МИНИСТЕРСТВО СЕЛЬСКОГО ХОЗЯЙСТВА РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ – МСХА имени К.А. ТИМИРЯЗЕВА»
(ФГБОУ ВО РГАУ - МСХА имени К.А. Тимирязева)

Институт экономики и управления АПК
Кафедра статистики и кибернетики

УТВЕРЖДАЮ:
Директор института
экономики и управления АПК
_____ Л.И. Хоружий
«28» августа 2025 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

Б1.В.23 «Программирование на языке С++»

для подготовки бакалавров

ФГОС ВО

Направление: 09.03.02 «Информационные системы и технологии»
Направленность: Компьютерные науки и технологии искусственного интеллекта

Курс 3, 4
Семестр 6, 7

Форма обучения: заочная

Год начала подготовки: 2025

Москва, 2025

Разработчики:

Демичев В.В., канд. экон. наук, доцент _____ «26» августа 2025 г.
(ФИО, ученая степень, ученое звание) (подпись)

Титов А.Д., ассистент _____ «26» августа 2025 г.
(ФИО, ученая степень, ученое звание) (подпись)

Рецензент:

Кийко П.В., канд. пед. наук. _____ «26» августа 2025 г.
(ФИО, ученая степень, ученое звание) (подпись)

Программа составлена в соответствии с требованиями ФГОС ВО по направлению подготовки 09.03.02 Информационные системы и технологии, профессионального стандарта и учебного плана 2025 года начала подготовки

Программа обсуждена на заседании кафедры статистики и кибернетики.
Протокол № 11 от «26» августа 2025 г.

И.о. зав. кафедрой Уколова А.В., канд. экон. наук, доцент _____
(ФИО, ученая степень, ученое звание) (подпись)

«26» августа 2025 г.

Согласовано:
Председатель учебно-методической
комиссии института экономики и управления АПК
Гупалова Т.Н. канд. экон. наук, доцент протокол №1 _____
(ФИО, ученая степень, ученое звание) (подпись)

«28» августа 2025 г.

И. о. зав. выпускающей кафедрой
статистики и кибернетики
Уколова А.В., канд. экон. наук, доцент _____
(ФИО, ученая степень, ученое звание) (подпись)

«26» августа 2025 г.

Заведующий отделом комплектования ЦНБ _____
(подпись)

)

СОДЕРЖАНИЕ

АННОТАЦИЯ
рабочей программы учебной дисциплины
Б1.В.23 «Программирование на языке С++»
для подготовки бакалавров по направлению 09.03.02 «Информационные системы и технологии по направленности «Компьютерные науки и технологии искусственного интеллекта»

Цель освоения дисциплины. Основная цель дисциплины «Программирование на языке С++» – овладение студентами основными методами разработки компьютерных программ посредством языка программирования С++ для решения практических задач.

Место дисциплины в учебном плане: дисциплина включена в часть учебного плана по направлению подготовки 09.03.02 «Информационные системы и технологии», формируемую участниками образовательных отношений.

Требования к результатам освоения дисциплины: в результате освоения дисциплины формируются следующие компетенции (индикаторы): ПКос-3 (ПКос-3.1; ПКос-3.2; ПКос-3.3), ПКос-4 (ПКос-4.1; ПКос-4.2; ПКос-4.3), ПКос-5 (ПКос-5.1; ПКос-5.2; ПКос-5.3), ПКос-6 (ПКос-6.1; ПКос-6.2; ПКос-6.3), ПКос-7 (ПКос-7.1; ПКос-7.2; ПКос-7.3), ПКос-8 (ПКос-8.1; ПКос-8.2; ПКос-8.3), ПКос-9 (ПКос-9.1; ПКос-9.2; ПКос-9.3).

Краткое содержание дисциплины: Среды программирования. Классическая программа на С++. Компиляция. Редактирование связей. Объекты, типы и значения. Ввод данных. Переменные. Операции и операторы. Присваивание и инициализация. Составные операторы присваивания. Имена. Типы и объекты. Безопасность типов. Преобразование типов. Вычисления. Выражения. Инструкции. Инструкция выбора. Итерация. Функции. Объявление функций. Вектор. Обход вектора. Ошибки. Источники ошибок. Ошибки времени компиляции: синтаксические ошибки; ошибки, связанные с типами. Ошибки времени редактирования связей. Ошибки времени выполнения программы. Обработка ошибок в вызывающем коде. Обработка ошибок в вызываемом коде. Сообщения об ошибках. Исключения. Логические ошибки. Отладка. Пред- постусловия. Тестирование. Написание программ. Задача. Стадии, стратегия разработки программ. Лексемы. Грамматики. Превращение грамматики в программу. Выражения. Термы. Первичные выражения. Потоки лексем. Структура программы. Объявления и определения. Инициализация. Заголовочные файлы. Область видимости. Функции. Вызов функции и возврат значения. Порядок вычислений. Пространство имен.

Типы, определенные пользователем. Классы и члены класса. Интерфейс и реализация. Разработка класса. Перечисления. Перегрузка операторов. Интерфейсы классов. Векторы и динамически выделяемая память. Память, адреса, указатели. Динами чески распределяемая память: размещение, доступ, освобождение памяти. Деструкторы. Указатели на объекты класса. Указатели и ссылки. Указатель this. Доступ к элементам вектора. Массивы. Графические классы. Проектирование графических классов. Графическое представление функций и данных. Графические пользовательские интерфейсы.

Общая трудоемкость дисциплины составляет 3 зачетные единицы (108 часа).

Промежуточный контроль: зачет.

1. Цель освоения дисциплины

Целью освоения дисциплины «Программирование на языке С++» является овладение студентами основными методами разработки компьютерных программ посредством языка программирования С++ для решения практических задач.

2. Место дисциплины в учебном процессе

Дисциплина «Программирование на языке С++» включена в часть, формируемую участниками образовательных отношений учебного плана. Дисциплина «Компьютерная графика» реализуется в соответствии с требованиями ФГОС ВО, ОПОП ВО и Учебного плана по направлению 09.03.02 «Информационные системы и технологии».

Предшествующими курсами, на которых непосредственно базируется дисциплина «Программирование на языке С++» являются «Алгоритмизация и программирование», «Программирование на языке Python».

Дисциплина «Программирование на языке С++» является основополагающей для изучения следующих дисциплин: «Инструментальные средства информационных систем», «Тестирование программного обеспечения», а также подготовки выпускной квалификационной работы.

Особенностью дисциплины является изучение инструментов языка С++ для создания высокопроизводительных компьютерных программ, разработка и программирование программного продукта для решения практических задач.

Рабочая программа дисциплины «Программирование на языке С++» для инвалидов и лиц с ограниченными возможностями здоровья разрабатывается индивидуально с учетом особенностей психофизического развития, индивидуальных возможностей и состояния здоровья таких обучающихся.

3. Перечень планируемых результатов обучения по дисциплине, соотнесенных с планируемыми результатами освоения образовательной программы

Образовательные результаты освоения дисциплины обучающимся, представлены в таблице 1.

Таблица 1

Требования к результатам освоения учебной дисциплины

№ п/п	Код компетенции	Содержание компетенции (или её части)	Индикаторы компетенций	В результате изучения учебной дисциплины обучающиеся должны:		
				знать	уметь	владеть
1	ПКос-4	Способен осуществлять разработку, отладку и рефакторинг кода программного обеспечения, интеграцию программных модулей и компонент, в том числе взаимодействующих с внешней средой, средствами выбранных языков программирования	ПКос-4.1 Знать: методы и приемы формализации и алгоритмизации поставленных задач; нотации и программные продукты для графического отображения алгоритмов; алгоритмы решения типовых задач, области и способы их применения; методологии разработки программного обеспечения; синтаксис выбранного языка программирования, особенности программирования на этом языке, стандартные библиотеки языка программирования; особенности выбранной среды программирования; методы и приемы отладки программного кода, повышения читаемости программного кода; типы и форматы сообщений об ошибках, предупреждений	базовые алгоритмические конструкции; синтаксис языка C++: типы данных, переменные, операторы, функции, структуры; принципы работы с памятью; стандартную библиотеку C++; основы модульной разработки: заголовочные файлы, компиляция, линковка	-	-

№ п/п	Код компетенции	Содержание компетенции (или её части)	Индикаторы компетенций	В результате изучения учебной дисциплины обучающиеся должны:		
				знать	уметь	владеть
			ПКос-4.2 Уметь: использовать методы и приемы формализации и алгоритмизации поставленных задач; использовать программные продукты для графического отображения алгоритмов; применять стандартные алгоритмы в соответствующих областях; применять выбранные языки программирования для написания программного кода; использовать выбранную среду программирования; применять инструментарий для создания и актуализации исходных текстов программ; выявлять ошибки в программном коде, интерпретировать сообщения об ошибках, предупреждения, записи технологических журналов; применять методы и приемы отладки программного кода	-	формализовать простую задачу в виде алгоритма; выбирать правильные типы данных и структуры для решения задачи; писать читаемый и структурированный код на C++; разделять программу на функции и модули	-
			ПКос-4.3 Владеть навыками; составления	-	-	навыками разработки консольных приложений на C++;

№ п/п	Код компетенции	Содержание компетенции (или её части)	Индикаторы компетенций	В результате изучения учебной дисциплины обучающиеся должны:		
				знать	уметь	владеть
			формализованных описаний решений поставленных задач в соответствии с требованиями технического задания; разработки алгоритмов решения поставленных задач в соответствии с требованиями технического задания или других принятых в организации нормативных документов; создания программного кода в соответствии с техническим заданием (готовыми спецификациями); оптимизации программного кода с использованием специализированных программных средств; анализа и проверки исходного программного кода; отладки программного кода на уровне программных модулей и межмодульных взаимодействий и взаимодействий с окружением			навыками оформления кода в соответствии с базовыми правилами (стиль, именование, комментирование); навыками создания и использования пользовательских функций и структур; навыками работы с динамической памятью
2	ПКос-3	Способность выполнять	ПКос-3.1	Этапы жизненного		

№ п/п	Код компетенции	Содержание компетенции (или её части)	Индикаторы компетенций	В результате изучения учебной дисциплины обучающиеся должны:		
				знать	уметь	владеть
		интеграцию программных модулей и компонент	Знать модели процесса и принципы разработки ИС, подходы к интеграции модулей.	цикла информационной системы; Модели: водопадная, итеративная, Agile; Принципы модульной архитектуры: разделение ответственности, интерфейсы, заголовочные файлы; Механизмы взаимодействия между модулями.		
			ПКос-3.2 Уметь интегрировать модули в ИС, отлаживать их.		Создавать многомодульные проекты с использованием .h и .cpp файлов; Объявлять функции в заголовочных файлах и определять их в реализациях; Подключать и использовать сторонние библиотеки	
			ПКос-3.3			Навыками

№ п/п	Код компетенции	Содержание компетенции (или её части)	Индикаторы компетенций	В результате изучения учебной дисциплины обучающиеся должны:		
				знать	уметь	владеть
			Владеть навыками интеграции и отладки модулей.			интеграции программных модулей в единую систему; Навыками отладки межмодульных взаимодействий; Навыками организации проекта с несколькими файлами
3	ПКос-5	Способность выполнять работы по созданию (модификации) и сопровождению информационных систем	ПКос-5.1 Знать состав и классификацию ИС, методики моделирования бизнес-процессов.	Методы моделирования: ER-диаграммы, DFD, Use Case; Принципы проектирования: модульность, абстракция, инкапсуляция		
			ПКос-5.2 Уметь анализировать предметную область, создавать ИС.		Анализировать простую предметную область; Реализовать простую ИС как консольное приложение на C++; Добавлять новые функции в существующий код	
			ПКос-5.3 Владеть навыками описания			Навыками создания простых

№ п/п	Код компетенции	Содержание компетенции (или её части)	Индикаторы компетенций	В результате изучения учебной дисциплины обучающиеся должны:		
				знать	уметь	владеть
			бизнес-процессов и сопровождения ИС.			информационных систем на C++; Навыками сопровождения и модификации программного кода
4	ПКос-6	Способность выполнять работы по обеспечению функционирования баз данных и обеспечению их информационной безопасности	ПКос-6.1 Знать понятие целостности БД, способы защиты.	Работа с файлами: чтение, запись, форматы (CSV, TXT).		
			ПКос-6.2 Уметь защищать БД, резервировать данные.		Создавать простую "БД" на основе файлов (например, хранение данных в текстовых файлах); Реализовывать проверку целостности; Выполнять резервное копирование данных в отдельный файл; Читать и записывать данные в файлы из C++ программы	
			ПКос-6.3 Владеть навыками сопровождения БД.			Навыками работы с файловой "базой данных" в C++; Навыками обеспечения целостности данных в простых системах;

№ п/п	Код компетенции	Содержание компетенции (или её части)	Индикаторы компетенций	В результате изучения учебной дисциплины обучающиеся должны:		
				знать	уметь	владеть
						Навыками резервного копирования и восстановления данных; Навыками сопровождения файловых систем.
5	ПКос-7	Способность выполнять работы по обслуживанию программно-аппаратными средствами сетей и инфокоммуникаций	ПКос-7.1 Знать технологии работы в сетях.	Принципы клиент-серверной архитектуры; Примеры сетевых библиотек; Концепция потокового ввода-вывода.		
			ПКос-7.2 Уметь сравнивать программные средства сетей.		Реализовывать простое сетевое взаимодействие; Использовать стандартные средства ввода-вывода для эмуляции сетевого обмена; Обрабатывать сетевые ошибки.	
			ПКос-7.3 Владеть навыками использования средств сетей.			Навыками эмуляции сетевого взаимодействия через файлы или консоль;

№ п/п	Код компетенции	Содержание компетенции (или её части)	Индикаторы компетенций	В результате изучения учебной дисциплины обучающиеся должны:		
				знать	уметь	владеть
						Навыками использования базовых сетевых API.
6	ПКос-8	Способность использовать компоненты системных программных продуктов: компиляторы, загрузчики, сборщики и системные утилиты	ПКос-8.1 Знать назначение и классификацию системных программ.	Назначение компилятора, линковщика, отладчика, сборщика (make, CMake); Основные флаги компиляции; Принципы работы с командной строкой; Стандартные утилиты: gcc, g++, cmake, gdb, valgrind, diff, grep.		
			ПКос-8.2 Уметь использовать компоненты для решения задач.		Компилировать программу с помощью gcc/g++ из командной строки; Использовать флаги компиляции для отладки и оптимизации; Использовать отладчик GDB для пошагового выполнения;	

№ п/п	Код компетенции	Содержание компетенции (или её части)	Индикаторы компетенций	В результате изучения учебной дисциплины обучающиеся должны:		
				знать	уметь	владеть
					Проверять код на утечки памяти.	
			ПКос-8.3 Владеть навыками выбора и конфигурирования компонентов.			Навыками работы с системными утилитами разработки; Навыками настройки среды сборки; Навыками использования отладчика и профилировщика; Навыками автоматизации сборки проекта.
7	ПКос-9	Способность создания технической документации на продукцию в сфере информационных технологий, управления технической информацией	ПКос-9.1 Знать стандарты оформления технической документации.	Стандарты документации: ГОСТ 19.101–77, IEEE 830, Doxygen-документация; Элементы документации; Инструменты: Doxygen, Markdown, HTML, LaTeX.		
			ПКос-9.2 Уметь применять стандарты.		Документировать код с помощью комментариев в стиле Doxygen; Создавать README-	

№ п/п	Код компетенции	Содержание компетенции (или её части)	Индикаторы компетенций	В результате изучения учебной дисциплины обучающиеся должны:		
				знать	уметь	владеть
					файл для проекта с описанием, установкой, использованием; Оформлять отчёт по практической работе по шаблону; Использовать комментарии для объяснения сложных участков кода.	
			ПКос-9.3 Владеть навыками составления документации.			Навыками написания технической документации к программным модулям; Навыками оформления отчётов и описаний решений; Навыками использования инструментов автоматической генерации документации; Навыками структурирования информации для сопровождения проекта.

4. Структура и содержание дисциплины

4.1 Распределение трудоёмкости дисциплины по видам работ по семестрам

Общая трудоёмкость дисциплины составляет 3 зачетные единицы (108 часа), их распределение по видам работ семестрам представлено в таблице 2.

ОЧНАЯ ФОРМА ОБУЧЕНИЯ

Таблица 2

Распределение трудоёмкости дисциплины по видам работ по семестрам

Вид учебной работы	Трудоёмкость		
	час. всего/ *	в т.ч. по семестрам	
		№6	№7/*
Общая трудоёмкость дисциплины по учебному плану	**108**	**36**	**72**
1. Контактная работа:	**16,25**	**2**	**14,25/2**
Аудиторная работа	**16,25**	**2**	**14,25/2**
в том числе:			
лекции (Л)	8	2	6
практические занятия (ПЗ)	8	-	8/2
контактная работа на промежуточном контроле (КРА)	0,25	-	0,25
2. Самостоятельная работа (СРС)	**87,75**	**34**	**53,75**
самостоятельное изучение разделов, самоподготовка (проработка и повторение материала учебников и учебных пособий, подготовка к практическим занятиям, коллоквиумам, устным опросам)	**87,75**	34	53,75
Подготовка к зачёту (контроль)	4		4
Вид промежуточного контроля:	зачет		

* в том числе практическая подготовка

4.2 Содержание дисциплины

Таблица 3

Тематический план учебной дисциплины

Наименование разделов и тем дисциплин (укрупнённо)	**Всего/ ***	**Аудиторная работа**			**Внеаудиторная работа СР**
		Л	**ПЗ всего/***	**ПКР**	
Установочная лекция	36	2	-	-	34
Всего за 1 семестр	**36**	**2**	**-**	**-**	**34**
Раздел 1. Основы программирования на С++	20	2	2		16
Раздел 2. Модульность и структура программ	20	2	2		16
Раздел 3. Взаимодействие с внешней средой и сопровождение	31,75/2	2	4/2		25,75
контактная работа на промежуточном контроле (КРА)	0,25	-	-	0,25	-
Всего за 2 семестр	72/2	6	8/2	0,25	57,75
Итого по дисциплине	108/2	8	8/2	0,25	91,75

* в том числе практическая подготовка

Раздел 1. Основы программирования на С++

Раздел вводит студентов в язык C++ как средство решения прикладных задач. Рассматриваются структура программы, базовые типы данных, переменные, операторы, ввод-вывод и управляющие конструкции (ветвление, циклы). Особое внимание уделяется алгоритмизации: студенты учатся формализовать задачу, строить блок-схемы и реализовывать алгоритмы на языке. Вводится работа со средой разработки (IDE), компилятором и отладчиком, что формирует базовые навыки отладки и интерпретации ошибок.

Раздел 2. Модульность и структура программ

Раздел посвящён переходу от монолитного к модульному программированию. Студенты осваивают функции, заголовочные файлы (.h) и файлы реализации (.cpp), учатся разделять логику программы на независимые компоненты. Вводятся пользовательские типы данных (struct) и основные контейнеры стандартной библиотеки (std::string, std::vector). На этой основе реализуются простые информационные системы (например, учёт студентов), что связывает программирование с предметной областью.

Раздел 3. Взаимодействие с внешней средой и сопровождение

Завершающий раздел фокусируется на взаимодействии программы с внешним миром: работа с файлами (чтение/запись, CSV), обеспечение целостности данных и резервное копирование. Вводятся базовые понятия сетевых технологий (клиент-сервер) — на уровне эмуляции через файлы. Студенты осваивают профессиональный инструментарий: компилятор, отладчик GDB, сборщик Make/CMake (ПКос-8), а также навыки технической документации — комментарии в стиле Doxygen, README-файлы, оформление отчётов. Итоговый проект интегрирует все навыки: анализ предметной области, проектирование, кодирование, отладка, документирование и сопровождение.

4.3 Лекции/практические занятия

Таблица 4

Содержание лекций/практических занятий и контрольные мероприятия

№ п/п	Название раздела, темы	№ и название лекций/ лабораторных/ практических/ семинарских занятий	Формируемые компетенции	Вид контрольного мероприятия	Кол-во часов/ из них практическая подготовка
1	**Раздел 1. Основы программирования на C++**				
	Тема 1. Введение в C++. Структура программы. Типы данных, переменные, операторы. Ввод и вывод.	Лекция 1. Введение в C++. Структура программы. Типы данных, переменные, операторы. Ввод и вывод. Среда разработки (IDE, компилятор, отладчик).	ПКос-4.1, ПКос-8.1, ПКос-8.2	-	1
		Практическая работа №1. Написание первой программы на C++. Работа с переменными, арифметическими операциями, вводом/выводом. Использование IDE и компилятора из командной строки.	ПКос-4.2, ПКос-4.3, ПКос-8.2, ПКос-8.3	Защита практической работы	1
	Тема 2. Управляющие конструкции: условные операторы, циклы. Алгоритмизация задач.	Лекция 2. Управляющие конструкции: условные операторы, циклы. Алгоритмизация задач. Блок-схемы и графическое представление алгоритмов.	ПКос-4.1, ПКос-5.1, ПКос-9.1	-	1
		Практическая работа №2. Реализация типовых алгоритмов. Построение блок-схем. Отладка с использованием точек останова.	ПКос-4.2, ПКос-4.3, ПКос-9.2, ПКос-9.3	Защита практической работы	1
2	**Раздел 2. Модульность и структура программ**				
	Тема 3. Функции и модульная разработка. Заголовочные файлы.	Лекция 3. Функции и модульная разработка. Заголовочные файлы (.h) и файлы реализации (.cpp). Принципы интеграции модулей. Жизненный цикл ИС.	ПКос-3.1, ПКос-4.1, ПКос-5.1	-	1

№ п/п	Название раздела, темы	№ и название лекций/ лабораторных/ практических/ семинарских занятий	Формируемые компетенции	Вид контрольного мероприятия	Кол-во часов/ из них практическая подготовка
		Практическая работа №3. Разделение программы на функции. Создание многомодульного проекта. Компиляция и линковка.	ПКос-3.2, ПКос-3.3, ПКос-4.2, ПКос-4.3	Защита практической работы	1
	Тема 4. Структуры данных: struct, std::vector, std::string	Лекция 4. Структуры данных: пользовательские типы, массивы, строки. Введение в стандартную библиотеку.	ПКос-4.1, ПКос-5.1	-	1
		Практическая работа №4. Работа с std::vector и std::string. Реализация простой информационной системы с хранением данных в структурах.	ПКос-4.2, ПКос-4.3, ПКос-5.2, ПКос-5.3	Защита практической работы	1
3	**Раздел 3. Взаимодействие с внешней средой и сопровождение**				
	Тема 5. Работа с файлами. Обеспечение целостности данных.	Лекция 5. Работа с файлами. Основы обеспечения целостности данных. Резервное копирование. Введение в сетевые взаимодействия.	ПКос-6.1, ПКос-7.1	-	1
		Практическая работа №5. Чтение и запись данных в текстовые файлы (CSV/TXT). Реализация резервного копирования. Обработка ошибок ввода-вывода.	ПКос-6.2, ПКос-6.3, ПКос-4.2, ПКос-4.3	Защита практической работы	1
	Тема 6. Инструментарий разработчика и техническая документация.	Лекция 6. Инструментарий разработчика: компилятор, отладчик (GDB), сборщик (Make/CMake). Техническая документация: Doxygen, README, ГОСТ.	ПКос-8.1, ПКос-8.2, ПКос-9.1, ПКос-9.2	-	1
		Практическая работа №6. Создание Makefile. Использование GDB для отладки. Генерация документации с помощью Doxygen.	ПКос-8.2, ПКос-8.3, ПКос-9.2, ПКос-9.3	Защита практической работы	1
		Практическая работа №7. Интеграционная задача: создание многомодульной системы	ПКос-3.2, ПКос-3.3, ПКос-4.3, ПКос-5.3,	Защита практической	1/1

№ п/п	Название раздела, темы	№ и название лекций/ лабораторных/ практических/ семинарских занятий	Формируемые компетенции	Вид контрольного мероприятия	Кол-во часов/ из них практическая подготовка
		с файловым хранением, резервированием и документацией.	ПКос-6.3, ПКос-9.3	работы	
		Практическая работа №8. Консольная информационная система с полным циклом.	ПКос-3.3, ПКос-4.3, ПКос-5.3, ПКос-6.3, ПКос-8.3, ПКос-9.3	Защита практической работы	1/1

Таблица 5

Перечень вопросов для самостоятельного изучения дисциплины

№ п/п	Название раздела, темы	Перечень рассматриваемых вопросов для самостоятельного изучения
Раздел 1. Основы программирования на C++		
1	Тема 1. Введение в C++. Структура программы. Типы данных, переменные, операторы. Ввод и вывод.	Что такое директива #include и зачем она нужна? Как устроена функция main()? Какие базовые типы данных есть в C++ и чем они отличаются по размеру и назначению? Как организован ввод и вывод с помощью std::cin и std::cout? Какие основные арифметические и логические операторы используются в C++? (ПКос-4.1, ПКос-8.1)
	Тема 2. Управляющие конструкции: условные операторы, циклы. Алгоритмизация задач.	В чём разница между if, else if и switch? Когда предпочтительнее использовать цикл for, а когда — while? Как избежать бесконечного цикла? Как представить алгоритм в виде блок-схемы? Какие стандартные алгоритмы (поиск, сумма, максимум) можно реализовать с помощью циклов? (ПКос-4.1, ПКос-5.1)
Раздел 2. Модульность и структура программ		
2	Тема 3. Функции и модульная разработка. Заголовочные файлы.	Зачем нужны функции? Как объявить функцию в заголовочном файле и определить её в .cpp? Что такое прототип функции? Как избежать ошибки «multiple definition» при компиляции? Как передавать параметры в функцию (по значению, по ссылке)? (ПКос-3.1, ПКос-4.1)
	Тема 4. Структуры данных: struct, std::vector, std::string.	Чем отличается struct от простых типов? Как создать массив объектов типа struct? Почему предпочтительнее использовать std::vector вместо обычных массивов? Как работать со строками через std::string? Как добавлять, удалять и перебирать элементы вектора? (ПКос-4.1, ПКос-5.1)
Раздел 3. Взаимодействие с внешней средой и сопровождение		
3	Тема 5. Работа с файлами. Обеспечение целостности данных.	Как открыть файл для чтения и записи с помощью std::ifstream и std::ofstream? Как обрабатывать ошибку, если файл не найден? Что такое целостность данных и как её обеспечить при записи в файл (например, уникальность ID)? Как реализовать резервное копирование данных в отдельный файл? (ПКос-6.1, ПКос-6.2)
	Тема 6. Инструментарий разработчика и техническая документация.	Как провести анализ предметной области (например, «библиотека»)? Как спроектировать структуру данных для хранения книг и читателей? Как организовать взаимодействие между модулями (ввод, поиск, сохранение)? Как документировать такой проект? Какие ошибки могут возникнуть при работе с файлами и как их обработать? (ПКос-3.2, ПКос-4.3, ПКос-5.2, ПКос-6.3, ПКос-9.3)

5. Образовательные технологии

Таблица 6

Применение активных и интерактивных образовательных технологий

№ п/п	Тема и форма занятия		Наименование используемых активных и интерактивных образовательных технологий (форм обучения)
1.	Практическая работа №1. Написание первой программы на C++. Работа с переменными, арифметическими операциями, вводом/выводом. Использование IDE и компилятора из командной строки.	ПЗ	Компьютерная симуляция
2.	Практическая работа №2. Реализация типовых алгоритмов. Построение блок-схем. Отладка с использованием точек останова.	ПЗ	Компьютерная симуляция
3.	Практическая работа №3. Разделение программы на функции. Создание многомодульного проекта. Компиляция и линковка.	ПЗ	Компьютерная симуляция
4.	Практическая работа №4. Работа с std::vector и std::string. Реализация простой информационной системы с хранением данных в структурах.	ПЗ	Компьютерная симуляция
5.	Практическая работа №5. Чтение и запись данных в текстовые файлы (CSV/TXT). Реализация резервного копирования. Обработка ошибок ввода-вывода.	ПЗ	Компьютерная симуляция

6. Текущий контроль успеваемости и промежуточная аттестация по итогам освоения дисциплины

6.1. Типовые контрольные задания или иные материалы, необходимые для оценки знаний, умений и навыков и (или) опыта деятельности

Пример практических работ

Практическая работа №1

Название: Написание первой программы на C++. Работа с переменными, арифметическими операциями, вводом/выводом. Использование IDE и компилятора из командной строки.

Цель:

Освоить базовую структуру программы на C++, научиться объявлять переменные, выполнять вычисления и взаимодействовать с пользователем через консоль.

Ход выполнения:

Установить среду разработки (например, Visual Studio Code с расширением C/C++ или CLion).

Написать программу, запрашивающую у пользователя два числа и выводящую их сумму, разность, произведение и частное.

Скомпилировать программу из IDE.

Повторить сборку из командной строки с помощью g++ -o calc main.cpp.

Запустить исполняемый файл из терминала.

Требуемый результат:

Работоспособная консольная программа, корректно обрабатывающая ввод и вывод. Отчёт содержит скриншоты: код в IDE, командная строка с компиляцией и запуском.

Вопросы для защиты:

6. Из каких обязательных элементов состоит функция main()?
7. Какие типы данных вы использовали и почему?
8. Что означает флаг -o в команде g++?
9. Какие ошибки могут возникнуть при делении и как их избежать?

Практическая работа №2

Название: Реализация типовых алгоритмов. Построение блок-схем. Отладка с использованием точек останова.

Цель:

Научиться формализовать задачу в виде алгоритма, реализовывать циклы и условия, а также отлаживать программу.

Ход выполнения:

Выбрать задачу: поиск максимального элемента в массиве или сумма чётных чисел.

Построить блок-схему алгоритма (вручную или в draw.io).

Реализовать программу на C++.

Установить точку останова внутри цикла и запустить отладку в IDE.

Проследить изменение значений переменных по шагам.

Требуемый результат:

Блок-схема + исходный код + скриншот отладчика с переменными в момент останова.

Вопросы для защиты:

1. Почему важно проверять граничные значения (например, пустой массив)?
2. Как работает точка останова?
3. В чём преимущество пошаговой отладки перед «printf-отладкой»?
4. Как изменится алгоритм, если данные будут считываться из файла?

Практическая работа №3

Название: Разделение программы на функции. Создание многомодульного проекта. Компиляция и линковка.

Цель:

Освоить модульную разработку: разделение кода на заголовочные и реализационные файлы.

Ход выполнения:

Создать файл math_utils.h с объявлениями функций: int add(int a, int b);, int multiply(int a, int b);.

Создать файл math_utils.cpp с определениями этих функций.

В main.cpp подключить заголовок и вызвать функции.

Скомпилировать проект: g++ -c math_utils.cpp main.cpp, затем g++ math_utils.o main.o -o app.

Запустить и проверить работу.

Требуемый результат:

Три файла (.h, .cpp, main.cpp), успешная сборка из командной строки, работающая программа.

Вопросы для защиты:

1. Зачем нужен заголовочный файл?
2. Что произойдёт, если не указать #include "math_utils.h" в main.cpp?
3. Почему нельзя определять функции в .h-файле (без inline)?
4. Что такое объектный файл (.o) и зачем он нужен?

Практическая работа №4

Название: Работа с std::vector и std::string. Реализация простой информационной системы с хранением данных в структурах.

Цель:

Научиться использовать стандартные контейнеры и создавать пользовательские типы данных для моделирования предметной области.

Ход выполнения:

Определить структуру Student { std::string name; int group; }.

Создать вектор std::vector<Student> students.

Реализовать функции: добавление студента, вывод списка, поиск по имени.

Протестировать программу с 3–5 записями.

Требуемый результат:

Работающая программа с меню (ввод через консоль), хранящая и отображающая данные.

Вопросы для защиты:

1. Почему std::vector безопаснее обычного массива?
2. Как передать std::vector в функцию без копирования?
3. Можно ли хранить векторы внутри структур?
4. Как обрабатывать ошибки при вводе некорректного имени?

Вопросы к зачету

1. Какова структура простейшей программы на C++? Что обязательно должно присутствовать?
2. Что такое компиляция? Чем отличается компилятор от интерпретатора?
3. Назовите основные типы данных в C++ и их размер (в байтах).
4. В чём разница между int и double? Когда использовать каждый?
5. Как объявить и инициализировать константу в C++? Приведите два способа.
6. Какие операторы ввода и вывода используются в C++? Приведите пример.
7. В чём разница между if, if-else и switch?
8. Когда предпочтительно использовать цикл for, а когда – while?
9. Что такое бесконечный цикл? Как его избежать?
10. Как вычислить факториал числа с помощью цикла?
11. Что такое функция? Зачем она нужна?

12. В чём разница между параметром и аргументом функции?
13. Как передать переменную в функцию по ссылке? Зачем это делать?
14. Что такое область видимости переменной? Приведите пример локальной и глобальной переменной.
15. Что такое массив? Как объявить одномерный массив из 10 целых чисел?
16. Можно ли изменить размер обычного массива после объявления? Почему?
17. В чём разница между char[] и std::string?
18. Как объединить две строки в C++?
19. Что такое указатель? Как получить адрес переменной?
20. Как получить значение по адресу указателя?
21. Что делает оператор new? Как правильно освободить память?
22. Почему после delete рекомендуется присваивать указателю nullptr?
23. Что такое утечка памяти? Как её избежать?
24. Что такое std::vector? Почему он безопаснее обычного массива?
25. Как добавить элемент в конец std::vector?
26. Что такое std::map? Как добавить и получить значение по ключу?
27. Зачем нужны заголовочные файлы (.h)?
28. Как избежать повторного включения заголовочного файла?
29. Как скомпилировать программу из двух .cpp-файлов с помощью g++?
30. Какие виды ошибок могут возникнуть при разработке на C++?
31. Что означает ошибка «segmentation fault»? Как её диагностировать?
32. Как поставить точку останова в отладчике?
33. Что такое стек вызовов (call stack)? Зачем он нужен при отладке?
34. Какие правила именования переменных и функций вы знаете?
35. Зачем нужны комментарии в коде? Какие виды комментариев есть в C++?
36. Что такое класс и объект? Приведите пример.
37. Что такое инкапсуляция? Как она реализуется в C++?
38. Зачем нужны разделы public и private в классе?
39. Что такое конструктор? Обязательно ли его писать?
40. Что такое деструктор? Когда он вызывается?
41. Может ли класс иметь несколько конструкторов? Как это называется?
42. Что такое наследование? Приведите пример.
43. Какие члены базового класса наследуются, а какие – нет?
44. Что такое полиморфизм? Как он достигается в C++?
45. Зачем нужен ключевое слово virtual?
46. Что произойдёт, если метод не объявлен как virtual, но вызывается через указатель на базовый класс?
47. Что такое чисто виртуальная функция? Как её объявить?
48. Можно ли создать объект абстрактного класса? Почему?
49. Зачем нужен виртуальный деструктор в базовом классе?
50. Что такое перегрузка операторов? Приведите пример.
51. Почему operator<< для cout обычно перегружается как дружественная функция?
52. Что такое исключение? Как его сгенерировать и поймать?

53. Какие стандартные классы исключений вы знаете (std::invalid_argument, std::out_of_range и др.)?
54. Почему лучше ловить исключения по константной ссылке (const &)?
55. Что такое «правило трёх» в C++? (Кратко)
56. Как организовать хранение нескольких объектов одного типа? Какой контейнер предпочтителен?
57. Можно ли хранить объекты разных производных классов в одном контейнере? Как?
58. Что такое рефакторинг? Приведите пример улучшения кода.
59. Как улучшить читаемость кода без изменения логики?
60. Почему важно разделять интерфейс (.h) и реализацию (.cpp) класса?

6.2. Описание показателей и критериев контроля успеваемости, описание шкал оценивания

Текущий контроль знаний, умений и навыков проводится в форме защиты практических работ, выполняемых каждым студентом на практических занятиях. Ликвидация студентами текущих задолженностей производится также в форме выполнения индивидуальной задачи по соответствующей теме и дальнейшей ее защиты преподавателю кафедры.

Студент допускается к промежуточной аттестации (зачёту в 7 семестре), если за семестр набрал не менее 60% от максимально возможного рейтинга по итогам всех практических занятий.

Максимальная оценка за защиту одной практической работы – 10 баллов.

10 баллов – работа выполнена полностью в соответствии с требованиями; код корректен, компилируется без предупреждений, соответствует best practices современного C++ (RAII, безопасное управление памятью через std::vector/std::string, корректная работа с указателями и ссылками); реализованы все заявленные функции (например, ввод/вывод, обработка данных, работа с классами, наследование); соблюдена модульность (при необходимости – разделение на .h/.cpp); код сопровождается читаемой структурой, комментариями и/или кратким отчётом; при защите студент демонстрирует глубокое понимание использованных концепций (указатели, передача по ссылке, инкапсуляция, виртуальные функции, исключения), отвечает на все вопросы без ошибок.

9 баллов – работа выполнена полностью; возможны незначительные недочёты в оформлении кода (например, нет единого стиля именования, отсутствуют комментарии к сложным блокам) или избыточность в архитектуре (например, дублирование логики в функциях, неоптимальный выбор контейнера); при защите допущены мелкие неточности в терминологии (например, путаница между указателем и ссылкой, неточное определение конструктора или виртуальной функции), не влияющие на суть ответа.

8 баллов – работа выполнена, но содержит негрубые ошибки (например, отсутствие проверки на корректность ввода, незащищённый доступ к памяти при работе с указателями, отсутствие обработки исключений, неиспользование

const при передаче строк); при защите студент верно объясняет общую логику решения, но допускает отдельные неточности (например, не может объяснить, зачем нужен деструктор, или путает public и private), не ведущие к искажению сути.

7 баллов – работа частично соответствует требованиям; имеются ошибки в реализации (например, программа падает при некорректном вводе, массив выходит за границы, метод класса не вызывается, наследование реализовано формально, но без полиморфизма); при защите сделаны неверные выводы (например, студент считает, что new не требует delete, или не понимает разницы между передачей по значению и по ссылке), но общее понимание темы сохранено.

6–5 баллов – работа выполнена фрагментарно; код компилируется, но не выполняет основную функцию (например, программа не обрабатывает данные, класс не хранит состояние, цикл не завершается); отсутствует обработка ошибок, пользовательского ввода или базовая логика; при защите нарушена логика объяснения, наблюдается поверхностное или искажённое понимание ключевых концепций (например, непонимание назначения функции main, путаница между типами данных, неумение объяснить, что такое класс или цикл).

Менее 5 баллов – работа не выполнена или содержит фундаментальные ошибки, свидетельствующие об отсутствии освоения компетенции: код не компилируется и не содержит попыток решения задачи; использован нерелевантный подход (например, попытка решить задачу через Python-синтаксис в C++); работа скопирована без понимания логики; отсутствуют базовые элементы программы (нет #include, main, переменных, операторов ввода-вывода).

Итоговая оценка учитывает результаты рейтинговой системы контроля знаний (вклад 80%), посещаемость занятий (вклад 20%) Критерии выставления оценок по системе:

0-59 % от максимального количества баллов – «незачтено»;

60 и более %– «зачтено».

7. Учебно-методическое и информационное обеспечение дисциплины

7.1 Основная литература

1. Ростовцев, В. С. Искусственные нейронные сети : учебник для вузов / В. С. Ростовцев. — 5-е изд., стер. — Санкт-Петербург : Лань, 2025. — 216 с. — ISBN 978-5-507-50568-5. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/447392

2. Красов, А. В. Разработка защищенного программного обеспечения : учебное пособие / А. В. Красов, А. Ю. Цветков. — Санкт-Петербург : СПбГУТ им. М.А. Бонч-Бруевича, 2023. — 154 с. — ISBN 978-5-89160-308-0. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/425906

3. Огнева, М. В. Программирование на языке C++: практический курс : учебное пособие для вузов / М. В. Огнева, Е. В. Кудрина. – Москва :

Издательство Юрайт, 2022. – 335 с. – (Высшее образование). – ISBN 978-5-534-05123-0. – Текст : электронный // Образовательная платформа Юрайт [сайт]. – URL: https://urait.ru/bcode/492984

7.2 Дополнительная литература

1. Гольдберг, Й. Нейросетевые методы в обработке естественного языка : руководство / Й. Гольдберг ; перевод с английского А. А. Слинкина. — Москва : ДМК Пресс, 2019. — 282 с. — ISBN 978-5-97060-754-1. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/131704

2. Ганегедара, Т. Обработка естественного языка с TensorFlow : руководство / Т. Ганегедара ; перевод с английского В. С. Яценкова. — Москва : ДМК Пресс, 2020. — 382 с. — ISBN 978-5-97060-756-5. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/140584

3. Объектно-ориентированное программирование на C++ : учебник / И. В. Баранова, С. Н. Баранов, И. В. Баженова [и др.]. – Красноярск : СФУ, 2019. – 288 с. – ISBN 978-5-7638-4034-6. – Текст : электронный // Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/157572

8. Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины

1. C++ reference https://en.cppreference.com/w/cpp
2. Справка по C++ https://ru.cppreference.com/w/cpp
3. Документация по языку C++ https://learn.microsoft.com/ru-ru/cpp/cpp/?view=msvc-170
4. C++ Programming Language https://devdocs.io/cpp/
5. Google's C++ Class. – URL: https://developers.google.com/edu/c++

9. Перечень программного обеспечения и информационных справочных систем

Таблица 9

Перечень программного обеспечения

№ п/п	Наименование раздела учебной дисциплины	Наименование программы	Тип программы	Автор	Год разработки
1	Разделы 1, 2, 3, 4	Microsoft Visual C++ (MSVC)	расчетная, обучающая, контролирующая	Microsoft Corporation	Текущая версия
2	Разделы 1, 2	Microsoft Word	обучающая, контролирующая	Microsoft Corporation	Текущая версия
3	Разделы 1, 2	Microsoft Excel	обучающая	Microsoft Corporation	Текущая версия

10. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине

Для проведения практических занятий нужен компьютерный класс с

доступом в «Интернет», оснащенный программным обеспечением в соответствии с разделом 9.

Таблица 10

Сведения об обеспеченности специализированными аудиториями, кабинетами, лабораториями

Наименование специальных помещений и помещений для самостоятельной работы (№ учебного корпуса, № аудитории)	Оснащенность специальных помещений и помещений для самостоятельной работы
1	2
учебная аудитория для проведения занятий лекционного типа, учебная аудитория для групповых и индивидуальных консультаций, учебная аудитория для текущего контроля и промежуточной аттестации (2й учебный корпус, 102 ауд.)	1. Экран с электроприводом 1 шт. (Инв. №558771/2) 2. Проектор 1 шт. (без инв. №) – приобретался не за счет средств вуза 3. Вандалоустойчивый шкаф 1 шт. (Инв.№558850/7) 4. Системный блок iP-4 541 3200 Mhz/1024 Mb/ 80 Gb / DVD-R с монитором 1 шт. (Инв. №558777/9) 5. Стенд «Сергеев Сергей Степанович 1910-1999» 1 шт. (Инв.№591013/25) 6. Огнетушитель порошковый 1 шт. (Инв. №559527) 7. Подвесное крепление к огнетушителю 1 шт. (Инв. № 559528) 8. Жалюзи 2шт. (Инв. №1107-221225, Инв. №1107-221225) 9. Лавка 20 шт. 10. Стол аудиторный 20 шт. 11. Стол для преподавателя 1 шт. 12. Стул 2 шт. 13. Доска маркерная 1 шт. 14. Трибуна напольная 1 шт. (без инв. №)
учебная аудитория для проведения занятий семинарского типа, учебная аудитория для групповых и индивидуальных консультаций, учебная аудитория для текущего контроля и промежуточной аттестации, помещение для самостоятельной работы (2й учебный корпус, 302 ауд.)	1. Системный блок Intel Core Intel Core i3-2100/4096Mb/500Gb/DVD-RW 10 шт. (Инв.№601997, Инв.№601998, Инв.№601999, Инв.№602000, Инв.№602001, Инв.№602002, Инв.№602003, Инв.№602004, Инв.№602005, Инв.№602006) 2. Монитор 10 шт. (без инв. №) - приобретались не за счет средств вуза 3. Шкаф 2 шт. (Инв.№594166, Инв.№594167) 4. Тумба 1 шт. (Инв.№594168) 5. Подвесное крепление к огнетушителю 1 шт. (Инв. № 559528) 6. Огнетушитель порошковый 1 шт. (Инв. №559527) 7. Жалюзи 1 шт. (Инв.№551557) 8. Доска магнитно-маркерная 1 шт. 9. Стол 5 шт. 10. Стол компьютерный 12 шт. 11. Стул офисный 21 шт. 12. Сейф 1 шт. (без Инв.№).
учебная аудитория для проведения занятий лекционного типа, учебная аудитория для проведения занятий семинарского типа, учебная аудитория для проведения курсового проектирования (выполнения курсовых работ), учебная аудитория для групповых и индивидуальных консультаций, учебная аудитория для текущего контроля и промежуточной аттестации (1й учебный корпус, 212 ауд.)	Количество рабочих мест: 24 Встроенные сетевые адаптеры (Intel I219-V или Realtek RTL8111H), интерфейс RJ-45, скорость 10/100/1000 Мбит/с. Точки доступа: Ubiquiti UniFi AP AC Pro, стандарты IEEE 802.11a/b/g/n/ac, частоты 2.4 ГГц (450 Мбит/с) и 5 ГГц (1300 Мбит/с), поддержка MU-MIMO, питание PoE. Структурное подразделение: Кафедра Цифровая кафедра
учебная аудитория для проведения занятий лекционного типа, учебная аудитория для проведения занятий семинарского типа, учебная аудитория для проведения курсового проектирования (выполнения курсовых работ),	Количество рабочих мест: 24 Встроенные сетевые адаптеры (Intel I219-V или Realtek RTL8111H), интерфейс RJ-45, скорость 10/100/1000 Мбит/с. Точки доступа: Ubiquiti UniFi AP AC Pro, стандарты IEEE 802.11a/b/g/n/ac, частоты 2.4 ГГц (450 Мбит/с) и 5 ГГц (1300

учебная аудитория для групповых и индивидуальных консультаций, учебная аудитория для текущего контроля и промежуточной аттестации (1й учебный корпус, 214 ауд.)	Мбит/с), поддержка MU-MIMO, питание PoE. Структурное подразделение: Кафедра Цифровая кафедра
Студенческое общежитие	Комнаты для самоподготовки
ЦНБ имени Н.И. Железнова	Читальный зал

11. Методические рекомендации обучающимся по освоению дисциплины

Предполагается, что студент выполняет практическое задание в аудитории, дома оформляет и готовится по теоретическим вопросам к защите отчета на следующем занятии.

Виды и формы отработки пропущенных занятий

Студент, пропустивший занятие, обязан предъявить преподавателю документы установленного образца, подтверждающие необходимость пропуска. Не допускается пропуск занятий без уважительной причины.

Студент, пропустивший занятия, осваивает материал самостоятельно (выполняет практическое задание по своему варианту в компьютерном классе кафедры в часы, свободные от занятий, изучает теоретические вопросы).

Студент, пропустивший лекцию, отвечает на вопросы по пропущенной теме.

12. Методические рекомендации преподавателям по организации обучения по дисциплине

На первом занятии преподаватель закрепляет за каждым студентом номер варианта для выполнения индивидуальных работ (как правило, номер варианта соответствует порядковому номеру студента в журнале преподавателя). По каждой индивидуальной работе должна быть поставлена оценка по факту ее защиты. Защиту рекомендуется проводить на следующем после получения задания занятии. Преподаватель обязан проверить соответствие выполненного задания исходным данным варианта студента. Таким образом, исключается вероятность плагиата.

Преподаватель должен стимулировать студентов к занятию научно-исследовательской работой, изучению научной литературы по теме искусственного интеллекта, в т.ч. отечественной и зарубежной периодики.

Программу разработали:

Демичев В.В., канд. экон. наук, доцент
(ФИО, ученая степень, ученое звание) _____ (подпись)

Титов А.Д., ассистент
(ФИО, ученая степень, ученое звание) _____ (подпись)

РЕЦЕНЗИЯ

на рабочую программу дисциплины
Б1.В.23 «Программирование на языке С++»
ОПОП ВО по направлению 09.03.02 «Информационные системы и технологии», направленность «Компьютерные науки и технологии искусственного интеллекта»
(квалификация выпускника – бакалавр)

Кийко Павлом Владимировичем, доцентом кафедры высшей математики, кандидатом педагогических наук (далее по тексту рецензент), проведено рецензирование рабочей программы дисциплины «Программирование на языке С++» ОПОП ВО по направлению *09.03.02 «Информационные системы и технологии»*, направленность «Компьютерные науки и технологии искусственного интеллекта» (бакалавриат) разработанной в ФГБОУ ВО «Российский государственный аграрный университет – МСХА имени К.А. Тимирязева», на кафедре статистики и кибернетики (разработчики – Демичев Вадим Владимирович, доцент, кандидат экономических наук, Титов Артем Денисович, ассистент кафедры статистики и кибернетики).

Рассмотрев представленные на рецензирование материалы, рецензент пришел к следующим выводам:

1. Предъявленная рабочая программа дисциплины « Программирование на языке С++» (далее по тексту Программа) *соответствует* требованиям ФГОС ВО по направлению *09.03.02 «Информационные системы и технологии»*. Программа *содержит* все основные разделы, *соответствует* требованиям к нормативно-методическим документам.

2. Представленная в Программе ***актуальность*** учебной дисциплины в рамках реализации ОПОП ВО *не подлежит сомнению* – дисциплина относится к части, формируемой участниками образовательных отношений учебного цикла – Б1.В

3. Представленные в Программе ***цели*** дисциплины *соответствуют* требованиям ФГОС ВО направления *09.03.02 «Информационные системы и технологии»*.

4. В соответствии с Программой за дисциплиной «Программирование на языке С++» закреплена 7 ***профессиональных компетенций, определяемых самостоятельно (21 индикатор)***. Дисциплина «Программирование на языке С++» и представленная Программа *способна реализовать* их в объявленных требованиях. Результаты обучения, представленные в Программе в категориях знать, уметь, владеть соответствуют специфике и содержанию дисциплины и демонстрируют возможность получения заявленных результатов.

5. Общая трудоёмкость дисциплины «Программирование на языке С++» составляет 3 зачётные единицы (108 часа/из них практическая подготовка 2 часа).

6. Информация о взаимосвязи изучаемых дисциплин и вопросам исключения дублирования в содержании дисциплин *соответствует* действительности. Дисциплина «Программирование на языке С++» взаимосвязана с другими дисциплинами ОПОП ВО и Учебного плана по направлению *09.03.02 «Информационные системы и технологии»* и возможность дублирования в содержании отсутствует.

7. Представленная Программа предполагает использование современных образовательных технологий, используемые при реализации различных видов учебной работы. Формы образовательных технологий *соответствуют* специфике дисциплины.

8. Программа дисциплины «Программирование на языке С++» предполагает занятия в интерактивной форме.

9. Виды, содержание и трудоёмкость самостоятельной работы студентов, представленные в Программе, *соответствуют* требованиям к подготовке выпускников, содержащимся во ФГОС ВО направления *09.03.02 «Информационные системы и технологии»*.

10. Представленные и описанные в Программе формы *текущей* оценки знаний (опрос, в форме обсуждения отдельных вопросов), *соответствуют* специфике дисциплины и требованиям к выпускникам.

Форма промежуточного контроля знаний студентов, предусмотренная Программой, осуществляется в форме зачета в 7 семестре, что *соответствует* статусу дисциплины, части, формируемой участниками образовательных отношений учебного цикла – Б1.В ФГОС ВО направления *09.03.02 «Информационные системы и технологии»*.

11. Формы оценки знаний, представленные в Программе, *соответствуют* специфике дисциплины и требованиям к выпускникам.

12. Учебно-методическое обеспечение дисциплины представлено: основной литературой – 3 источника (базовый учебник), дополнительной литературой – 3 наименования, Интернет-ресурсы – 5 источников и *соответствует* требованиям ФГОС ВО направления *09.03.02 «Информационные системы и технологии»*.

13. Материально-техническое обеспечение дисциплины соответствует специфике дисциплины «Программирование на языке С++» и обеспечивает использование современных образовательных, в том числе интерактивных методов обучения.

14. Методические рекомендации студентам и методические рекомендации преподавателям по организации обучения по дисциплине дают представление о специфике обучения по дисциплине «Программирование на языке С++».

ОБЩИЕ ВЫВОДЫ

На основании проведенного рецензирования можно сделать заключение, что характер, структура и содержание рабочей программы дисциплины «Программирование на языке С++» ОПОП ВО по направлению *09.03.02 «Информационные системы и технологии»*, направленность «Компьютерные науки и технологии искусственного интеллекта» (квалификация выпускника – бакалавр), разработанная Демичевым Вадимом Владимировичем, доцентом, кандидатом экономических наук, Титовым Артемом Денисовичем, ассистентом кафедры статистики и кибернетики, соответствует требованиям ФГОС ВО, современным требованиям экономики, рынка труда и позволит при её реализации успешно обеспечить формирование заявленных компетенций.

Рецензент: Кийко Павел Владимирович, доцент кафедры высшей математики, кандидат педагогических наук ФГБОУ ВО «Российский государственный аграрный университет– МСХА имени К.А. Тимирязева»

«26» августа 2025 г.

(подпись)