программные компоненты и их взаимодействие	Защита лабораторных работ, защита РГЗ
ПК-1.3 Проектирует программное обеспечение, в том числе для беспилотных авиационных систем	Защита лабораторных работ, защита РГЗ

5.2. Типовые контрольные задания для промежуточной аттестации

5.2.1. Перечень контрольных вопросов (типовых заданий) дифференцированного зачёта

№ п/п	Наименование раздела дисциплины	Содержание вопросов (типовых заданий)
1	Становление и основные тенденции развития вычислительной техники	Этапы развития вычислительной техники. ЭВМ первого поколения на базе электронно-вакуумных ламп. ЭВМ на базе транзисторов и интегральных схем. Микропроцессорные ЭВМ. Характеристики современных вычислительных систем. Закон Мура. Технологический процесс. Перспективы развития вычислительной техники и систем искусственного интеллекта.
2	Виды и классификация вычислительных систем	Классификация вычислительных систем по Флинну. Единицы для измерения производительности вычислительных систем. Пакеты программ для оценки производительности. Преимущества VLIW-архитектуры по сравнению с x86. Различия между CISC и RISC-системами. Для решения какого класса задач больше подходят системы с параллельной архитектурой? Отличия 64-разрядной и 32-разрядной архитектур. Распараллеливание вычислений с помощью графических ускорителей.
3	Архитектура Джона фон Неймана	Структурная схема классической фон Неймановской архитектуры. Основные принципы архитектуры Джона фон Неймана. Цикл выполнения команды. Назначение устройства управления и АЛУ. Порядок выборки команд из памяти. Какую информацию содержат поля **d**, **w**, **r/m**, **mod** в коде команды. В каком случае в коде команды присутствует байт **SIB** и какая в нём содержится информация?
4	Устройство процессора	Структура регистров центрального процессора. Назначение регистров ESP и EIP. Принцип работы команд CALL и RET. Использование указателя EBP для фиксирования адреса в

		стеке внутри подпрограммы. Назначение индексных регистров. Назначение флагов CF, AF, OF, ZF, SF. Написать последовательность команд, моделирующую выполнение команд PUSH, POP, PUSHAD, POPAD. Отличие команд DIV и IDIV. Почему команды SUB, ADD работают и со знаковыми и с беззнаковыми числами? Каким образом ЦП расширяет целые числа? Отличие знаковой и беззнаковой арифметики. Работа команд CMP, LOOP.
5	Организация шин	Характеристики шин USB, SATA, AGP, PCI-Express. Централизованный и децентрализованный арбитраж шин. Гнездо центрального процессора.
6	Память	Способы адресации операндов в памяти. Явная и неявная адресация операндов. Назначение стека. Выделение памяти для локальных переменных и массивов. Отличие способов выделения памяти для глобальных и локальных переменных. Очистка стека. Чем в техническом смысле отличается передача аргумента в виде void (t_item a), void (t_item* a) и void (t_item& a)? Куда помещается возвращаемое функцией значение. Задачи по программированию: 1. Определить местонахождение переменных и массивов, объявленных в программе, в сегменте данных. 2. Написать программу на ассемблере для заполнения массива длиной n кубами чисел 1^3, 2^3, 3^3,…, n^3. 3. Написать на ассемблере подпрограмму `int eval(int* a, int* b, int n)`, которая возвращает значение выражения: $$A = \sum_{i=1}^{n} \frac{a_i}{5} + ib_i.$$ 4. Написать на ассемблере подпрограмму `void output(double* a, int n)` для вывода на экран массива вещественных чисел в виде: a[1] = …; a[2] = …;…;a[n] = …; 5. С использованием команд сопроцессора написать подпрограмму `res(int n)`, возвращающую значение выражения: $$R = \sum_{k=1}^{n} \left(\sin^2\left(\frac{1}{k}\right) + \cos^2\left(\frac{1}{2k}\right) - 1 \right).$$

7	Программирование на ассемблере	Соглашения о вызовах. Декорирование названий подпрограмм. Стили вызова stdcall, cdecl, fastcall, pascal. Отличие динамического и статического способа подключения dll-библиотеки.
		1. Написать подпрограмму count в стиле fastcall, которая возвращает количество цифр в восьмеричном представлении числа n. В основной программе ввести число n с клавиатуры и вывести результат. Глобальные переменные в подпрограмме count использовать не разрешается.
		…
		`.data`
		`n dd ?`

		.code
		count proc
		…
		count endp
		start:
		… ; Ввод числа n с клавиатуры
		…
		; Передать число n в качестве аргумента подпрограмме
		…
		call count
		; Вывод результата
		2. Написать на ассемблере подпрограмму в стиле thiscall для сложения и произведения комплексных чисел и подключить её к классу на языке C++.
		class complex
		{
		Double Re, Im;
		complex operator + (const complex& a);
		complex operator * (const complex& a);
		};

5.3. Типовые контрольные задания (материалы) для текущего контроля в семестре

Текущий контроль проходит в течение семестра в виде выполнения, защиты лабораторных работ и одного РГЗ. Каждая лабораторная работа проходит процедуру допуска и защиты. Работа допускается к защите в том случае, если выполнены требования к её оформлению и поставленная задача решена правильно. Положительную оценку за выполненную лабораторную работу студент получает в том случае, если он выполнил все требования, предъявляемые к лабораторной работе, и защитил её. Защита лабораторных работ проводиться в форме беседы с преподавателем. Для защиты необходимо выучить теоретический материал и выполнить задачу по программированию по теме защищаемой лабораторной работы. Оценивается уровень усвоения теоретического материала, а также качество разработанных программ и исходного кода.

Примерный перечень контрольных вопросов для защиты лабораторных работ приведен в таблице:

Тематика лабораторной работы	Контрольные вопросы
Лабораторная работа №1. Разработка программ на ассемблере. Работа с отладчиком OllyDbg и пакетом masm32	1. Из каких частей состоит программа на ассемблере? 2. Каково назначение стека? 3. Для чего предназначена программа OllyDbg? 4. Какие директивы используются для задания размеров и объявления массивов и переменных на ассемблере? 5. Какая информация хранится в регистрах EIP и ESP? 6. Какая информация отображается в окнах отладчика OllyDbg? 7. Что такое регистры и флаги центрального процессора? 8. Какие регистры процессора можно использовать как регистры общего назначения?

	9. Как объявить массив длиной 100 вещественных чисел типа float, состоящий из одних единиц? Какой будет его размер в байтах? 10. Из каких этапов состоит компиляция программ на ассемблере? 11. Как соответствуют директивы объявления переменных типам языка C++? 12. Как хранятся в памяти целые числа? 13. Как вызываются подпрограммы на ассемблере? 14. Где хранятся локальные и глобальные данные программы? 15. Примерные задачи: 15.1. С помощью отладчика узнать значения вещественных чисел, записанных в памяти по произвольному адресу. 15.2. Узнать местоположение переменных и массивов программы в сегменте данных.
Лабораторная работа №2. Структура команд процессора	1. Что такое операнд команды? 2. Какие существуют виды адресации? 3. Что такое машинная команда? Где в программе OllyDbg отображаются коды команд? 4. Какая информация хранится в поле r/m? 5. Для чего в код команды добавляется префикс? 6. Какое поле свидетельствует о том, что в команде присутствует байт SIB? 7. Какое назначение имеет регистр EIP? От чего зависит величина, на которую он изменяется при выполнении команды? 8. Примерные задачи: 8.1. Узнать с использованием отладчика структуру какой-либо конкретной инструкции процессора. 8.2. По известному коду команды узнать её мнемоническое описание и действие, которое она выполняет. 8.3. Исследовать инструкцию процессора (например, CMP, ADD, MUL или др., выданную преподавателем) и составить для неё таблицы кодирования в общем случае.
Лабораторная работа №3. Арифметические команды центрального процессора	1. Для чего применяются команды расширения? 2. С какими операндами работают команды DIV, MUL? Какие из них имеют явную адресацию, какие – неявную? 3. Чем отличается команда DIV от команды IDIV? 4. Нужно ли применять команды расширения к беззнаковым числам? 5. Как правильно расширять беззнаковые числа? 6. Почему команды сложения и вычитания могут выполнять действия как над знаковыми числами, так и над беззнаковыми числами? 7. Где у команды DIV CX размещается делимое, делитель, частное, остаток? 8. Где у команды IMUL AX, 5 размещается результат? 9. Как хранится знак целого числа? 10. Примерные задачи: 10.1. С помощью подпрограмм библиотеки msvcrt выполнить форматный вывод значений в консоль.

	10.2. С использованием псевдографики вывести в консоль таблицу с информацией, записанной в сегменте данных. 10.3. Используя системную функцию MessageBoxA, вывести на экран сообщение с информацией из сегмента данных.
Лабораторная работа №4. Команды передачи управления	1. Какие существуют способы организации циклов на ассемблере? 2. Как работают команды CALL, RET? Как они изменяют стек и регистры? 3. Как передаются аргументы в подпрограммы на ассемблере? 4. Как обратиться к аргументам в стеке без использования команды POP? 5. Как работают команды условного перехода? 6. Как передать подпрограмме аргумент типа *long*? 7. Куда помещается возвращаемое подпрограммой значение? 8. Примерные задачи: 8.1. С клавиатуры вводится положительное целое число **a**. Вывести все его простые делители. Использовать подпрограммы: simple(x) - для проверки числа **x** на простоту; write_dividers(a) - нахождения и вывода всех простых делителей числа **a**. 8.2. С клавиатуры вводится положительное целое число **a** в десятичной системе счисления и число **k** (**k** ≥ 2 и **k** ≤ 16). Нужно вывести на экран число **a** в **k**-ной системе счисления. 8.3. С клавиатуры вводится два положительных целых числи **a** и **b**. Нужно вывести множество общих цифр данных чисел в десятичной системе счисления. Например, пусть **a** = 34567, **b** = 846533. Программа должна выводить в консоль: Множество общих цифр чисел 34567 и 846533: {3, 4, 5, 6}.
Лабораторная работа №5. Команды сопроцессора	1. Какую структуру имеет стек сопроцессора? 2. Каким образом можно преобразовать с помощью сопроцессора число из одного типа в другое, например, из *double* в тип *float*? 3. Какими командами сравниваются вещественные числа? 4. По каким признакам определить, работает команда сопроцессора с вещественными числами или с целыми? 5. Как сохранить вещественное число типа *double* из стека сопроцессора в стек оперативной памяти? 6. Что такое логические и физические номера регистров сопроцессора? 7. Как вычислить с помощью команд сопроцессора $lg(a)$, 2^x? 8. Какими командами можно освободить регистры стека сопроцессора? 9. Какая информация хранится в регистрах SWR, CWR? 10. Примерные задачи: 10.1. С клавиатуры вводится массив вещественных чисел. Положительные числа перенести в начало массива. Сохранить порядок следования как положительных, так и отрицательных чисел. Написать подпрограммы для ввода массива с клавиатуры, преобразования массива, вывода массива на экран. Использовать глобальные переменные в подпрограммах не разрешается. 10.2. С клавиатуры вводится квадратная матрица вещественных чисел. Удалить главную диагональ матрицы (тем самым уменьшить размер матрицы). Написать подпрограммы для ввода матрицы с клавиатуры, преобразования матрицы, вывода матрицы

	на экран. Матрицу следует хранить в одномерном массиве по строкам. Использовать глобальные переменные в подпрограммах не разрешается. 10.3. С клавиатуры вводится массив вещественных чисел. Определить и вывести в консоль, как отсортирован данный массив: по невозрастанию, по неубыванию, не отсортирован. Написать подпрограммы для ввода массива (input), проверки массива (is_sort). Использовать глобальные переменные в подпрограммах не разрешается.
Лабораторная работа №6. Логические команды и команды сдвига	1. Чем отличается арифметический сдвиг от логического? 2. Что может быть вторым операндом команды сдвига? 3. Как работают команды расширенного сдвига? 4. Как с помощью логических команд вычислить остаток от деления на 8; как разделить целое число на 16? 5. Как работают команды циклического сдвига? 6. Примерные задачи: 6.1. Вводится число **n** и количество сдвигов **k**. Сдвинуть циклически цифры числа **n** в восьмеричном (16-ном) представлении на **k** разрядов вправо. Учесть, что **k** может быть очень большим. Основной алгоритм реализовать без циклов. Например, для **n**=1234567, **k**=3 (10, 80) ответ такой: 5671234. 6.2. Удалить каждую вторую, начиная с младших разрядов, цифру в 16-ном представлении данного натурального числа.
Лабораторная работа №7. Цепочечные команды	1. Как работает цепочечная команда, если её использовать без префикса повторения? 2. Что такое префикс повторения? 3. Какие существуют виды префиксов повторения цепочечных команд? 4. Для каких алгоритмов удобно использовать команды MOVS, STOS? 5. Изменяется ли значение регистра ECX, если использовать цепочечную команду без префикса повторения? 6. Как с помощью цепочечных команд вычислить длину строки, которая заканчивается нулевым символом? 7. Чем отличаются действия команд CMP и CMPS? 8. Примерные задачи: Написать на ассемблере полный аналог функции memset/memcpy (или другой, на выбор преподавателя) с использованием цепочечных команд. Сравнить её работу со стандартной функций crt_memset/crt_memcpy, которая есть в заголовочном файле msvcrt.inc при разных тестовых данных. Убедится, что самописная функция идентична стандартной функции.
Лабораторная работа №8. Способы вызова ассемблерных подпрограмм в языках высокого уровня	1. Для чего используются dll-библиотеки? 2. Чем отличаются статический и динамический способы подключения dll-библиотеки? 3. Что такое декорирование названий подпрограмм? 4. Какие существуют стили вызова подпрограмм? 5. Что такое соглашения о вызовах (стили вызова)? 6. Какой стиль вызова используется компиляторами C++? 7. В чём особенность стиля вызова fastcall? 8. Что такое пролог и эпилог функции?

	9. Для чего необходима инициализирующая функция dll-библиотеки? 10. Как передать в качестве аргумента подпрограмме число типа int*; типа double? 11. Что такое ассемблерная вставка? 12. По каким правилам декорируются названия подпрограмм стилей вызова cdecl, stdcall, fastcall? 13. Какой стиль вызова используется в ООП? 14. Примерные задачи: Написать в стиле thiscall на ассемблере подпрограмму для сложения/умножения/деления/вычитания комплексных чисел (или работы с какими-либо геометрическими объектами: круг, точка, прямоугольник). Скомпилировать данную подпрограмму в виде dll-библиотеки и вызвать в программе на C++.

Критерии оценки лабораторной работы: лабораторная работа считается защищённой, если студент выполнил задание к работе полностью и во время устного опроса по работе правильно ответил на заданные преподавателем дополнительные вопросы.

Критерии оценки РГЗ:

Оценка	Критерии оценивания
5	Написанная студентом программа полностью отлажена, не имеет ошибок, пояснительная записка составлена грамотно, имеются блок-схемы и спецификации основных подпрограмм, приведены результаты работы программы и тесты.
4	В написанной программе имеются незначительные ошибки-артефакты при визуализации созданной анимации. Пояснительная записка содержит незначительные ошибки.
3	Анимация имеет большое количество артефактов, т.е. программа является работоспособной, но плохо отлаженной. Пояснительная записка содержит незначительные ошибки.
2	Написанная программа является неработоспособной, пояснительная записка не соответствует предъявляемым требованиям.

5.4. Описание критериев оценивания компетенций и шкалы оценивания

При промежуточной аттестации в форме дифференцированного зачета используется следующая шкала оценивания: 2 – неудовлетворительно, 3 – удовлетворительно, 4 – хорошо, 5 – отлично.

Критериями оценивания достижений показателей являются:

Наименование показателя оценивания результата обучения по дисциплине	Критерий оценивания
Знания	Знание архитектур вычислительных систем
	Знание директив и команд ассемблера
	Объем освоенного материала
	Полнота ответов на вопросы
	Четкость изложения и интерпретации знаний

Умения	Умение решать стандартные профессиональные задачи по низкоуровневому программированию
	Умение проверять решение и анализировать результаты
Навыки	Владение навыками разработки и отладки программ на ассемблере
	Самостоятельность разработки и отладки программ на ассемблере

Оценка преподавателем выставляется интегрально с учётом всех показателей и критериев оценивания.

Оценка сформированности компетенций по показателю Знания.

Критерий	Уровень освоения и оценка			
	2	3	4	5
Знание архитектур вычислительных систем	Не знает архитектур вычислительных систем	Знает некоторые архитектуры вычислительных систем	Знает основные архитектуры вычислительных систем	Знает архитектуры вычислительных систем
Знание основных закономерностей, соотношений, принципов	Не знает основные закономерности и соотношения, принципы построения знаний	Знает основные закономерности, соотношения, принципы построения знаний	Знает основные закономерности, соотношения, принципы постро-ения знаний, их интерпретирует и использует	Знает основные закономерности, соотношения, принципы построения знаний, может самостоятельно их получить и использовать
Знание директив и команд ассемблера	Не знает директивы и команды ассемблера	Знает некоторые директивы и команды ассемблера	Знает основные директивы и команды ассемблера	Знает директивы и команды ассемблера
Полнота ответов на вопросы	Не дает ответы на большинство вопросов	Дает неполные ответы на все вопросы	Дает ответы на вопросы, но не все - полные	Дает полные, развернутые ответы на поставленные вопросы
Четкость изложения и интерпретации знаний	Излагает знания без логической последовательности	Излагает знания с нарушениями в логической последовательности	Излагает знания без нарушений в логической последователь-ности	Излагает знания в логической последовательности, самостоятельно их интерпретируя и анализируя
	Неверно излагает и интерпретирует знания	Допускает неточности в изложении и интерпретации знаний	Грамотно и по существу излагает знания	Грамотно и точно излагает знания, делает самостоятельные выводы

Оценка сформированности компетенций по показателю Умения.

Критерий	Уровень освоения и оценка			
	2	3	4	5
Умение решать стандартные профессиональные задачи по низкоуровневому программированию	Не умеет решать стандартные профессиональные задачи по низкоуровневому программированию	Допускает неточности в решении стандартных профессиональных задач по низкоуровневому программированию	Умеет решать стандартные профессиональные задачи по низкоуровневому программиро-ванию	Безошибочно решает стандартные профессиональные задачи по низкоуровневому программированию

Умение проверять решение и анализировать результаты	Не умеет проверять решение и анализировать результаты	Умеет проверять решение некоторых задач	Умеет проверять решение некоторых задач и анализировать результаты	Умеет проверять решение и анализировать результаты

Оценка сформированности компетенций по показателю Навыки.

Критерий	Уровень освоения и оценка			
	2	3	4	5
Владение навыками разработки и отладки программ на ассемблере	Не владеет навыками разработки и отладки программ на ассемблере	Не достаточно хорошо владеет навыками разработки и отладки программ на ассемблере	Владеет навыками разработки и отладки программ на ассемблере	Профессионально владеет навыками разработки и отладки программ на ассемблере
Самостоятельность разработки и отладки программ на ассемблере	Не может самостоятельно разрабатывать и отлаживать программы на ассемблере	Разрабатывает и отлаживает программы на ассемблере с посторонней помощью	При разработке и отладке программ на ассемблере иногда требуется посторонняя помощь	Самостоятельно выполняет разработку и отладку программ на ассемблере

Критерии оценки: для получения зачёта необходимо знать теоретический лекционный материал, а также выполнить и защитить все 8 лабораторных работ и РГЗ.

Критерии оценки дифференцированного зачёта:

Оценка	Критерии оценивания
5	Студент имеет целостное понимание всего изученного теоретического материала и способен на высоком уровне самостоятельно решить технические задачи, связанные с программированием на ассемблере. При написании программ способен создавать хорошо оптимизированный код с минимальным количеством логических ошибок. При получении зачёта студент правильно решил задачу по программированию и ответил на все дополнительные вопросы, заданные преподавателем.
4	При наличии некоторых незначительных пробелов в знании теоретического материала студент имеет целостное понимание всего изученного курса и способен на достаточном уровне самостоятельно решить технические задачи, связанные с программированием на ассемблере. При получении зачёта студент правильно решил задачу по программированию с непринципиальным ошибками или некачественной оптимизацией, но ответил на большинство дополнительных вопросов, заданных преподавателем.
3	Студент имеет калейдоскопические знания из всего изученного курса, т.е. при наличии отдельных сведений не имеет целостного понимания всего пройденного материала, и способен только с посторонней помощью решать задачи по программированию на ассемблере. При получении зачёта студент решил простую задачу по целочисленным командам ассемблера с незначительными ошибками. Студент ответил на дополнительные вопросы с некоторым количеством ошибок.
2	Студент не знает теоретический материал даже по отдельным разделам дисциплины и не ответил на дополнительные вопросы. При получении зачёта студент не решил даже простую задачу по программированию на ассемблере, содержащую только целочисленные команды процессора и один вложенный цикл.

6. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ И УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ

6.1. Материально-техническое обеспечение

№	Наименование специальных помещений и помещений для самостоятельной работы	Оснащенность специальных помещений и помещений для самостоятельной работы
1	Учебная аудитория для проведения лекционных занятий	Специализированная мебель. Мультимедийная установка, экран, доски
2	Учебная аудитория для проведения лабораторных занятий	Специализированная мебель. Компьютеры на базе процессоров Intel или AMD.
3	Читальный зал библиотеки для самостоятельной работы	Специализированная мебель. Компьютерная техника, подключенная к сети интернет и имеющая доступ в электронно-образовательную среду

6.2. Лицензионное и свободно распространяемое программное обеспечение

№	Перечень лицензионного программного обеспечения.	Реквизиты подтверждающего документа
1.	Microsoft Windows 10 Корпоративная	(Соглашение Microsoft Open Value Subscription V9221014 Соглашение действительно с 01.11.2020 по 31.10.2023). Договор поставки ПО № 128-21 от 30.10.2021.
2.	Microsoft Office Professional Plus 2016	(Соглашение Microsoft Open Value Subscription V9221014 Соглашение действительно с 01.11.2020 по 31.10.2023). Договор поставки ПО № 128-21 от 30.10.2021.
3.	Kaspersky Endpoint Security «Стандартный Russian Edition»	Сублицензионный договор № 102 от 24.05.2018. Срок действия лицензии до 19.08.2020 Гражданско-правовой Договор (Контракт) № 27782 «Поставка продления права пользования (лицензии) Kaspersky Endpoint Security от 03.06.2020. Срок действия лицензии 19.08.2022г.
4.	Google Chrome	Свободно распространяемое ПО согласно условиям лицензионного соглашения
5.	Среды программирования Dev C++ , CodeBlocks, Visual Studio Community Edition	Свободно распространяемое ПО согласно условиям лицензионного соглашения

6.3. Перечень учебных изданий и учебно-методических материалов

Перечень основной литературы

1. Богданов А.В. Архитектуры и топологии многопроцессорных вычислительных систем [Электронный ресурс]/ А.В. Богданов [и др.].– Электрон. текстовые данные.– М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. – 135 с. – Режим доступа: http://www.iprbookshop.ru/52189.– ЭБС «IPRbooks».
2. Осипов О.В. Организация ЭВМ и вычислительных систем: методические указания к выполнению лабораторных работ для студентов специальности 090303.65 – Информационная безопасность автоматизированных систем / сост.: О.В. Осипов. – Белгород: Изд-во БГТУ, 2015. – 115 с.
3. Организация ЭВМ и систем. Основы программирования на языке Ассемблер: методические указания к выполнению лабораторных работ для студентов направлений бакалавриата 230100 – Информатика и вычислительная техника, 231000 – Программная инженерия и специальности 090303 – Информационная безопасность автоматизированных систем / сост.: А.И. Гарибов, О.В. Осипов. – Белгород: Изд-во БГТУ, 2014. – 35 с.
4. Цилькер Б.Я., Орлов С.А. Организация ЭВМ и систем: Учебник для вузов. – СПб.: Питер, 2006. – 686 с.: ил.
5. Мищенко В.К. Архитектура высокопроизводительных вычислительных систем [Электронный ресурс]: учебное пособие/ Мищенко В.К.– Электрон. текстовые данные.– Новосибирск: Новосибирский государственный технический университет, 2013.– 40 с.– Режим доступа: http://www.iprbookshop.ru/44898.– ЭБС «IPRbooks».
6. Чекмарев Ю.В. Вычислительные системы, сети и телекоммуникации [Электронный ресурс]/ Чекмарев Ю.В.– Электрон. текстовые данные.– М.: ДМК Пресс, 2013.– 184 с.– Режим доступа: http://www.iprbookshop.ru/5083.– ЭБС «IPRbooks».
7. Аблязов Р. З. Программирование на ассемблере на платформе x86_64. Учеб.пособие / Р. З. Аблязов. – Москва: ДМК Пресс, 2011. – 305 с.
8. Калашников О.А. Ассемблер – это просто. Учимся программировать. – 2-е изд., перераб. и доп. – СПб.: БХВ-Петербург, 2011 – 336 с.: ил.
9. Юров В.И. Assembler. Учебник для вузов. 2-е изд. – СПб.: Питер, 2006.

Перечень дополнительной литературы

1. Каган Б. М. Электронные вычислительные машины и системы. М.: Энергоатомиздат, 1991.
2. Таненбаум, Э. Архитектура компьютера / Э. Таненбаум. – 4-е изд. – СПб.: Питер, 2003. – 698 с. – (Классика computer science). – ISBN 5-318-00298-6.
3. Пирогов, В. Ю. Ассемблер и дизассемблирование / В. Ю. Пирогов. – СПб. : БХВ-Петербург, 2006. – 447 с. + 1 эл. опт. диск (CD-ROM). – ISBN 5-94157-677-3.
4. Дашевский Л. Н., Шкабара Е. А. Как это начиналось (Воспоминания о создании первой отечественной электронной вычислительной машины –

МЭСМ). – М.: «Знание», 1981. – 64 с. (Новое в жизни, науке, технике. Сер. «Математика, кибернетика», № 1).

5. Шандаров, Е.С. Архитектура вычислительных систем. Компьютерный лабораторный практикум. [Электронный ресурс] : Учебные пособия – Электрон. дан. – М. : ТУСУР, 2012. – 44 с. – Режим доступа: http://e.lanbook.com/book/11261.

Справочная и нормативная литература

1. ГОСТ 27201-87 Машины вычислительные электронные персональные. Типы, основные параметры, общие технические требования.

2. ГОСТ 2.708-81 Единая система конструкторской документации. Правила выполнения электрических схем цифровой вычислительной техники.

3. ГОСТ 25123-82 Машины вычислительные и системы обработки данных. Техническое задание. Порядок построения, изложения и оформления.

4. ГОСТ Р МЭК 62623-2015 Компьютеры настольные и ноутбуки. Измерение потребления энергии.

5. ГОСТ 25861-83 Машины вычислительные и системы обработки данных. Требования по электрической и механической безопасности и методы испытаний.

6. ГОСТ 24750-81 Средства технические вычислительной техники. Общие требования технической эстетики.

7. ГОСТ 23336-78 Машины вычислительные аналоговые и аналого-цифровые. Правила выполнения схем моделирования.

Перечень интернет ресурсов

1. http://prog-cpp.ru/asm/
2. http://www.club155.ru/x86cmd
3. http://asmworld.ru/
4. http://www.i-assembler.ru/
5. https://ru.wikipedia.org/wiki/Соглашение_о_вызове
6. http://natalia.appmat.ru/c&c++/dll.html
7. http://www.programmersclub.ru/assembler

6.4. Перечень интернет ресурсов, профессиональных баз данных, информационно-справочных систем

1. Электронная библиотека (на базе ЭБС «БиблиоТех») — Режим доступа: http://ntb.bstu.ru

2. Электронно-библиотечная система IPRbooks — Режим доступа: http://www.iprbookshop.ru

3. Электронно-библиотечная система «Университетская библиотека ONLINE» — Режим доступа: http://www.biblioclub.ru/