

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение высшего
образования
"САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ"

Кафедра № 13

УТВЕРЖДАЮ

Руководитель образовательной программы

доц. к.т.н.

(должность, уч. степень, звание)

Н.А. Овчинникова

(инициалы, фамилия)

(подпись)

«18» февраля 2025 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

«Введение в информационные технологии»

(Наименование дисциплины)

Код направления подготовки/ специальности	24.05.06
Наименование направления подготовки/ специальности	Системы управления летательными аппаратами
Наименование направленности	Приборы систем управления летательных аппаратов
Форма обучения	очная
Год приема	2025

Лист согласования рабочей программы дисциплины

Программу составил (а)

Старший преподаватель

(должность, уч. степень, звание)

(подпись, дата)

Н.И. Ускова

(инициалы, фамилия)

Программа одобрена на заседании кафедры № 13

«18» февраля 2025 г. протокол № 7

Заведующий кафедрой № 13

к.т.н.

(уч. степень, звание)

(подпись, дата)

Н.А. Овчинникова

(инициалы, фамилия)

Заместитель директора института №1 по методической работе

доц. к.т.н.

(должность, уч. степень, звание)

(подпись, дата)

В.Е. Таратун

(инициалы, фамилия)

Аннотация

Дисциплина «Введение в информационные технологии» входит в образовательную программу высшего образования – программу специалитета по направлению подготовки/ специальности 24.05.06 «Системы управления летательными аппаратами» направленности «Приборы систем управления летательных аппаратов». Дисциплина реализуется кафедрой «№13».

Дисциплина нацелена на формирование у выпускника следующих компетенций:

УК-2 «Способен управлять проектом на всех этапах его жизненного цикла»

ОПК-2 «Способен понимать принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности»

ОПК-9 «Способен разрабатывать алгоритмы и компьютерные программы, пригодные для практического применения»

Содержание дисциплины охватывает круг вопросов, связанных с изучением основ программирования и основных концепций компьютерных наук, технологического процесса создания компонент программного обеспечения, удовлетворяющих современным требованиям к программному продукту.

Преподавание дисциплины предусматривает следующие формы организации учебного процесса: лекции, лабораторные работы, практические занятия, самостоятельная работа обучающегося, курсовое проектирование.

Программой дисциплины предусмотрены следующие виды контроля: текущий контроль успеваемости, промежуточная аттестация в форме дифференцированного зачета.

Общая трудоемкость освоения дисциплины составляет 3 зачетных единицы, 108 часов.

Язык обучения по дисциплине «русский»

1. Перечень планируемых результатов обучения по дисциплине

1.1. Цели преподавания дисциплины

Получение обучающимися знаний основ программирования и основных концепций компьютерных наук, умений и навыков практического программирования, реализации базовых алгоритмов на языках высокого уровня; освоение технологического процесса создания компонент программного обеспечения, удовлетворяющих современным требованиям к программному продукту.

1.2. Дисциплина входит в состав обязательной части образовательной программы высшего образования (далее – ОП ВО).

1.3. Перечень планируемых результатов обучения по дисциплине, соотнесенных с планируемыми результатами освоения ОП ВО.

В результате изучения дисциплины обучающийся должен обладать следующими компетенциями или их частями. Компетенции и индикаторы их достижения приведены в таблице 1.

Таблица 1 – Перечень компетенций и индикаторов их достижения

Категория (группа) компетенции	Код и наименование компетенции	Код и наименование индикатора достижения компетенции
Универсальные компетенции	УК-2 Способен управлять проектом на всех этапах его жизненного цикла	УК-2.3.2 знать цифровые инструменты, предназначенные для разработки проекта/решения задачи; методы и программные средства управления проектами УК-2.В.2 владеть навыками решения профессиональных задач в условиях цифровизации общества
Общепрофессиональные компетенции	ОПК-2 Способен понимать принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности	ОПК-2.3.1 знать современные информационные технологии для решения типовых задач профессиональной деятельности ОПК-2.У.1 уметь применять программные средства для решения типовых задач профессиональной деятельности ОПК-2.В.1 владеть навыками работы с современными программами в области компьютерной математики
Общепрофессиональные компетенции	ОПК-9 Способен разрабатывать алгоритмы и компьютерные программы, пригодные для практического применения	ОПК-9.3.1 знать языки и платформы программирования для решения задач в профессиональной деятельности на основе компьютерных технологий ОПК-9.У.1 уметь составлять алгоритмы и компьютерные программы для решения типовых задач профессиональной деятельности ОПК-9.В.1 владеть навыками отладки, верификации и применения программ, в том числе разработанных с использованием современных интеллектуальных технологий для решения задач в профессиональной деятельности

2. Место дисциплины в структуре ОП

Дисциплина может базироваться на знаниях, ранее приобретенных обучающимися при изучении следующих дисциплин:

- «Информатика»,
- «Математика»,
- «Физика».

Знания, полученные при изучении материала данной дисциплины, имеют как самостоятельное значение, так и используются при изучении других дисциплин:

- «Информатика. Основы информационной безопасности»,
- «Учебная ознакомительная практика»,
- «Основы теории вероятностей и математическая статистика»,
- «Моделирование систем и процессов»,
- «Авиационные приборы и информационно-измерительные системы»,
- «Автоматика и управление»

3. Объем и трудоемкость дисциплины

Данные об общем объеме дисциплины, трудоемкости отдельных видов учебной работы по дисциплине (и распределение этой трудоемкости по семестрам) представлены в таблице 2.

Таблица 2 – Объем и трудоемкость дисциплины

Вид учебной работы	Всего	Трудоемкость по семестрам
		№3
1	2	3
Общая трудоемкость дисциплины, ЗЕ/ (час)	3/ 108	3/ 108
Из них часов практической подготовки		
Аудиторные занятия, всего час.	68	68
в том числе:		
лекции (Л), (час)	17	17
практические/семинарские занятия (ПЗ), (час)	17	17
лабораторные работы (ЛР), (час)	17	17
курсовой проект (работа) (КП, КР), (час)	17	17
экзамен, (час)		
Самостоятельная работа, всего (час)	40	40
Вид промежуточной аттестации: зачет, дифф. зачет, экзамен (Зачет, Дифф. зач, Экз.**)	Дифф. Зач.	Дифф. Зач.

Примечание: ** кандидатский экзамен

4. Содержание дисциплины

4.1. Распределение трудоемкости дисциплины по разделам и видам занятий.

Разделы, темы дисциплины и их трудоемкость приведены в таблице 3.

Таблица 3 – Разделы, темы дисциплины, их трудоемкость

Разделы, темы дисциплины	Лекции (час)	ПЗ (СЗ) (час)	ЛР (час)	КП (час)	СРС (час)
--------------------------	--------------	---------------	----------	----------	-----------

Семестр 3					
Раздел 1. Системы счисления. Алгоритмы. Визуальный язык Scratch и программы на нём.	1	1			
Раздел 2. Основные команды Linux. Языки C, C++, Java, Python, синтаксис.	3	3	13		4
Раздел 3. Что такое криптография? Простые криптографические шифры. Баги. C/C++: строки и массивы.	1	2			4
Раздел 4. Алгоритмы сортировки. Компилятор. Побитовые операции.	1	1	2		2
Раздел 5. Рекурсия. Стек памяти и локальные переменные. Кодирование изображения. Структуры (struct). Основы адресной арифметики.	2	2			4
Раздел 6. Указатели, структура памяти, стек, очереди и связанные списки.	1				4
Раздел 7. Веб-программирование. HTML, CSS, протокол TCP/IP и HTTP.	1	2			2
Раздел 8. Язык программирования PHP. Динамическая типизация	1				4
Раздел 9. Шаблон MVC. Введение в язык запросов SQL.	1				4
Раздел 10. Глобальное информационное поле. Искусственный интеллект.	3	2			2
Раздел 11. Языки применяемые в разработке систем управления летательными аппаратами.	1				4
Раздел 12. Процедурные языки программирования. Математические пакеты.	1	4	2		
Выполнение курсовой работы				17	6
Итого в семестре:	17	17	17	17	40
Итого	17	17	17	17	40

Практическая подготовка заключается в непосредственном выполнении обучающимися определенных трудовых функций, связанных с будущей профессиональной деятельностью.

4.2. Содержание разделов и тем лекционных занятий.

Содержание разделов и тем лекционных занятий приведено в таблице 4.

Таблица 4 – Содержание разделов и тем лекционного цикла

Номер раздела	Название и содержание разделов и тем лекционных занятий
1.	Введение. Бинарная система счисления. Как она связана с десятичной, почему именно она применяется в

	<p>компьютерах и причём здесь транзисторы. Что такое алгоритм? Эффективность алгоритмов. Псевдокод. Таблицы ASCII. КОИ-8, UTF-8 и UTF-16 Ввод данных – алгоритм – вывод данных.</p> <p>Применение алгоритма «Разделяй и властвуй».</p> <p>Демонстрация на студентах в аудитории. Баги.</p> <p>Scratch: Демонстрация возможностей.</p> <p>Основные команды и структуры: переменные, массивы, циклы.</p>
2.	<p>Основные команды Linux, компиляция и запуск новосозданных программ в командной строке.</p> <p>Программа типа «Hello, world» с пониманием её синтаксиса.</p> <p>Библиотеки C/C++, для чего и как их подключать к собственным программам.</p> <p>Что такое компилятор, как он устроен в C/C++.</p> <p>Функции C/C++, аргументы и значения, void, функция main и вызов из неё других функций.</p> <p>Численные и символьные типы данных.</p> <p>Ввод и вывод данных, спецификаторы вывода, плейсхолдеры.</p> <p>Точность при использовании разных типов данных.</p> <p>Арифметические и логические операторы. Деление по модулю.</p> <p>Оператор присваивания. Условные выражения if-else.</p> <p>Переключатели switch.</p> <p>Циклы do-while, while-do, for. Бесконечные циклы. Баги и к чему они приводят.</p>
3.	<p>Поиски багов. Бесконечный цикл.</p> <p>Функциональная декомпозиция: прием для повышения читаемости кода и удобства кодирования;</p> <p>Операторные скобки: область действия переменных;</p> <p>Объявление функций до реализации: специально для C/C++;</p> <p>Строки и как с ними работать; Что такое ошибка сегментации;</p> <p>Что такое массивы, одномерные и многомерные;</p> <p>Что такое аргументы командной строки, какова их связь с элементами массивов и как их использовать непосредственно в программах;</p> <p>Криптография — что это вообще такое? Простейшие шифры.</p>
4.	<p>Сортировка и поиск. Алгоритмы сортировки.</p> <p>«Пузырьковая» сортировка, сортировка выбором, сортировка вставками. Эффективность алгоритмов, O-символика. Компиляторы и интерпретаторы. Этапы разработки программы. Побитовые операции.</p>
5.	<p>Рекурсия.</p> <p>Сумма последовательности чисел: циклический и рекурсивный алгоритмы.</p> <p>Swap: меняем местами два числа.</p> <p>Отладка программы. Указатели.</p>

	Кодирование изображения ВІТМАР. JPEG. Шестнадцатеричная система счисления. BMP. Структура Structs в Си..Строки и указатели.Выделение памяти. Основы адресной арифметики.
6.	Указатели, структура памяти, стек, очереди и связанные списки.
7.	Что такое DHCP-сервер и IP-адреса. Что такое IPv6 и IPv4. Адресация в интернете, прокси-сервер. DNS. Протокол TCP/IP. Гипертекст и HTTP: Hyper Text Transfer Protocol. Запросы HTTP. HTML и CSS
8.	Что такое встроенные инструменты Chrome и как с ними работать. Что такое веб-серверы. Теги HTML. В чём особенности языка PHP. Синтаксис PHP, сходство и различие с Си. Как можно переслать тысячу сообщений с помощью программирования.
9.	Парадигма MVC (Model-View-Controller). На какие блокиправильно делить программу или что такое MVC? Основы языка запросов SQL. Транзакции.SQL-инъекции.
10.	Глобальное информационное поле. Искусственный интеллект.
11.	Языки, применяемые в разработке систем управления летательными аппаратами.
12.	Процедурные языки программирования. Математические пакеты: Mathematica, Mathcad, MatLAB, Maple

4.3. Практические (семинарские) занятия

Темы практических занятий и их трудоемкость приведены в таблице 5.

Таблица 5 – Практические занятия и их трудоемкость

№ п/п	Темы практических занятий	Формы практических занятий	Трудоемкость, (час)	Из них практической подготовки, (час)	№ раздела дисциплины
Семестр 3					
1.	Системы счисления. Алгоритмы. Визуальный язык Scratch и программы на нём.	Решение задач Игровое проектирование	1		1
2.	Сравнение синтаксиса языков C, C++, Java, Python,.	Решение задач	3		2
3.	C/C++: строки имассивы.	Решение задач	2		3
4.	Алгоритмы сортировки	Решение задач	1		4
5.	Рекурсия.	Решение задач	2		5
6.	Веб-программирование. HTML	Решение задач	2		7

7.	Модели представления знаний	Решение задач	2		10
8.	Среда MatLab. Элементарные вычисления и визуализация	Решение задач	4		12
Всего			17		

4.4. Лабораторные занятия

Темы лабораторных занятий и их трудоемкость приведены в таблице 6.

Таблица 6 – Лабораторные занятия и их трудоемкость

№ п/п	Наименование лабораторных работ	Трудоемкость, (час)	Из них практической подготовки, (час)	№ раздела дисциплины
Семестр 3				
	Вводное занятие	1		1,2
	Создание проекта в Microsoft Visual Studio	2		4
	Линейный вычислительный процесс	2		2
	Разветвляющийся вычислительный процесс:			2
	Условный оператор	2		
	Оператор выбора	2		
	Циклический вычислительный процесс:			2
	Цикл с параметром	3		
	Цикл с условием	3		
	Элементарная Алгебра в Mathcad	1		12
	Графика в среде MatLab	1		12
Всего		17		

4.5. Курсовое проектирование/ выполнение курсовой работы

Цель курсовой работы:

Самостоятельное творческое исследование практического характера. В ходе выполнения курсовой работы приобретается опыт самостоятельной разработки программного продукта, автоматизации решения задач в различных сферах профессиональной деятельности, вырабатываются навыки составления технической документации на программную продукцию.

Примерные темы заданий на курсовую работу приведены в разделе 10 РПД.

4.6. Самостоятельная работа обучающихся

Виды самостоятельной работы и ее трудоемкость приведены в таблице 7.

Таблица 7 – Виды самостоятельной работы и ее трудоемкость

Вид самостоятельной работы	Всего, час	Семестр 3, час
1	2	3

Изучение теоретического материала дисциплины (ТО)	30	30
Курсовое проектирование (КП, КР)	6	6
Расчетно-графические задания (РГЗ)		
Выполнение реферата (Р)		
Подготовка к текущему контролю успеваемости (ТКУ)		
Домашнее задание (ДЗ)		
Контрольные работы заочников (КРЗ)		
Подготовка к промежуточной аттестации (ПА)	4	4
Всего:	40	40

5. Перечень учебно-методического обеспечения
для самостоятельной работы обучающихся по дисциплине (модулю)

Учебно-методические материалы для самостоятельной работы обучающихся указаны в п.п. 7-11.

6. Перечень печатных и электронных учебных изданий

Перечень печатных и электронных учебных изданий приведен в таблице 8.

Таблица 8– Перечень печатных и электронных учебных изданий

Шифр/ URL адрес	Библиографическая ссылка	Количество экземпляров в библиотеке (кроме электронных экземпляров)
https://elar.rsvpu.ru/bitstream/123456789/21893/1/Shireva_CCPP.pdf	Ширева, С. Н. Основы программирования на языке C/C++ [Электронный ресурс] : практикум / С. Н. Ширёва ; Рос. гос. проф.-пед. ун-т. - Екатеринбург : РГППУ, 2017. - 147 с.	
004 Ш 96	Шумова, Елена Олеговна. Объектно-ориентированное программирование : учебное пособие / Е. О. Шумова ; С.-Петерб. гос. ун-т аэрокосм. приборостроения. - Санкт-Петербург : Изд-во ГУАП, 2021. - 115 с. - Библиогр.:	5

	с. 112	
316 III 20	Шанахан, М. Технологическая сингулярность / М. Шанахан. - М. : Точка : Альпина Паблишер, 2017. - 256 с.	3
004.4 Б24	Бар, Р. Дж. Язык Ада в проектировании систем = System design with Ada / Р. Дж. Бар ; пер. с англ. : О. С. Багатурова, В. М. Храпкин, В. С. Явнилович; ред. Е. К. Масловский. - М. : Мир, 1988. - 320 с.	2
007 Н66	Нильсон, Н. Искусственный интеллект. Методы поиска решений / Н. Нильсон. - М. : Мир, 1973. - 270 с.	1

7. Перечень электронных образовательных ресурсов
информационно-телекоммуникационной сети «Интернет»

Перечень электронных образовательных ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины приведен в таблице 9.

Таблица 9 – Перечень электронных образовательных ресурсов информационно-телекоммуникационной сети «Интернет»

URL адрес	Наименование
https://javarush.ru/	CS-50
https://stepik.org/	<u>Введение в Data Science и машинное обучение</u> <u>Быстрый старт в искусственный интеллект</u> <u>Введение в программирование (C++)</u> <u>Алгоритмы: теория и практика. Методы</u> <u>Алгоритмы: теория и практика. Структуры данных</u> <u>Программирование на языке C++. МНМИЦ СПбГУ</u> <u>Программирование на языке C++ (продолжение). МНМИЦ СПбГУ</u> <u>Веб-разработка для начинающих: HTML и CSS</u> <u>Программирование на Python</u>
http://www.ada-ru.org/arm83/index.html	СПРАВОЧНОЕ РУКОВОДСТВО ПО ЯЗЫКУ АДА 83

8. Перечень информационных технологий

8.1. Перечень программного обеспечения, используемого при осуществлении образовательного процесса по дисциплине.

Перечень используемого программного обеспечения представлен в таблице 10.

Таблица 10– Перечень программного обеспечения

№ п/п	Наименование
	Не предусмотрено

8.2. Перечень информационно-справочных систем,используемых при осуществлении образовательного процесса по дисциплине

Перечень используемых информационно-справочных систем представлен в таблице 11.

Таблица 11– Перечень информационно-справочных систем

№ п/п	Наименование
	Не предусмотрено

9. Материально-техническая база

Состав материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине, представлен в таблице12.

Таблица 12 – Состав материально-технической базы

№ п/п	Наименование составной части материально-технической базы	Номер аудитории (при необходимости)
1	Мультимедийная лекционная аудитория	
2	Компьютерный класс	

10. Оценочные средства для проведения промежуточной аттестации

10.1. Состав оценочных средств для проведения промежуточной аттестации обучающихся по дисциплине приведен в таблице 13.

Таблица 13 – Состав оценочных средств для проведения промежуточной аттестации

Вид промежуточной аттестации	Перечень оценочных средств
Дифференцированный зачёт	Список вопросов; Тесты; Задачи.
Выполнение курсовой работы	Экспертная оценка на основе требований к содержанию курсовой работы по дисциплине.

10.2. В качестве критериев оценки уровня сформированности (освоения) компетенций обучающимися применяется 5-балльная шкала оценки сформированности компетенций, которая приведена в таблице 14. В течение семестра может использоваться 100-балльная шкала модульно-рейтинговой системы Университета, правила использования которой, установлены соответствующим локальным нормативным актом ГУАП.

Таблица 14 –Критерии оценки уровня сформированности компетенций

Оценка компетенции	Характеристика сформированных компетенций
5-балльная шкала	

Оценка компетенции 5-балльная шкала	Характеристика сформированных компетенций
«отлично» «зачтено»	<ul style="list-style-type: none"> – обучающийся глубоко и всесторонне усвоил программный материал; – уверенно, логично, последовательно и грамотно его излагает; – опираясь на знания основной и дополнительной литературы, тесно привязывает усвоенные научные положения с практической деятельностью направления; – умело обосновывает и аргументирует выдвигаемые им идеи; – делает выводы и обобщения; – свободно владеет системой специализированных понятий.
«хорошо» «зачтено»	<ul style="list-style-type: none"> – обучающийся твердо усвоил программный материал, грамотно и по существу излагает его, опираясь на знания основной литературы; – не допускает существенных неточностей; – увязывает усвоенные знания с практической деятельностью направления; – аргументирует научные положения; – делает выводы и обобщения; – владеет системой специализированных понятий.
«удовлетворительно» «зачтено»	<ul style="list-style-type: none"> – обучающийся усвоил только основной программный материал, по существу излагает его, опираясь на знания только основной литературы; – допускает несущественные ошибки и неточности; – испытывает затруднения в практическом применении знаний направления; – слабо аргументирует научные положения; – затрудняется в формулировании выводов и обобщений; – частично владеет системой специализированных понятий.
«неудовлетворительно» «не зачтено»	<ul style="list-style-type: none"> – обучающийся не усвоил значительной части программного материала; – допускает существенные ошибки и неточности при рассмотрении проблем в конкретном направлении; – испытывает трудности в практическом применении знаний; – не может аргументировать научные положения; – не формулирует выводов и обобщений.

10.3. Типовые контрольные задания или иные материалы.

Вопросы (задачи) для экзамена представлены в таблице 15.

Таблица 15 – Вопросы (задачи) для экзамена

№ п/п	Перечень вопросов (задач) для экзамена	Код индикатора
	Учебным планом не предусмотрено	

Вопросы (задачи) для зачета / дифф. зачета представлены в таблице 16.

Таблица 16 – Вопросы (задачи) для зачета / дифф. Зачета

№ п/п	Перечень вопросов (задач) для экзамена	Код индикатора
1. Бинарная система счисления. 2. Как она связана с десятичной, почему именно она применяется в компьютерах и причём здесь транзисторы. 3. Что такое алгоритм? 4. Эффективность алгоритмов. 5. Псевдокод. 6. Таблицы ASCII.		УК-2.3.2 УК-2.B.2 ОПК-2.3.1 ОПК-2.Y.1 ОПК-2.B.1 ОПК-9.3.1

<p>7. Основные команды Linux</p> <p>8. Библиотеки C/C++, для чего и как их подключать к собственным программам.</p> <p>9. Что такое компилятор, как он устроен в C/C++</p> <p>10. Функции C/C++, аргументы и значения, void, функция main и вызов из неё других функций.</p> <p>11. Численные и символьные типы данных.</p> <p>12. Ввод и вывод данных, спецификаторы вывода, плейсхолдеры.</p> <p>13. Точность при использовании разных типов данных.</p> <p>14. Арифметические и логические операторы. Деление по модулю.</p> <p>15. Оператор присваивания.</p> <p>16. Условные выражения if-else.</p> <p>17. Переключатели switch.</p> <p>18. Циклы do-while, while-do, for. Бесконечные циклы.</p> <p>19. Баги и к чему они приводят.</p> <p>20. Бесконечный цикл.</p> <p>21. Операторные скобки: область действия переменных;</p> <p>22. Объявление функций до реализации: специально для C/C++;</p> <p>23. Строки и как с ними работать;</p> <p>24. Что такое ошибка сегментации;</p> <p>25. Что такое массивы, одномерные и многомерные;</p> <p>26. Что такое аргументы командной строки, какова их связь с элементами массивов и как их использовать непосредственно в программах;</p> <p>27. Криптография — что это вообще такое? Простейшие шифры.</p> <p>28. Алгоритмы сортировки.</p> <p>29. Компилятор.</p> <p>30. Рекурсия. Сумма последовательности чисел: циклический и рекурсивный алгоритмы.</p> <p>31. Указатели.</p> <p>32. Кодирование изображения BITMAP. JPEG. Шестнадцатеричная система счисления. BMP.</p> <p>33. Структура Structs в Си. Что такое адресная арифметика?</p> <p>34. Что такое DHCP-сервер и IP-адреса.</p> <p>35. Что такое IPv6 и IPv4. Адресация в интернете, прокси-сервер. DNS.</p> <p>36. Протокол TCP/IP. Гипертекст и HTTP: Hyper Text Transfer Protocol. Запросы HTTP.</p> <p>37. HTML и CSS</p> <p>38. Что такое встроенные инструменты Chrome и как с ними работать.</p> <p>39. Что такое веб-серверы.</p> <p>40. Теги HTML.</p> <p>41. В чём особенности языка PHP.</p> <p>42. Синтаксис PHP, сходство и различие с Си.</p> <p>43. Парадигма MVC (Model-View-Controller). На какие блоки правильно делить программу или что такое MVC?</p> <p>44. Основы языка запросов SQL. Транзакции. SQL-инъекции.</p> <p>45. Глобальное информационное поле. Искусственный интеллект.</p> <p>46. Языки применяемые в разработке систем управления</p>	<p>ОПК-9.У.1</p> <p>ОПК-9.В.1</p>
--	-----------------------------------

<p>летательными аппаратами.</p>	
<p>Задачи:</p> <p>1. Напишите функцию power, реализующую возведение целого числа в неотрицательную целую степень. Функция power должна принимать на вход два целых числа и возвращать целое число. При выполнении задания учтите, что функция обязательно должна называться power, функция ничего не должна читать со входа или выводить.</p> <p>Требования к реализации: в этом задании вам нужно реализовать <i>только</i> функцию power. Вы можете определять вспомогательные функции, если они вам нужны. Реализовывать функции main не нужно.</p> <p>Ограничения: библиотеку cmath (и math.h) использовать запрещено.</p>	
<p>2. Напишите программу, которая суммирует целые числа. На вход программе подаются целые числа в следующем формате: на первой строке идет целое число T — количество тестов, далее следует T строк, в каждой из которых через пробел идут два целых числа a_i и b_i. На выводе для каждой из T строк нужно вывести сумму $a_i + b_i$ в том порядке, в котором пары поступают на вход. Ничего, кроме этого, выводить не нужно.</p>	
<p>3. Напишите программу для решения квадратных уравнений вида $ax^2 + bx + c = 0$ (относительно x). На вход программа получает три целых числа: a, b и c, соответственно. При этом гарантируется, что $a \neq 0$. На вывод программа должна вывести два вещественных корня уравнения, разделённые пробелом. Если вещественных корней нет, то программа должна вывести строку "No real roots". Если у уравнения имеется только один корень (кратный корень), то программа должна вывести его дважды. Порядок вывода корней не важен. Ничего, кроме этого, выводить не нужно. Для вычислений с плавающей точкой используйте тип double. При выполнении задания вам может оказаться полезной функция sqrt из заголовочного файла cmath.</p>	
<p>4. Дано натуральное число, выведите сумму его первой и последней цифры.</p> <p>Формат входных данных</p> <p>На вход дается натуральное число N, не превосходящее 10000.</p> <p>Формат выходных данных</p>	

<p>Выведите одно целое число - ответ на задачу.</p>	
<p>5. Дано целое трехзначное число. Найдите сумму его цифр.</p> <p>Формат входных данных</p> <p>На вход дается число от 100 до 999.</p> <p>Формат выходных данных</p> <p>Выведите одно целое число - ответ на задачу.</p>	
<p>6. Напишите программу, которая запрашивает пятиразрядное число и выводит цифры данного числа задом наперед.</p>	
<p>7. Напишите программу, которая запрашивает пять натуральных чисел и выводит 1, если это число нечётное и 0 -- если числочётное.</p>	
<p>8. В этой задаче необходимо было ввести два числа a и b, и вывести их в обратном порядке через пробел. Программист ошибся, и в итоге программа не работает. Найдите и исправьте ошибки в тексте программы.</p> <pre>#include <iostream>using namespace std;int main(){ int a, b; cin << a << b; cout >> a >> " " >> b >> endl;return 0; }</pre>	
<p>9. В этой задаче вводятся три числа a, b, c и должна выводиться в первой строке сумма d этих чисел, а во второй -- их произведение e. Программист, создававший код, допустил ошибку. Исправьте её.</p> <pre>#include <iostream>using namespace std;int main(){ int a, b, c, d, e; cin >> a >> b >> c; d = a + b + c; e = a * b * c; cout << d; cout << e; return 0; }</pre>	
<p>10. Напишите программу, которая переводит показания секундомера в минуты и секунды. На вход программе подаётся одно число -- показание секундомера в секундах. Программа должна вывести два числа через пробел -- количество минут и количество секунд в этом показании.</p>	

11. Напишите программу, которая запрашивает число (от 500 до 5000), и выводит, к какому веку относится год с этим номером.	
12. Напишите программу, которая запрашивает два промежутка времени в формате "часы минуты секунды", и выводит количество секунд между ними. Гарантируется, что второй промежуток времени наступил после первого, и показания взяты в одни и те же сутки.	
13. Напишите программу, которая выдаёт из банкомата запрошенную сумму денег (кратную 10) в имеющихся купюрах (100 рублей, 50 рублей, 10 рублей). Сумму необходимо выдавать по возможности наиболее крупными купюрами. Сумма не будет превышать 20000 руб.	
14. Вводится информационный объём сообщения -- X бит (кратно 8). Найдите количество Килобайт (KB) и Байт (B) в данном сообщении. В 1 Килобайте -- 1024 Байта; В 1 Байте -- 8 бит.	
15. В доме N подъездов, в каждом из них M этажей, на каждом этаже K квартир. В первой строке через пробел вводятся три числа N , M и K . Во второй строке вводится X -- номер квартиры. Определить, в каком подъезде находится эта квартира и на каком этаже. Гарантируется, что квартира с таким номером в данном доме существует.	
16. По данному целому числу N распечатайте все квадраты натуральных чисел, не превосходящие N , в порядке возрастания.	
17. По данному числу N распечатайте все целые степени двойки, не превосходящие N , в порядке возрастания.	
18. Напишите программу, которая запрашивает натуральное трёхзначное число, и если сумма его крайних цифр равна средней цифре -- пишет "YES", в противном случае -- "NO"	
19. Напишите программу, которая запрашивает натуральное число N (не более 50), и выводит два ряда из N звёздочек.	

20. Напишите программу, которая запрашивает два натуральных числа a и b , ($a \leq b$) и выводит все натуральные числа от a до b включительно через пробел.	
---	--

Перечень тем для курсового проектирования/выполнения курсовой работы представлены в таблице 17.

Таблица 17 – Перечень тем для курсового проектирования/выполнения курсовой работы

№ п/п	Примерный перечень тем для курсового проектирования/выполнения курсовой работы

Вопросы для проведения промежуточной аттестации в виде тестирования представлены в таблице 18.

Таблица 18 – Примерный перечень вопросов для тестов

№ п/п	Примерный перечень вопросов для тестов	Код индикатора
1.	<p>Выберите все верные утверждения из списка.</p> <ol style="list-style-type: none"> 1. C++ компилируемый язык программирования. 2. C++ язык с динамической типизацией. 3. C++ интерпретируемый язык программирования. 4. C++ ориентирован на быстрое создание прототипов приложений. 5. C++ язык со статической типизацией. 6. C++ не поддерживает процедурное программирование. 	<p>УК-2.3.2 УК-2.В.2 ОПК-2.3.1 ОПК-2.У.1 ОПК-2.В.1 ОПК-9.3.1 ОПК-9.У.1 ОПК-9.В.1</p>
2.	<p>Отметьте все верные утверждения. Выберите все подходящие ответы из списка</p> <ol style="list-style-type: none"> 1. Для запуска программы, код которой был написан на интерпретируемом языке, на компьютере должен быть установлен интерпретатор этого языка. 2. Код программы, написанный на языке, который компилируется в байт код виртуальной машины, достаточно скомпилировать однажды, чтобы программу можно было запускать на любой операционной системе, где есть соответствующая виртуальная машина. 3. Код программы, написанный на языке, который компилируется в машинный код, достаточно скомпилировать однажды, и потом программу можно будет запустить на любой операционной системе, для которой существует компилятор этого языка. 4. Для запуска программы, код которой был написан на компилируемом языке, на компьютере должен быть установлен компилятор этого языка. 	<p>УК-2.3.2 УК-2.В.2 ОПК-2.3.1 ОПК-2.У.1 ОПК-2.В.1 ОПК-9.3.1 ОПК-9.У.1 ОПК-9.В.1</p>

	<p>5. Код программы, написанный на интерпретируемом языке, можно без предварительной компиляции запустить на любой операционной системе, где установлен интерпретатор этого языка.</p> <p>6. Скомпилировать программу на C++ для некоторой архитектуры X можно только на компьютере с архитектурой X.</p>	
3.	<p>Выберите из списка объявления, которые не стоит помещать в заголовочные файлы</p> <p>Выберите все подходящие ответы из списка</p> <ol style="list-style-type: none"> 1. <code>void foo();</code> 2. <code>extern int a;</code> 3. <code>int a;</code> 4. <code>void foo() { std::cout << "Hello, World!\n"; }</code> 5. <code>void bar() { foo(); }</code> 	<p>УК-2.3.2 УК-2.B.2 ОПК-2.3.1 ОПК-2.Y.1 ОПК-2.B.1 ОПК-9.3.1 ОПК-9.Y.1 ОПК-9.B.1</p>
4.	<p>Предположим, что в программе определён макрос <code>sqr</code>.</p> <pre>#define sqr(x) x * x</pre> <p>Какое значение будет иметь следующее выражение?</p> <p><code>sqr(3 + 0)</code></p> <p>Введите численный ответ</p>	<p>УК-2.3.2 УК-2.B.2 ОПК-2.3.1 ОПК-2.Y.1 ОПК-2.B.1 ОПК-9.3.1 ОПК-9.Y.1 ОПК-9.B.1</p>
5.	<p>В коде программы определена следующая функция:</p> <pre>int foo(int n) { if (n <= 0) return 1; return foo((n * 2) / 3) + foo(n - 2); }</pre> <p>Нужно посчитать, сколько всего раз будет вызвана функция <code>foo</code>, если ее вызвать с аргументом 3 (т.е. <code>foo(3)</code>). Самый первый вызов тоже нужно посчитать.</p> <p>Подсказка: для решения этой задачи будет полезно нарисовать дерево рекурсивных вызовов.</p> <p>Введите численный ответ</p>	<p>УК-2.3.2 УК-2.B.2 ОПК-2.3.1 ОПК-2.Y.1 ОПК-2.B.1 ОПК-9.3.1 ОПК-9.Y.1 ОПК-9.B.1</p>

	<p>В коде определена следующая структура:</p> <pre>struct ivector3d { int array[3]; };</pre> <p>И определена следующая функция:</p> <pre>void scale(ivector3d *v, int k) { for (int i = 0; i != 3; ++i) v->array[i] *= k; }</pre> <p>Пусть у вас есть экземпляр iv3d структуры ivector3d. Изначально массив array экземпляра iv3d заполнен единицами, и вы вызываете функцию scale следующим образом:</p> <pre>scale(&iv3d, 2);</pre> <p>Какова будет сумма элементов массива array внутри iv3d по завершению функции?</p> <p>Введите численный ответ</p>	
6.	<p>структура ivector3d определена следующим образом:</p> <pre>struct ivector3d { int *array; };</pre> <p>У вас есть экземпляр iv3d этой структуры, поле array которого указывает на массив из трех элементов, заполненный единицами. Кроме того у вас есть функция scale, определенная следующим образом:</p> <pre>void scale(ivector3d v, int k) { for (int i = 0; i != 3; ++i) v.array[i] *= k; }</pre> <p>Задача та же, какова будет сумма элементов в массиве, на который указывает поле array, после следующего вызова:</p> <pre>scale(iv3d, 2);</pre> <p>Введите численный ответ</p>	<p>УК-2.3.2 УК-2.B.2 ОПК-2.3.1 ОПК-2.У.1 ОПК-2.B.1 ОПК-9.3.1 ОПК-9.У.1 ОПК-9.B.1</p>
7.	<p>Компилятору C++ позволено выполнять оптимизации, связанные с удалением ненужных вызовов конструктора копирования (copy elision) при определенных условиях, даже если конструктор копирования содержит сайд эффекты (например, вывод в std::cout), поэтому на большинстве современных компиляторов конструктор копирования не будет вызван ни в одной из указанных ситуаций. Однако, компилятор не обязан выполнять эту оптимизацию.</p> <p>В каких из приведенных ниже ситуаций может быть вызван</p>	<p>УК-2.3.2 УК-2.B.2 ОПК-2.3.1 ОПК-2.У.1 ОПК-2.B.1 ОПК-9.3.1 ОПК-9.У.1 ОПК-9.B.1</p>

	<p>конструктор копирования класса String:</p> <p>1.</p> <pre>String spaces(size_t n) { const String s(n, ' '); return s; }</pre> <pre>int main() { std::cout << spaces(10).str << "\n"; return 0; }</pre> <p>2.</p> <pre>int main() { String ten_spaces; ten_spaces = String(10, ' '); std::cout << ten_spaces.str << "\n"; return 0; }</pre> <p>3.</p> <pre>void foo(String str) { std::cout << str.str << "\n"; }</pre> <pre>int main() { foo(String(10, ' ')); return 0; }</pre> <p>4.</p> <pre>void bar(const String &str) { std::cout << str.str << "\n"; }</pre> <pre>int main() { bar(" "); // ten spaces; return 0; }</pre> <p>Выберите все подходящие ответы из списка</p> <p>1. 1</p>	
--	--	--

	<p>2. 2</p> <p>3. 3</p> <p>4. 4</p>	
8.	<p>Ниже даны несколько вариантов реализации оператора присваивания для класса String, выберите из них некорректные реализации, т.е. такие, которые неправильно работают с памятью (допускают утечки, обращаются к освобожденной памяти, используют неинициализированные указатели), нарушают семантику оператора присваивания или инвариант класса.</p> <p>1.</p> <pre>String &operator=(const String &other) { str = new char[other.size + 1]; strcpy(str, other.str); size = other.size; return *this; }</pre> <p>2.</p> <pre>String &operator=(const String &other) { delete[] str; str = new char[other.size + 1]; strcpy(str, other.str); size = other.size; return *this; }</pre> <p>3.</p> <pre>String &operator=(const String &other) { if (this != &other) { str = other.str; size = other.size; } return *this; }</pre> <p>Выберите все подходящие ответы из списка</p> <p>1. 1</p> <p>2. 2</p> <p>3. 3</p>	<p>УК-2.3.2</p> <p>УК-2.В.2</p> <p>ОПК-2.3.1</p> <p>ОПК-2.У.1</p> <p>ОПК-2.В.1</p> <p>ОПК-9.3.1</p> <p>ОПК-9.У.1</p> <p>ОПК-9.В.1</p>
9.	<p>Есть три версии функции foo:</p>	<p>УК-2.3.2</p>

	<pre>void foo(char) { std::cout << "char" << std::endl; } void foo(signed char) { std::cout << "signed char" << std::endl; } void foo(unsigned char) { std::cout << "unsigned char" << std::endl; }</pre> <p>Отметьте все верные утверждения относительно вызова функции foo.</p> <p>Выберите все подходящие ответы из списка</p> <ol style="list-style-type: none"> 1. в результате вызова foo('a') будет выведено signed char 2. вызов foo('a') приведет к ошибке компиляции 3. в результате вызова foo(97) будет выведено char 4. вызов foo(97) приведет к ошибке компиляции 5. в результате вызова foo('a') будет выведено unsigned char 6. в результате вызова foo(97) будет выведено signed char 7. в результате вызова foo('a') будет выведено char 	<p>УК-2.B.2 ОПК-2.3.1 ОПК-2.У.1 ОПК-2.B.1 ОПК-9.3.1 ОПК-9.У.1 ОПК-9.B.1</p>
10.	<p>В программе определены две функции:</p> <pre>float square(float value) { return value * value; } double square(float value) { return (double)value * value; }</pre> <p>Далее в программе есть вызов:</p> <pre>double sq = square(2.0);</pre> <p>Отметьте все верные утверждения из списка.</p> <p>Выберите все подходящие ответы из списка</p> <ol style="list-style-type: none"> 1. программа не скомпилируется, потому что такая перегрузка функции square не допустима 2. программа не скомпилируется, потому что вызов double sq = square(2.0) неоднозначен 3. программа скомпилируется, будет вызвана функция double square(float value) 4. программа скомпилируется, будет вызвана функция float square(float value) 	<p>УК-2.3.2 УК-2.B.2 ОПК-2.3.1 ОПК-2.У.1 ОПК-2.B.1 ОПК-9.3.1 ОПК-9.У.1 ОПК-9.B.1</p>
11.	<p>В программе есть следующие определения:</p> <pre>void promotion(char &) { std::cout << "char" << std::endl; } void promotion(int &) { std::cout << "int" << std::endl; } void promotion(long &) { std::cout << "long" << std::endl; }</pre> <p>Кроме того в программе есть вызов:</p> <pre>short sh = 10;</pre>	<p>УК-2.3.2 УК-2.B.2 ОПК-2.3.1 ОПК-2.У.1 ОПК-2.B.1 ОПК-9.3.1 ОПК-9.У.1 ОПК-9.B.1</p>

	<p><code>promotion(sh);</code> Отметьте все верные утверждения.</p> <p>Выберите все подходящие ответы из списка</p> <ol style="list-style-type: none"> 1. вызов <code>promotion(sh)</code> не скомпилируется, так как есть несколько подходящих функций для вызова 2. при вызове <code>promotion(sh)</code> произойдет преобразование типа и будет вызвана функция <code>promotion(char &)</code> 3. при вызове <code>promotion(sh)</code> произойдет преобразование типа и будет вызвана функция <code>promotion(int &)</code> 4. при вызове <code>promotion(sh)</code> произойдет преобразование типа и будет вызвана функция <code>promotion(long &)</code> 5. вызов <code>promotion(sh)</code> не скомпилируется, так как нет ни одной подходящей функции для вызова 	
12.	<p>Выберите все верные утверждения.</p> <p>Выберите все подходящие ответы из списка</p> <ol style="list-style-type: none"> 1. Все виртуальные методы базового класса являются виртуальными и для производных классов. 2. Невиртуальные методы не могут вызывать виртуальные методы. 3. Для того, чтобы вызвать виртуальный метод, объект должен быть создан через <code>new</code>. 4. Виртуальные методы могут вызывать неvirtуальные методы. 5. Обращение к объекту по ссылке не позволяет вызывать виртуальные методы. 6. Виртуальный метод можно вызвать только через указатель на объект. 7. Виртуальные методы могут быть константными. 	<p>УК-2.3.2 УК-2.B.2 ОПК-2.3.1 ОПК-2.Y.1 ОПК-2.B.1 ОПК-9.3.1 ОПК-9.Y.1 ОПК-9.B.1</p>
13.	<p>Выберите все верные утверждения.</p> <p>Выберите все подходящие ответы из списка</p> <ol style="list-style-type: none"> 1. Производные классы не могут переопределять <code>private</code>-виртуальные методы базового класса, если они унаследованы от базового класса с модификатором <code>private</code>. 2. У интерфейсов нет таблиц виртуальных методов. 3. Если в базовом классе виртуальная функция определена как 	<p>УК-2.3.2 УК-2.B.2 ОПК-2.3.1 ОПК-2.Y.1 ОПК-2.B.1 ОПК-9.3.1 ОПК-9.Y.1 ОПК-9.B.1</p>

	<p>public, то в производном классе её можно переопределить как private.</p> <p>4. Производные классы не видят protected-предков своего (непосредственного) базового класса, если они унаследованы от базового класса с модификатором private.</p> <p>5. Чистый виртуальный метод с определением — это не то же самое, что обычный виртуальный метод.</p> <p>6. Производные классы не видят private-предков своего базового класса.</p> <p>7. Если в базовом классе виртуальная функция определена как private, то в производном классе её можно переопределить как public.</p>	
14.	<p>В заголовочном файле count.hpp определена следующая функция:</p> <pre>static int count() { static int counter = 0; return ++counter; }</pre> <p>Этот файл подключается тремя файлами foo.cpp, bar.cpp и zoo.cpp.</p> <p>Сколько различных экземпляров переменной counter будут существовать в программе?</p> <p>Введите численный ответ</p>	<p>УК-2.3.2 УК-2.B.2 ОПК-2.3.1 ОПК-2.Y.1 ОПК-2.B.1 ОПК-9.3.1 ОПК-9.Y.1 ОПК-9.B.1</p>
15.	<p>В заголовочном файле foo.hpp есть определение функции:</p> <pre>inline void foo(int i) { std::cout << "i = " << i << std::endl; }</pre> <p>В программе есть три корректных файла с кодом first.cpp, second.cpp и third.cpp, которые подключают foo.hpp.</p> <p>Отметьте все верные утверждения из списка.</p> <p>Выберите все подходящие ответы из списка</p> <ol style="list-style-type: none"> 1) При компоновке лишние определения функции foo будут отброшены. 2) Программа компилируется и компоуется без проблем. 3) Файлы first.cpp, second.cpp и third.cpp компилируются без проблем. 4) Файлы first.cpp, second.cpp и third.cpp компилируются без проблем, но на этапе компоновки возникает ошибка из-за множественного определения функции foo. 	<p>УК-2.3.2 УК-2.B.2 ОПК-2.3.1 ОПК-2.Y.1 ОПК-2.B.1 ОПК-9.3.1 ОПК-9.Y.1 ОПК-9.B.1</p>
16.	<p>В заголовочном файле count.hpp определена следующая функция:</p>	<p>УК-2.3.2 УК-2.B.2</p>

	<pre>inline int count() { static int counter = 0; return ++counter; }</pre> <p>Этот файл подключается тремя файлами foo.cpp, bar.cpp и zoo.cpp.</p> <p>Сколько различных экземпляров переменной counter будут существовать в программе?</p> <p>Введите численный ответ</p>	ОПК-2.3.1 ОПК-2.У.1 ОПК-2.В.1 ОПК-9.3.1 ОПК-9.У.1 ОПК-9.В.1
17.	<p>Давайте вспомним как решается линейное уравнение $ax+b=0$.. Будем считать, что $a \neq 0$. На следующем шаге необходимо выбрать правильный вариант формулы.</p> <p>Выберите один вариант из списка</p> <p>1) $x = a / b$</p> <p>2) $x = b / a$</p> <p>3) $x = - a / b$</p> <p>4) $x = - b / a$</p>	УК-2.3.2 УК-2.В.2 ОПК-2.3.1 ОПК-2.У.1 ОПК-2.В.1 ОПК-9.3.1 ОПК-9.У.1 ОПК-9.В.1
18.	<p>Используя предыдущую программу нахождения корня линейного уравнения как образец, создайте программу нахождения площади прямоугольника.</p> <p>Нарисуйте блок-схему (выполняется в случае очного обучения).</p> <p>Изображение блок-схемы можно загрузить с помощью пункта "Insert Image".</p> <p>При любом ответе система выдает положительную оценку, но качество решения оценивает только преподаватель!</p> <p>Напишите развернутый ответ</p>	УК-2.3.2 УК-2.В.2 ОПК-2.3.1 ОПК-2.У.1 ОПК-2.В.1 ОПК-9.3.1 ОПК-9.У.1 ОПК-9.В.1
19.	<p>Создайте программу решения квадратного уравнения (считать, что дискриминант D всегда больше 0). Нарисуйте блок-схему (выполняется в случае очного обучения).</p> <p>Изображение блок-схемы можно загрузить с помощью пункта "Insert Image".</p> <p>При любом ответе система выдает положительную оценку, но качество решения оценивает только преподаватель!</p> <p>Напишите развернутый ответ</p>	УК-2.3.2 УК-2.В.2 ОПК-2.3.1 ОПК-2.У.1 ОПК-2.В.1 ОПК-9.3.1 ОПК-9.У.1 ОПК-9.В.1
20.	<p>Можно ли угадать задуманное целое число от 1 до 1000, задав только 500 вопросов? (Разрешаются вопросы с ответом "да" или "нет".)</p> <p>Выберите один вариант из списка</p> <p>1) да</p>	УК-2.3.2 УК-2.В.2 ОПК-2.3.1 ОПК-2.У.1 ОПК-2.В.1 ОПК-9.3.1 ОПК-9.У.1

	2) нет	ОПК-9.В.1
21.	<p>Слева указаны формулы, записанные с помощью логических операций, а справа — записанные с помощью арифметических операций по модулю 2 (значения 0 и 1, умножение обычное, а сложение по модулю 2, то есть $1+1=0$). Составьте пары из формул, задающих одну и ту же функцию (в разных обозначениях).</p> <p>Сопоставьте значения из двух списков</p> <p>1) A or B 2) A and B 3) not (A \Rightarrow B) 4) A xor B</p> <p>a) $A + B - A*B$ b) $A * B$ c) $A - A*B$ d) $(A + B) \bmod 2$</p>	УК-2.3.2 УК-2.В.2 ОПК-2.3.1 ОПК-2.У.1 ОПК-2.В.1 ОПК-9.3.1 ОПК-9.У.1 ОПК-9.В.1
22.	<p>окажите, что разные логические связки можно выразить как многочлены, если считать истину единицей, а ложь нулём. (Отсюда, зная о возможности представления любой булевой функции в конъюнктивной нормальной форме, можно было бы вывести, что любая функция представляется многочленом. Но нам нужна оценка для числа операций в этом многочлене, так что мы дадим другое доказательство.)</p> <p>Сопоставьте значения из двух списков</p> <p>1) a или b 2) a и b 3) не a 4) a xor b</p> <p>a) $a \cdot b$ b) $a+1$ c) $a+b+a \cdot b$ d) $a+b$</p>	УК-2.3.2 УК-2.В.2 ОПК-2.3.1 ОПК-2.У.1 ОПК-2.В.1 ОПК-9.3.1 ОПК-9.У.1 ОПК-9.В.1
23.	<p>Мы видели, что любая булева функция $B(x_1, \dots, x_n)$ может быть представлена многочленом по модулю 2, то есть суммой мономов, каждый из которых является произведением некоторых переменных (допустим также пустой моном, не содержащий переменных; он равен 1). Какие свойства функции B соответствуют каким свойствам представляющего её многочлена?</p> <p>Сопоставьте значения из двух списков</p> <p>1) $B(0,0,0,\dots)+B(1,0,0,\dots)=1 \pmod{2}$ 2) $B(0,0,0,\dots)=1$ 3) в таблице значений булевой функции нечётное число единиц</p>	УК-2.3.2 УК-2.В.2 ОПК-2.3.1 ОПК-2.У.1 ОПК-2.В.1 ОПК-9.3.1 ОПК-9.У.1 ОПК-9.В.1

	<ul style="list-style-type: none"> a) многочлен включает в себя моном, содержащий все переменные b) многочлен включает в себя пустой моном c) многочлен включает в себя моном x_1 (содержащий единственную переменную) 	
24.	<p>Соотнесите проблемы, связанные с обучением нейронной сети с помощью градиентного спуска, с наиболее очевидными методами их решения.</p> <p>Сопоставьте значения из двух списков</p> <ul style="list-style-type: none"> 1) Застревание в локальном минимуме 2) Веса и целевая функция почти/вообще не меняются 3) Переобучение 4) Форма изменения целевой функции хорошая, но обучение протекает долго <ul style="list-style-type: none"> a) Устроить кросс-валидацию b) Попробовать несколько начальных точек (начальных весов) сети c) Увеличить learning rate d) Нормировать входы 	УК-2.3.2 УК-2.В.2 ОПК-2.3.1 ОПК-2.У.1 ОПК-2.В.1 ОПК-9.3.1 ОПК-9.У.1 ОПК-9.В.1
25.	<p>Расставьте китов алгоритма обратного распространения ошибки в том порядке, в котором они нужны, чтобы вычислить $\partial J / \partial b_3^2$ для сети, в которой $L=6$ слоёв. Неиспользованных китов, если таковые найдутся, поставьте последними.</p> <p>Расположите элементы списка в правильном порядке</p> <hr/> $\frac{\partial J}{\partial b_j^l} = \delta_j^l$ <hr/> <hr/> $\delta^l = \left((W^{l+1})^T \delta^{l+1} \right) \odot (a^l)'(z^l)$ <hr/> <hr/> $\delta^L = \nabla_{a^L} J \odot (a^L)'(z^L)$ <hr/> <hr/> $\frac{\partial J}{\partial w_{jk}^l} = a_k^{l-1} \delta_j^l$ <hr/>	УК-2.3.2 УК-2.В.2 ОПК-2.3.1 ОПК-2.У.1 ОПК-2.В.1 ОПК-9.3.1 ОПК-9.У.1 ОПК-9.В.1

Перечень тем контрольных работ по дисциплине обучающихся заочной формы обучения, представлены в таблице 19.

Таблица 19 – Перечень контрольных работ

№ п/п	Перечень контрольных работ
	Не предусмотрено

10.4. Методические материалы, определяющие процедуры оценивания индикаторов, характеризующих этапы формирования компетенций, содержатся в локальных нормативных актах ГУАП, регламентирующих порядок и процедуру проведения текущего контроля успеваемости и промежуточной аттестации обучающихся ГУАП.

11. Методические указания для обучающихся по освоению дисциплины
(Ниже приводятся рекомендации по составлению данного раздела)

11.1. Методические указания для обучающихся по освоению лекционного материала (если предусмотрено учебным планом по данной дисциплине).

Основное назначение лекционного материала – логически стройное, системное, глубокое и ясное изложение учебного материала. Назначение современной лекции в рамках дисциплины не в том, чтобы получить всю информацию по теме, а в освоении фундаментальных проблем дисциплины, методов научного познания, новейших достижений научной мысли. В учебном процессе лекция выполняет методологическую, организационную и информационную функции. Лекция раскрывает понятийный аппарат конкретной области знания, её проблемы, дает цельное представление о дисциплине, показывает взаимосвязь с другими дисциплинами.

Планируемые результаты при освоении обучающимися лекционного материала:

- получение современных, целостных, взаимосвязанных знаний, уровень которых определяется целевой установкой к каждой конкретной теме;
- получение опыта творческой работы совместно с преподавателем;
- развитие профессионально-деловых качеств, любви к предмету и самостоятельного творческого мышления.
- появление необходимого интереса, необходимого для самостоятельной работы;
- получение знаний о современном уровне развития науки и техники и о прогнозе их развития на ближайшие годы;
- научиться методически обрабатывать материал (выделять главные мысли и положения, приходить к конкретным выводам, повторять их в различных формулировках);
- получение точного понимания всех необходимых терминов и понятий.

Лекционный материал может сопровождаться демонстрацией слайдов и использованием раздаточного материала при проведении коротких дискуссий об особенностях применения отдельных тематик по дисциплине.

Структура предоставления лекционного материала:

- _____;
- _____;
- ...

Если методические указания по освоению лекционного материала имеются в изданном виде, в виде электронных ресурсов библиотеки ГУАП, системы LMS, кафедры и т.д., необходимо дать на них ссылку или привести URL адрес.

11.2. Методические указания для обучающихся по участию в семинарах (если предусмотрено учебным планом по данной дисциплине)

Основной целью для обучающегося является систематизация и обобщение знаний по изучаемой теме, разделу, формирование умения работать с дополнительными источниками информации, сопоставлять и сравнивать точки зрения, конспектировать прочитанное, высказывать свою точку зрения и т.п. В соответствии с ведущей дидактической целью содержанием семинарских занятий являются узловые, наиболее

трудные для понимания и усвоения темы, разделы дисциплины. Спецификой данной формы занятий является совместная работа преподавателя и обучающегося над решением поставленной проблемы, а поиск верного ответа строится на основе чередования индивидуальной и коллективной деятельности.

При подготовке к семинарскому занятию по теме прослушанной лекции необходимо ознакомиться с планом его проведения, с литературой и научными публикациями по теме семинара.

Требования к проведению семинаров

Обязательно для заполнения преподавателем

Если методические указания по участию в семинарах имеются в изданном виде, в виде электронных ресурсов библиотеки ГУАП, системы LMS, кафедры и т.д., необходимо дать на них ссылку или привести URL адрес.

11.3. Методические указания для обучающихся по прохождению практических занятий (если предусмотрено учебным планом по данной дисциплине)

Практическое занятие является одной из основных форм организации учебного процесса, заключающаяся в выполнении обучающимися под руководством преподавателя комплекса учебных заданий с целью усвоения научно-теоретических основ учебной дисциплины, приобретения умений и навыков, опыта творческой деятельности.

Целью практического занятия для обучающегося является привитие обучающимся умений и навыков практической деятельности по изучаемой дисциплине.

Планируемые результаты при освоении обучающимися практических занятий:

- закрепление, углубление, расширение и детализация знаний при решении конкретных задач;
- развитие познавательных способностей, самостоятельности мышления, творческой активности;
- овладение новыми методами и методиками изучения конкретной учебной дисциплины;
- выработка способности логического осмысления полученных знаний для выполнения заданий;
- обеспечение рационального сочетания коллективной и индивидуальной форм обучения.

Требования к проведению практических занятий

Обязательно для заполнения преподавателем

Если методические указания по прохождению практических занятий имеются в изданном виде, в виде электронных ресурсов библиотеки ГУАП, системы LMS, кафедры и т.д., необходимо дать на них ссылку или привести URL адрес.

11.4. Методические указания для обучающихся по выполнению лабораторных работ (если предусмотрено учебным планом по данной дисциплине)

В ходе выполнения лабораторных работ обучающийся должен углубить и закрепить знания, практические навыки, овладеть современной методикой и техникой эксперимента в соответствии с квалификационной характеристикой обучающегося. Выполнение лабораторных работ состоит из экспериментально-практической, расчетно-аналитической частей и контрольных мероприятий.

Выполнение лабораторных работ обучающимся является неотъемлемой частью изучения дисциплины, определяемой учебным планом, и относится к средствам, обеспечивающим решение следующих основных задач обучающегося:

- приобретение навыков исследования процессов, явлений и объектов, изучаемых в рамках данной дисциплины;
- закрепление, развитие и детализация теоретических знаний, полученных на лекциях;
- получение новой информации по изучаемой дисциплине;
- приобретение навыков самостоятельной работы с лабораторным оборудованием и приборами.

Задание и требования к проведению лабораторных работ

Обязательно для заполнения преподавателем

Структура и форма отчета о лабораторной работе

Обязательно для заполнения преподавателем

Требования к оформлению отчета о лабораторной работе

Обязательно для заполнения преподавателем

Если методические указания по прохождению лабораторных работ имеются в изданном виде, в виде электронных ресурсов библиотеки ГУАП, системы LMS, кафедры и т.д., необходимо дать на них ссылку или привести URL адрес.

11.5. Методические указания для обучающихся по прохождению курсового проектирования/выполнения курсовой работы (если предусмотрено учебным планом по данной дисциплине)

Курсовой проект/ работа проводится с целью формирования у обучающихся опыта комплексного решения конкретных задач профессиональной деятельности.

Курсовой проект/ работа позволяет обучающемуся:

Структура пояснительной записки курсового проекта/ работы

Обязательно для заполнения преподавателем

Требования к оформлению пояснительной записки курсового проекта/ работы

Обязательно для заполнения преподавателем

Если методические указания по курсовому проектированию/ выполнению курсовой работы имеются в изданном виде, в виде электронных ресурсов библиотеки ГУАП, системы LMS, кафедры и т.д., необходимо дать на них ссылку или привести URL адрес.

11.6. Методические указания для обучающихся по прохождению самостоятельной работы

В ходе выполнения самостоятельной работы, обучающийся выполняет работу по заданию и при методическом руководстве преподавателя, но без его непосредственного участия.

Для обучающихся по заочной форме обучения, самостоятельная работа может включать в себя контрольную работу.

В процессе выполнения самостоятельной работы, у обучающегося формируется целесообразное планирование рабочего времени, которое позволяет им развивать умения и навыки в усвоении и систематизации приобретаемых знаний, обеспечивает высокий уровень успеваемости в период обучения, помогает получить навыки повышения профессионального уровня.

Методическими материалами, направляющими самостоятельную работу обучающихся являются:

- учебно-методический материал по дисциплине;
- методические указания по выполнению контрольных работ (для обучающихся по заочной форме обучения).

Если методические указания по прохождению самостоятельной работы имеются в изданном виде, в виде электронных ресурсов библиотеки ГУАП, системы LMS, кафедры и т.д., необходимо дать на них ссылку или привести URL адрес.

11.7. Методические указания для обучающихся по прохождению текущего контроля успеваемости.

Текущий контроль успеваемости предусматривает контроль качества знаний обучающихся, осуществляемого в течение семестра с целью оценивания хода освоения дисциплины.

Обязательно для заполнения преподавателем: указываются требования и методы проведения текущего контроля успеваемости, а также как результаты текущего контроля успеваемости будут учитываться при проведении промежуточной аттестации.

11.8. Методические указания для обучающихся по прохождению промежуточной аттестации.

Промежуточная аттестация обучающихся предусматривает оценивание промежуточных и окончательных результатов обучения по дисциплине. Она включает в себя:

- экзамен – форма оценки знаний, полученных обучающимся в процессе изучения всей дисциплины или ее части, навыков самостоятельной работы, способности применять их для решения практических задач. Экзамен, как правило, проводится в период экзаменационной сессии и завершается аттестационной оценкой «отлично», «хорошо», «удовлетворительно», «неудовлетворительно».

- зачет – это форма оценки знаний, полученных обучающимся в ходе изучения учебной дисциплины в целом или промежуточная (по окончании семестра) оценка знаний обучающимся по отдельным разделам дисциплины с аттестационной оценкой «зачтено» или «не зачтено».

- дифференцированный зачет – это форма оценки знаний, полученных обучающимся при изучении дисциплины, при выполнении курсовых проектов, курсовых работ, научно-исследовательских работ и прохождении практик с аттестационной оценкой «отлично», «хорошо», «удовлетворительно», «неудовлетворительно».

Обязательно для заполнения преподавателем: указываются требования и методы проведения промежуточной аттестации.

Лист внесения изменений в рабочую программу дисциплины

Дата внесения изменений и дополнений. Подпись внесшего изменения	Содержание изменений и дополнений	Дата и № протокола заседания кафедры	Подпись зав. кафедрой