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 энергетики, информационных технологий и управляющих систем
_____ Белоусов А.В.
«20» сентября 2021 г.

РАБОЧАЯ ПРОГРАММА
дисциплины

Архитектура вычислительных систем

направление подготовки:

10.05.03 Информационная безопасность автоматизированных систем

Специализация программы:

Безопасность открытых информационных систем

Квалификация

Специалист по защите информации

Форма обучения

очная

Институт энергетики, информационных технологий и управляющих систем

Кафедра Программного обеспечения вычислительной техники и автоматизированных систем

Белгород 2021

Рабочая программа составлена на основании требований:

- Федерального государственного образовательного стандарта высшего образования – специалитет по специальности 10.05.03 Информационная безопасность автоматизированных систем, утвержденного приказом Минобрнауки России от 26.11.2020 №1457
- учебного плана, утвержденного ученым советом БГТУ им. В.Г. Шухова в 2021 году.

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

Рабочая программа обсуждена на заседании кафедры

«14» 05 2021 г., протокол № 8

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

«14» 05 2021 г.

Рабочая программа одобрена методической комиссией института

«20» 05 2021 г., протокол № 9

Председатель к.т.н., доцент (Семернин А.Н.)
(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
Общепрофес-сиональные компетенции	ОПК-12. Способен применять знания в области безопасности вычислительных сетей, операционных систем и баз данных при разработке автоматизированных систем	ОПК-12.1. Применяет знания в области безопасности вычислительных сетей при разработке автоматизированных систем	В результате освоения дисциплины обучающийся должен **Знать:** – виды и классификацию архитектур современных вычислительных систем; – основные принципы построения вычислительных систем для решения задач различного рода; – принципы функционирования современных многопроцессорных ЭВМ на базе архитектуры x86; – логику работы центрального процессора, сопроцессора, графического процессора, устройств ввода/вывода, внешней и оперативной памяти; – принципы взаимодействия узлов вычислительной системы посредством общей шины, основные виды шин; – основные технические характеристики и показатели быстродействия включённых в состав ЭВМ устройств; **Уметь:** – оценивать технические возможности (быстродействие, производительность и др.) вычислительных систем и их отдельных компонентов; – поддерживать работоспособность вычислительной системы и восстанавливать её после аппаратного сбоя; – собирать вычислительную систему и производить замену её отдельных элементов; **Владеть:** – навыками работы с программным обеспечением для замера производительности и технического состояния вычислительной системы; навыками настройки и конфигурирования ЭВМ.
		ОПК-12.2. Применяет знания в области безопасности операционных систем при разработке	В результате освоения дисциплины обучающийся должен **Знать:** – структуру команд центрального процессора и сопроцессора; – виды адресации данных в оперативной памяти, принципы преобразования виртуальных адресов в физические;

Категория (группа) компетенций	Код и наименование компетенции	Код и наименование индикатора достижения компетенции	Наименование показателя оценивания результата обучения по дисциплине
		автоматизированных систем	– принципы оптимизации и отладки программного кода; – структуру программы в оперативной памяти ЭВМ; – способы соединения программного кода на ассемблере с программами, написанными на языках высокого уровня; – структуру и принципы формирования программного кода компиляторами языков структурного и объектно-ориентированного программирования; – техники самых распространённых атак на процессы ОС семейства Windows; – структуру исполняемых файлов и их представление в оперативной памяти при запуске; – основные принципы работы антивирусного программного обеспечения; – основные принципы обфускации данных и системных вызовов; **Уметь:** – анализировать содержимое произвольного участка оперативной памяти, определять местонахождение команд и данных (объектов, переменных, массивов и т.д.); – анализировать машинный код и оценивать его качество с точки зрения оптимальности; – выявлять угрозы безопасности и идентифицировать заражение операционной системы; **Владеть:** – навыками разработки программного обеспечения, в том числе dll-библиотек, на ассемблере с использованием пакета masm32; – навыками создания 64-разрядного ассемблерного кода в среде Microsoft Visual Studio; – навыками дизассемблирования, отладки и оптимизации низкоуровневого программного кода с использованием отладчика x32dbg; навыками соединения ассемблерного кода с программами на языках C++, C# в среде Microsoft Visual Studio.

2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

Компетенция ОПК-12. Способен применять знания в области безопасности вычислительных сетей, операционных систем и баз данных при разработке автоматизированных систем.

Данная компетенция формируется следующими дисциплинами:

Стадия	Наименования дисциплины
1.	Архитектура вычислительных систем
2.	Операционные системы
3.	Базы данных
4.	Сети и системы передачи информации
5.	Безопасность операционных систем
6.	Безопасность систем баз данных

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 трудоёмкость дисциплины, час	144	144
Контактная работа (аудиторные занятия), в т.ч.:	70	70
лекции	17	17
лабораторные	34	34
практические	17	17
групповые консультации в период теоретического обучения и промежуточной аттестации	2	2
контроль самостоятельной работы	-	-
Самостоятельная работа студентов, включая индивидуальные и групповые консультации, в том числе:	74	74
Курсовой проект	–	–
Курсовая работа	–	–
Расчётно-графическое задание	18	18
Индивидуальное домашнее задание	–	–
Самостоятельная работа на подготовку к аудиторным занятиям (лекции, практические занятия, лабораторные занятия)	56	56
Форма промежуточная аттестация	дифф. зачёт	дифф. зачёт

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 занятиям
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.	2	2	2	5
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.	2	2	2	5
3.	Архитектура Джона фон Неймана				
	Основные принципы фон Неймановской архитектуры. Тактовый генератор. Цикл выполнения команды. Представление команд в памяти. Принцип хранимой в памяти программы.	2	2	2	6
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.	2	2	10	20
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.	2	2	4	6
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.	3	3	4	9
7.	Программирование на ассемблере				
	Структура ассемблерной программы. Компиляция и отладка ассемблерных программ с использованием пакета masm32 и отладчика x32dbg. Создание 64-разрядных ассемблерных подпрограмм в среде Microsoft Visual Studio. Представление в сегменте данных чисел, строк, массивов. Директивы объявления данных. Назначение стека. Вызов подпрограмм. Стек вызовов. Устройство сопроцессора. Стили вызова stdcall, cdecl, fastcall, thiscall, pascal. Разработка dll-библиотек на ассемблере. Статический и динамический способы подключения dll-библиотек.	4	4	10	23
	ВСЕГО	17	17	34	74

4.2. Содержание практических (семинарских) занятий

№ п/п	Наименование раздела дисциплины	Тема практического (семинарского) занятия	Кол-во часов	Самостоятельная работа на подготовку к практическим занятиям
семестр №5				
1	Устройство процессора	Структура машинной команды	2	2
2	Программирование на ассемблере	Сложение, вычитание длинных чисел на ассемблере	2	2
3	Программирование на ассемблере	Умножение 8-байтовых целых чисел типа long long на ассемблере	3	3
4	Память	Вызов системных API-функций на ассемблере	2	2
5	Память	Декомпиляция машинного кода	2	2
6	Память	Структура PE-файлов	2	2
7	Память	Обфусцирующий компилятор LLVMO	2	2
8	Программирование на ассемблере	Установка ловушек на системные вызовы	2	2
		ВСЕГО:	17	17

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 x32dbg 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	4
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	3
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	4
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-	4	4

		ориентированного кода. Стиль вызова thiscall. Таблица виртуальных методов. Стиль вызова fastcall в 64-разрядном компиляторе языка C++. Карта памяти процесса. Стек вызовов.		
		ИТОГО:	34	34

4.4. Содержание курсового проекта/работы

Выполнение курсового проекта/работы не предусмотрено учебным планом.

4.5. Содержание расчетно-графического задания, индивидуальных домашних заданий

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

Цель РГЗ: разработка приложения на ассемблере с использованием пакета masm32, среды Microsoft Visual Studio и языка C++, отладчика x32dbg (или x64dbg). Разрабатываемая программа должна решать поставленную задачу с использованием API-функций ОС Windows.

Типовые задания РГЗ:

1. Создать на ассемблере Windows-приложение и реализовать в нём анимацию движения абсолютно упругих мячей в невесомости, которые взаимодействуют с границей окна, с другими графическими примитивами и друг с другом.
2. Создать на ассемблере Windows-приложение и реализовать в нём анимацию движения упругих мячей в поле силы тяжести, параметры которого задаются в программе.
3. Вывести стек вызовов программы с подробной информацией: адрес возврата, название функции, название модуля.
4. Построить карту памяти процесса в ОС Windows.
5. Собрать подробную информацию о PE-файле.

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 ОПК-12. Способен применять знания в области безопасности вычислительных сетей, операционных систем и баз данных при разработке автоматизированных систем.

Наименование индикатора достижения компетенции	Используемые средства оценивания
ОПК-12.1. Применяет знания в области безопасности вычислительных сетей при разработке автоматизированных систем	Защита лабораторных работ, дифференцированный зачёт
ОПК-12.2. Применяет знания в области безопасности операционных систем при разработке автоматизированных систем	Защита лабораторных работ, защита РГЗ
ОПК-12.3. Применяет знания в области безопасности баз данных при разработке автоматизированных систем	Защита лабораторных работ, защита РГЗ, дифференцированный зачёт

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);`
		`};`

5.2.2. Перечень контрольных материалов для защиты курсового проекта/ курсовой работы

Выполнение курсового проекта/курсовой работы учебным планом не предусмотрено.

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 x32dbg и пакетом masm32	1. Из каких частей состоит программа на ассемблере? 2. Каково назначение стека? 3. Для чего предназначена программа x32dbg? 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 x32dbg? 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	1. Что такое операнд команды? 2. Какие существуют виды адресации?

процессора	3. Что такое машинная команда? Где в программе x32dbg 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 и прокомментирована, не имеет ошибок. Пояснительная записка составлена грамотно, имеются блок-схемы и спецификации основных подпрограмм, приведены результаты работы программы и тесты.
4	Исходный код программы плохо прокомментирован. В написанной программе имеются несущественные ошибки. Пояснительная записка содержит незначительные ошибки.
3	Программа является работоспособной, но плохо отлаженной или не прокомментированной. 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 терминов, определений, понятий теории вычислительных систем
	Знание основных закономерностей, соотношений, принципов проектирования архитектуры вычислительных систем
	Объём освоенного материала
	Полнота ответов на вопросы
	Чёткость изложения и интерпретации знаний
Умения	Умение решать стандартные профессиональные задачи с применением методов теории вычислительных систем
	Умение использовать теоретические знания для выбора методики решения профессиональных задач
Навыки	Владение навыками решения задач низкоуровневого программирования
	Качество проектирования архитектуры вычислительных систем
	Самостоятельность решения задач низкоуровневого программирования

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 терминов, определений, понятий теории вычислительных систем	Не знает терминов и определений теории архитектуры вычислительных систем	Знает термины и определения теории архитектуры вычислительных систем, но допускает	Знает термины и определения теории архитектуры вычислительных систем	Знает термины и определения теории архитектуры вычислительных систем, может корректно

		неточности формулировок		сформулировать их самостоятельно
Знание основных закономерностей, соотношений, принципов проектирования архитектуры вычислительных систем	Не знает основные закономерности и соотношения, принципы проектирования архитектуры вычислительных систем	Знает основные закономерности, соотношения, принципы проектирования архитектуры вычислительных систем	Знает основные закономерности, соотношения, принципы проектирования архитектуры вычислительных систем, интерпретирует их и использует	Знает основные закономерности, соотношения, принципы проектирования архитектуры вычислительных систем, может самостоятельно их воспроизвести и использовать
Объём освоенного материала	Не знает значительной части материала дисциплины	Знает только основной материал дисциплины, не усвоил его деталей	Знает материал дисциплины в достаточном объёме	Обладает твёрдым и полным знанием материала дисциплины, владеет дополнительными знаниями
Полнота ответов на вопросы	Не даёт ответы на большинство вопросов	Даёт неполные ответы на все вопросы	Даёт ответы на вопросы, но не все из них полные	Даёт полные, развёрнутые ответы на поставленные вопросы
Чёткость изложения и интерпретации знаний	Излагает знания без логической последовательности	Излагает знания с нарушениями в логической последовательности	Излагает знания без нарушений в логической последовательности	Излагает знания в логической последовательности, самостоятельно их интерпретируя и анализируя
	Не иллюстрирует изложение поясняющими схемами, рисунками и примерами	Выполняет поясняющие схемы и рисунки небрежно и с ошибками	Выполняет поясняющие рисунки и схемы корректно и понятно	Выполняет поясняющие рисунки и схемы точно и аккуратно, раскрывая полноту усвоенных знаний
	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 с применением методов теории вычислительных систем	Не умеет решать стандартные профессиональные задачи с применением методов теории вычислительных систем	Допускает неточности в решении стандартных профессиональных задач с применением методов теории вычислительных систем	Умеет решать стандартные профессиональные задачи с применением методов теории вычислительных систем	Безошибочно решает стандартные профессиональные задачи с применением методов теории вычислительных систем
Умение использовать теоретические знания для выбора методики решения профессиональных задач	Не умеет использовать теоретические знания для выбора методики решения профессиональных задач	Использование теоретических знаний для выбора методики решения профессиональных задач вызывает затруднения	Умеет использовать теоретические знания для выбора методики решения профессиональных задач	Умело использует теоретические знания для выбора методики решения профессиональных задач

Оценка сформированности компетенций по показателю Навыки.

Критерий	Уровень освоения и оценка			
	2	3	4	5
Владение навыками решения задач низкоуровневого программирования	Не владеет навыками решения задач низкоуровневого программирования	Недостаточно хорошо владеет навыками решения задач низкоуровневого программирования	Владеет навыками решения задач низкоуровневого программирования	Профессионально владеет навыками низкоуровневого программирования
Качество проектирования архитектуры вычислительных систем	Некачественно выполняет проектирование архитектуры вычислительных систем	Недостаточно качественно выполняет проектирование архитектуры вычислительных систем, допускает и исправляет ошибки с посторонней помощью	Недостаточно качественно выполняет проектирование архитектуры вычислительных систем, допускает и исправляет ошибки самостоятельно	Качественно выполняет проектирование архитектуры вычислительных систем
Самостоятельность решения задач низкоуровневого программирования	Не может самостоятельно решать задачи низкоуровневого программирования	Решает задачи низкоуровневого программирования с посторонней помощью	При решении задач низкоуровневого программирования иногда требуется посторонняя помощь	Самостоятельно решает задачи низкоуровневого программирования

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	Среды программирования Free Pascal, Dev C++ или CodeBlocks	Свободно распространяемое ПО согласно условиям лицензионного соглашения

6.3. Перечень учебных изданий и учебно-методических материалов

Перечень основной литературы

1. Цилькер Б.Я., Орлов С.А. Организация ЭВМ и систем: Учебник для вузов. – СПб.: Питер, 2006. – 686 с.: ил.
2. Юров В.И. Assembler. Учебник для вузов. 2-е изд. – СПб.: Питер, 2006.
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. 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».
5. 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».
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. 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.

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/

7. УТВЕРЖДЕНИЕ РАБОЧЕЙ ПРОГРАММЫ

Рабочая программа утверждена на 2022/2023 учебный год без изменений / с изменениями, дополнениями

Протокол № _____ заседания кафедры от «____»_____ 20___ г.

Заведующий кафедрой_____
подпись, ФИО

Директор института _____
подпись, ФИО