МИНОБРНАУКИ РОССИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«ИРКУТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ФГБОУ ВО «ИГУ»
Кафедра Физико-химической биологии, биоинженерии и биоинформатики

Рабочая программа дисциплины (модуля)

Наименование дисциплины: Б1.О.18 **«Основы программирования»**

Специальность: 06.05.01 «Биоинженерия и биоинформатика»

Специализация: «Биоинженерия и биоинформатика»

Квалификация выпускника: биоинженер и биоинформатик

Форма обучения: очная с элементами электронного обучения и дистанционных образовательных технологий

Согласовано с УМК биолого-почвенного факультета
Протокол №7 от 20.05.2024
Председатель _____ А. Н. Матвеев

Рекомендовано кафедрой физико-химической биологии, биоинженерии и биоинформатики
Протокол №15 от 17.04.2024
Зав. кафедрой _____ В.П. Саловарова

Иркутск 2024 г.

Содержание

стр.

I. Цель и задачи дисциплины …… 3

II. Место дисциплины в структуре ОПОП …… 3

III. Требования к результатам освоения дисциплины ……. 3

IV. Содержание и структура дисциплины ……. 7

4.1 **Содержание дисциплины, структурированное по темам, с указанием видов учебных занятий и отведенного на них количества академических часов**

……

4.2 План внеаудиторной самостоятельной работы обучающихся по дисциплине ……

4.3 Содержание учебного материала ……

4.3.1 Перечень семинарских, практических занятий и лабораторных работ ……..

4.3.2. Перечень тем (вопросов), выносимых на самостоятельное изучение в рамках самостоятельной работы студентов ……...

4.4. Методические указания по организации самостоятельной работы студентов …….

4.5. Примерная тематика курсовых работ (проектов) ……

- V. Учебно-методическое и информационное обеспечение дисциплины …… 14
- а) перечень литературы ……..
- б) периодические издания …….

в) список авторских методических разработок ……..

г) базы данных, поисково-справочные и информационные системы…….

VI. Материально-техническое обеспечение дисциплины …… 17

6.1. Учебно-лабораторное оборудование ……..

6.2. Программное обеспечение ……

6.3. Технические и электронные средства обучения ……

VII. Образовательные технологии …… 20

VIII. Оценочные материалы для текущего контроля и промежуточной аттестации …….. 21

I. Цель и задачи дисциплины:

Цель: Изучит основы и методы разработки алгоритмов их реализации на языках программирования высокого уровня, и научить применять полученные знания и навыки для решения профессиональных задач.

Задачи дисциплины:

- Изучить базовые алгоритмические конструкции и из реализацию на языках программирования высокого уровня С и С++;
- Овладение студентами приемами программирования на языах программирования высокого уровня С и С++;
- Овладеть методами обработки строк, файлов и элементами компьютерной графики языков программирования С и С++;
- Приобрести навыки практического использования языков программирования С, С++.

II. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОПОП ВО

2.1. Учебная дисциплина Б1.О.18 «**Основы программирования**» относится к обязательной части образовательной программы.

2.2. Для изучения данной учебной дисциплины необходимы знания, умения и навыки, формируемые предшествующими дисциплинами: «Математика», «Современное естествознание», «Информатика», «Иностранный язык».

2.3. Перечень последующих учебных дисциплин, для которых необходимы знания, умения и навыки, формируемые данной учебной дисциплиной: «Специальные главы математики», «Биоинформатика», «Математическая обработка результатов исследований», «Алгоритмы биоинформатики», «Моделирование биологических процессов», выполнение ВКР.

III. ТРЕБОВАНИЯ К РЕЗУЛЬТАТАМ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Процесс освоения дисциплины направлен на формирование компетенций в соответствии с ФГОС ВО и ОП ВО по данному направлению подготовки 06.05.01 «Биоинженерия и биоинформатика», специализация «Биоинженерия и биоинформатика»:

ОПК-6: Способен разрабатывать алгоритмы и компьютерные программы, пригодные для практического применения.

ОПК-7: Способен понимать принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности.

Компетенция	Индикаторы компетенций	Результаты обучения
ОПК-6 Способен разрабатывать алгоритмы и компьютерные программы, пригодные для практического применения	*ИДК ОПК-6.1* Знает принципы создания компьютерных программ, используемых в биоинформатике и биоинженерии	Знать: основные синтаксические конструкции и типы данных языков программирования С и С++ . Уметь: строить сложные алгоритмы с помощью принципов структурного и нисходящего программирования. Владеть: навыками полстроения сложных алгоритмов для обработки биологических данных, производить отладку и тестирование разработанных алгоритмов
	ИДК ОПК-6.2 Использует современные	Знать: современные библиотеки и наборы функций для языков программирования С и С++, применяемы для анализа и

	IT-технологии при сборе, анализе, обработке и представлении информации	визуализации сложных данных. Уметь: использовать различные библиотеки функции для анализа сложных данных при построении алгоритмов.
	ИДК ОПК-6.3 Использует навыки создания компьютерных программ, баз данных и иных программных продуктов, используемых в биоинженерии и биоинформатике	Знать: классификацию алгоритмов, основные типы алгоритмов, синтаксис базовых алгоритмов в языках программирования С и С++. Уметь: анализировать входные и выходные данные разрабатываемого алгоритма. Владеть: навыками использования полученные знания для решения профессиональных задач
ОПК-7 Способен понимать принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности	*ИДК ОПК-7.1* Демонстрирует теоретические и практические навыки использования современных информационных технологий в области профессианальной деятельности	Знать: основные понятие и термины, применяемы при описании и разработки алгоритмов на языках программирования С и С++. Уметь: использовать полученные знания для поиска готовых алгоритмических решений различных задач в сети интернет и научной литературе. Владеть: навыками использования готовых алгоритмов при построения собственных решений для анализа сложных данных
	ИДК ОПК-7.2 Использует современные информационные технологии в рамках освоения материала и реализации задач в области профессиональной деятельности	Знать: спектр современных средств языков программирования С и С++ для решения задач в рамках современные информационные технологии Уметь: использовать полученные знания и навыки для решения профессиональных задач Владеть: теоретическими званиями и практическими навыками в области профессиональной деятельности.

VI. СОДЕРЖАНИЕ И СТРУКТУРА ДИСЦИПЛИНЫ

Объем дисциплины составляет 4 зачетных единицы, 144 часов.
Из них реализуется с использованием электронного обучения и дистанционных образовательных технологий 45 часов.

Форма промежуточной аттестации: экзамен.

4.1 Содержание дисциплины, структурированное по темам, с указанием видов учебных занятий и отведенного на них количества академических часов

№ п/н	Раздел дисциплины/тема	Семестр	Всего часов	Из них практическая подготовка обучающихся	Виды учебной работы, включая самостоятельную работу обучающихся , практическую подготовку и трудоемкость (в часах)				Форма текущего контроля успеваемости/ Форма промежуточной аттестации (по семестрам)
					Контактная работа преподавателя с обучающимися			Самостоятельная работа	
					Лекция	Семинар/ Практическое, лабораторное занятие/	Консультация		
1	2	3	4	5	6	7	8	9	10
1	**Тема 1.** Введение в предмет основы программирования, создание проектов и компиляция программ, среда разработки Dev-Cpp 5.1	1	8	2	2	2		4	КСР
2	**Тема 2.** Переменные и линейные алгоритмы в языке программирования С\С++.	1	8	2	2	2		4	КСР
3	**Тема 3.** Ветвящиеся алгоритмы в С и С++,операторы условного перехода, составные операторы.	1	12	4	4	4		4	КСР

4	**Тема 4.** Циклические алгоритмы в языке программирования C\C++.	1	12	4	4	4		4	КСР
5	**Тема 5.** Массивы в языке программирования C/C++.	1	12	4	4	4		4	КСР
6	**Тема 6.** Обработка массивов данных в C\C++.	1	12	4	4	4		4	КСР
7	**Тема 7.** Пользовательские функции в C\C++, структурное программирование.	1	12	4	4	4		4	КСР
8	**Тема 8.** Чтение и запись данных в файл в C\C++.	1	13	4	4	4		5	КСР
9	**Тема 9.** Работа со строковыми переменными в языке программирования C\C++.	1	12	4	4	4		4	КСР
10	**Тема 10.** Структуры в языке программирования C/C++.	1	12	4	4	4		4	КСР
11	**Тема 11.** Обработка векторной графики	1	4					4	КСР

4.2 План внеаудиторной самостоятельной работы обучающихся по дисциплине

Семестр	Название раздела, темы	Самостоятельная работа обучающихся			Оценочное средство	Учебно-методическое обеспечение самостоятельной работы
		Вид самостоятельной работы	Сроки выполнения	Трудоемкость (час.)		
1	**Тема 1.** Введение в предмет основы программирования, создание проектов и компиляция программ, среда разработки Dev-Cpp 5.1	1. Создание проекта в среде программирования Dev-Cpp 5.1. 2. Создание простейшей программы для вывода строковой переменной на экран	1	4	КСР	Раздел V а-г

Семестр	Название раздела, темы	Самостоятельная работа обучающихся			Оценочное средство	Учебно-методическое обеспечение самостоятельной работы
		Вид самостоятельной работы	Сроки выполнения	Трудоемкость (час.)		
1	**Тема 2.** Переменные и линейные алгоритмы в языке программирования C\C++.	1. Создание проекта в среде программирования Dev-Cpp 5.1. 2. Создание простейшей программы для вывода строковой переменной на экран 1. Программа калькулятор на С и С++ вычисление сложного математического выражения, числовые значения переменных вводиться с клавиатуры.	2	4	КСР	- « -
1	**Тема 3.** Ветвящиеся алгоритмы в С и С++,операторы условного перехода, составные операторы.	1. Задачи на ветвящиеся алгоритмы, вычисление выражений в зависимости от выбора условия	3	4	КСР	- « -
1	**Тема 4.** Циклические алгоритмы в языке программирования C\C++.	1. Задачи на суммирования числовых рядов. 1. Задачи на нахождение факториала числа.	4	4	КСР	- « -
1	**Тема 5.** Массивы в языке программирования C/C++.	1. Задачи на задание массивов генератором случайных чисел и нахождение среднего значения.	6	4	КСР	- « -
1	**Тема 6.** Обработка массивов данных в C\C++.	1. Задачи на поиск элементов массивов, соответствующих условию 2. Задача на сортировку элементов массивов.	8	4	КСР	- « -
1	**Тема 7.** Пользовательские функции в C\C++, структурное программирование.	1. Задачи на разработку пользовательской функции для вычисления среднего значения числового ряда 2. Задачи на разработку пользовательской функции для вычисления стандартного отклонения для элементов числового ряда	9	4	КСР	- « -

Семестр	Название раздела, темы	Самостоятельная работа обучающихся			Оценочное средство	Учебно-методическое обеспечение самостоятельной работы
		Вид самостоятельной работы	Сроки выполнения	Трудоемкость (час.)		
1	**Тема 8.** Чтение и запись данных в файл в C\C++.	1. Задачи на разработку алгоритмов для чтения и записи данных в текстовый файл	11	5	КСР	- « -
1	**Тема 9.** Работа со строковыми переменными в языке программирования C\C++.	1. Задачи, связанные с обработкой сроковых массивов, считываемых из файла с диска..	12	4	КСР	- « -
1	**Тема 10.** Структуры в языке программирования C/C++.	1. Задачи, связанные с обработкой сроковых массивов, считываемых из файла с диска с применим структур. .	13	4	КСР	- « -
1	**Тема 11.** Обработка векторной графики	1. Задачи, связанные с редактированием векторных графических файлов.	16	4	КСР	- « -
Общий объем самостоятельной работы по дисциплине (час) – **45**						
Из них объем самостоятельной работы с использованием электронного обучения и дистанционных образовательных технологий 45 часов.						

4.3 Содержание учебного материала

Тема 1. Введение в предмет основы программирования, создание проектов и компиляция программ, среда разработки Dev-Cpp 5.1.
В рамках темы рассказывается об особенностях установки настройки и использования среды разработки Dev-Cpp 5.1 для языков программирования С и С++

Вторая часть темы посвящена определению понятия алгоритма классификации и классификации алгоритмов. Рассматриваться способ записи алгоритмов в виде блок-схем, изучается правила начертания элементов в блок-схемах.

Тема 2. Переменные и линейные алгоритмы в языке программирования С\С++.

В рамках темы вводиться понятие переменной в языках программирования. Рассматривается вопрос хранения переменных в оперативной памяти компьютера. Изучаться действия над переменными с точки зрения логики работы аппаратной части компьютера.

Рассматриваться основные типы данных в языке программирования С, С++ (целые числа – int, дробные числа - float, дробные числа с двойной точностью - double, символьный тип данных - char). Изучается синтаксис задания переменны в языке программирования С, С++ и способы преобразования типов данных.
Даться понятие линейного алгоритма, изучается техника записи линейных алгоритмов в виде блок-схем. Изучается синтаксис реализации линейных алгоритмов на языке программирования С, С++.

Тема 3. Ветвящиеся алгоритмы в С и С++,операторы условного перехода, составные операторы.
В рамках темы вводиться понятие ветвящихся алгоритмов и составных операторов. Изучается синтаксис этих алгоритмов в языке программирования С, С++ и запись этих алгоритмов с помощью блок-схем. Водиться понятие составного оператора.

Тема 4. Циклические алгоритмы в языке программирования С\С++.

В рамках темы вводиться понятие циклических алгоритмов и связанных с ними составных операторов. Изучается синтаксис этих алгоритмов в языке программирования С, С++ и запись этих алгоритмов с помощью блок-схем. Рассматриваться две конструкции языка программирования С, С++ для реализации цикла «пока» и цикла «до». Проводиться решение типовых задач (суммирование рядов, множественное перемножение) средствами языка программирования С, С++.

Тема 5. Массивы в языке программирования С/С++.
Вводиться понятие массива переменных для хранения данных и их обработки. Рассматривается синтаксис задания простых и динамических массивов в языке программирования С, С++.

Тема 6. Обработка массивов данных в С\С++.
Рассматриваться серия типовых задач, связанных с обработкой массивов переменных в С, С++ (поиск максимального и минимального элементов, поиск элементов, удовлетворяющих условию, сортировка элементов массива возрастанию и убыванию).

Тема 7. Указатели в С и С++, разработка алгоритмы с использованием указателей.

Вводиться понятие указателя в С, С++, определяется связь указателя и массива данных. Рассматриваться ряд типовых задач с использованием указателей.

Тема 7. Пользовательские функции в С\С++, структурное программирование.

В рамках темы вводиться понятие функции, изучается синтаксис задания функции в С, С++. Рассматриваться серия примеров, в которых задаться пользовательские функции для решения типовых задач программирования.

Вводиться понятие нисходящего (структурного) программирования – способа разработки программ путем разбиения большой программной задачи на отдельные блоки с реализации каждого блока в виде отдельной функции. Изучаться типовые способы отладки и тестирования сложных программных проектов.

Рассматривается серия типовых задач, реализуемых с помощью метода нисходящего программирования, формирующие навыки реализации, отладки и тестирования сложных программных проектов.

Тема 8. Чтение и запись данных в файл в С\С++.

Рассматривается ряд вопросов связанных с чтением и записью данных в тактовые и бинарные файлы. Изучаться примеры формирования файлов для чтения и записи пользовательских данных полученных в ходе компьютерных вычислений

Тема 9. Работа со строковыми переменными в языке программирования С\С++.

В рамках темы рассматривается процесс обработки строковых переменных в языках программирования С, С++. Изучается ряд задач связанных с обработкой срок с помощью циклических алгоритмов и библиотечных функций языков программирования С и С++. Изучается серия типовых функции для обработки строк (разделение строки на подстроки, слияние двух или нескольких строк, сравнение строк). Приводиться примеры работы со строками, хранящими информации о последовательностях ДНК и аминокислотных последовательностях.

Тема 10. Структуры в языке программирования С/С++.

В рамках темы вводиться понятие структур. Рассматривается синтаксис определения структур с из внутренним содержимым. Рассматриваться вопросы, связанные с применением структур как входных параметров и результирующих переменных для пользовательских функций.

Тема 11. Обработка векторной графики.

В рамках темы рассматривается векторная графика – это тип компьютерной графики, в котором изображения описываются с помощью геометрических примитивов, таких как точки, линии, кривые и многоугольники, а не пикселей. Векторные изображения определяются математическими уравнениями, что позволяет масштабировать их до любого размера без потери качества и четкости. Изучаться вопросы использование векторной графики для отображения результатов биоинформационного анализа.

3.3.1. Перечень семинарских и практических занятий.

№ п/н	№ раздела и темы	Наименование семинаров, практических и лабораторных работ	Трудоемкость (час.)		Оценочные средства	Формируемые компетенции (индикаторы) *
			Всего часов	Из них практическая подготовка		
1	**2**	**3**	**4**	**5**	**6**	**7**
1	**Тема 1**	Введение в предмет основы программирования, создание проектов и компиляция программ, среда разработки Dev-	2	2	КСР	ОПК-6 *ИДК ОПК-6.1* *ИДК ОПК-6.2* ОПК-7 *ИДК ОПК-7.1* *ИДК ОПК-7.2*

		Срр 5.1				
2	**Тема 2**	Переменные и линейные алгоритмы в языке программирования С\С++.	2	2	КСР	ОПК-6 *ИДК ОПК-6.1* *ИДК ОПК-6.2* ОПК-7 *ИДК ОПК-7.1* *ИДК ОПК-7.2*
3	**Тема 3**	Ветвящиеся алгоритмы в С и С++,операторы условного перехода, составные операторы.	4	4	КСР	ОПК-6 *ИДК ОПК-6.1* *ИДК ОПК-6.2* ОПК-7 *ИДК ОПК-7.1* *ИДК ОПК-7.2*
4	**Тема 4**	Циклические алгоритмы в языке программирования С\С++.	4	4	КСР	ОПК-6 *ИДК ОПК-6.1* *ИДК ОПК-6.2* ОПК-7 *ИДК ОПК-7.1* *ИДК ОПК-7.2*
5	**Тема 5**	Массивы в языке программирования С/С++.	4	4	КСР	ОПК-6 *ИДК ОПК-6.1* *ИДК ОПК-6.2* ОПК-7 *ИДК ОПК-7.1* *ИДК ОПК-7.2*
6	**Тема 6**	Обработка массивов данных в С\С++.	4	4	КСР	ОПК-6 *ИДК ОПК-6.1* *ИДК ОПК-6.2* ОПК-7 *ИДК ОПК-7.1* *ИДК ОПК-7.2*
7	**Тема 7**	Пользовательские функции в С\С++, структурное программирование.	4	4	КСР	ОПК-6 *ИДК ОПК-6.1* *ИДК ОПК-6.2* ОПК-7 *ИДК ОПК-7.1* *ИДК ОПК-7.2*
8	**Тема 8**	Чтение и запись данных в файл в С\С++.	4	4	КСР	ОПК-6 *ИДК ОПК-6.1* *ИДК ОПК-6.2* ОПК-7 *ИДК ОПК-7.1* *ИДК ОПК-7.2*
	Тема 8	Работа со строковыми переменными в языке программирования С\С++.	4	4	КСР	ОПК-6 *ИДК ОПК-6.1* *ИДК ОПК-6.2* ОПК-7 *ИДК ОПК-7.1* *ИДК ОПК-7.2*
	Тема 10	Структуры в языке программирования С/С++.	4	4	КСР	ОПК-6 *ИДК ОПК-6.1* *ИДК ОПК-6.2* ОПК-7 *ИДК ОПК-7.1*

						ИДК ОПК-7.2

4.3.2. Перечень тем (вопросов), выносимых на самостоятельное изучение студентами в рамках самостоятельной работы (СРС)

№ п/п	Тема	Задание	Формируемая компетенция	ИДК
1.	**Тема 11**. Обработка векторной графики.	Самостоятельное изучение темы «Сводобно распространяемая программа для обработки векторной графики *Inkscape* и ее использование для визуализации научной графики». Выполнение и подготовка отчета по самостоятельному заданию.	ОПК-7	*ИДК* ОПК-7.*1* *ИДК* ОПК-7.*2*

4.4. Методические указания по организации самостоятельной работы студентов

Самостоятельная работа студентов является составной частью учебного процесса и имеет целью закрепление и углубление полученных знаний и навыков, поиск и приобретение новых знаний, а также выполнение учебных заданий, подготовку к предстоящим занятиям, и экзамену по предмету.

Для организации самостоятельной работы по дисциплине «Основы программирования» используются следующие формы самостоятельной учебной работы:

- Работа по изучение темы с использованием материалов практического занятия.
- Подбор, изучение, анализ рекомендованной литературы.
- Изучения тем занятий, вынесенных на самостоятельное изучение.
- Самостоятельное изучение синтаксических конструкции и дополнительных библиотек для С и С++.
- Самостоятельное решения домашних задач по разработки алгоритмов по изучаемым темам.
- Подготовка письменных отчетов по решению самостоятельных заданий задач.
- Подготовка к экзамену.

Письменный отчет по решению домашних заданий – это отчет о выполнении домешенного задания по темам дисциплины, содержащий следующую информацию:

- Ф.И.О. номер группы магистранта;
- номер задания;
- формулировка задания;
- описание результат решения задания с приведением файлов алгоритмов и их блок-схем в соответствии с формулировкой задания.

Критерий оценки отчета по решению домашнего задания:

- Оценка «зачтено». Задание выполнено правильно и в полном объёме, все алгоритмы и графики, согласно формулировке задания, предоставлены в отчете.
- Оценка «не зачтено». Задание выполнено неправильно или не в полном объёме, возвращается на переделку и доработку.

Подготовка к экзамену в виде тестирования. К экзамену в виде тестирования допускаются студенты, получившие зачеты по всем самостоятельным заданиям.

4.5. Примерная тематика курсовых работ (проектов): не предусмотрены учебным планом.

- **V. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ**

а) основная литература

1. Нейбауэр А. Моя первая программа на C/C++. Издательство: Питер, 2002. – 267 с. Книга доступна по ссылке: https://codernet.ru/books/c_plus/moya_pervaya_programma_na_c_a_nejbauer/
2. МакГрат М. Программирование на C для начинающих. Пер. с англ. Рейтман М. А. Издательство: Москва, 2016. – 193 с. Книга доступна по ссылке: https://codernet.ru/books/c_plus/programmirovanie_na_s_dlya_nachinayushhix_makgrat/

б) дополнительная литература

1. Голуб А. И. Правила программирования на Си и Си++**.** Пер. с англ. Зацепин В. Издательство: Москва, 2001. – 241 с. Книга доступна по ссылке: https://codernet.ru/books/c_plus/pravila_programmirovaniya_na_si_i_si/
2. Крупник А. Изучаем Си. Издательство: Питер, 2001. – 233 с. Книга доступна по ссылке: https://codernet.ru/books/c_plus/izuchaem_ci_a_krupnik/
3. Перри Г., Миллер Д. Программирование на C для начинающих. 3-е изд. Пер. с англ. Рейтман М. А. Издательство: Москва, 2015. – 369 с. Книга доступна по ссылке: https://codernet.ru/books/c_plus/programmirovanie_na_s_dlya_nachinayushhix_3-e_izd/
4. Кобзарь А.И. «Прикладная математическая статистика», для инженеров и научных работников. Издательство Москва: ФИЗМАТЛИТ, 2006 – 816 с. Книга доступна по ссылке: http://www.ph4s.ru/books/book_mat/statistika/kobzar.rar

в) периодические издания

1. https://www.matbio.org/ - сайт журнала «Математическая биология и биоинформатика». Содержит большое количество статей в pdf – формате.

г) базы данных, информационно-справочные и поисковые системы

1. http://www.biometrica.tomsk.ru/ - электронный журнал «Биометрика» для медиков и биологов – сторонников доказательной биомедицины. Содержит большое количество статей и иных материалов, посвященных математическим моделям в биологии.
2. http://www.dmb.biophys.msu.ru/models - ресурс по динамическим моделям в билогии, модели динамики популяций.
3. https://www.elibrary.ru – электронная библиотека научных статей, монографии и материалов конференций, выпущенных Российскими учеными.
4. https://pubmed.ncbi.nlm.nih.gov/ - международная база данных научных статей и монографий, посвященная различным вопросам биологии.
5. https://apps.webofknowledge.com – международная база данных, индексирующая научные публикации в высокорейтинговых изданиях
6. https://stackoverflow.com - сервис вопросов и ответов для программистов Stack Overflow.
7. https://www.programmersforum.ru – форум программистов.
8. http://e-maxx.ru/algo/ — различные алгоритмы на C ++
9. https://code-live.ru/tag/cpp-manual/ - уроки программирования, освоить **основы программирования** на C++
10. ЭБС «Издательство Лань». Адрес доступа http://e.lanbook.com/
11. ЭБС «Руконт».. Адрес доступа http://rucont.ru/
12. ЭБС «Айбукс». Адрес доступа http://ibooks.ru

13. ЭБС «Юрайт». Адрес доступа: http://biblio-online.ru/

VI. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

6.1. Учебно-лабораторное оборудование:

- Аудитория для проведения занятий лабораторного типа. Компьютерный класс (учебная аудитория). Аудитория оборудована: специализированной (учебной) мебелью на 20 посадочных мест, доской меловой; оборудована техническими средствами обучения: Системный блок PentiumG850, Монитор BenQ G252HDA-1 шт.; Системный блокAthlon 2 X2 250, Монитор BenQ G252HDA – 8 шт.; Системный блок PentiumD 3.0GHz, Монитор Samsung 740N – 3 шт.; Моноблок IRU T2105P – 2 шт.; Системный блок Pentium G3250, Монитор BenQG955 – 1 шт.; Системный блок Pentium G3250, Монитор BenQ GL2250 – 1 шт.; Системный блок Pentium G3250, Монитор Samsung T200 HD – 1 шт.; Системный блок Pentium G3250, Монитор Samsung T190N – 1 шт.; Системный блок Pentium G3250, Монитор Samsung 740N – 1 шт.; Проектор BenQ MX503; экран ScreenVtdiaEcot. С неограниченным доступом к сети Интернет и обеспечением доступа в электронную информационно-образовательную среду организации, учебно-наглядными пособиями, обеспечивающими тематические иллюстрации по дисциплине «Биоинформационные технологии» в количестве 8 шт., презентации по каждой теме программы.

- Компьютерный класс (учебная аудитория) для групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, организации самостоятельной работы. Аудитория оборудована: специализированной (учебной) мебелью на 20 посадочных мест, доской меловой; оборудована техническими средствами обучения: Системный блок PentiumG850, Монитор BenQ G252HDA-1 шт.; Системный блокAthlon 2 X2 250, Монитор BenQ G252HDA – 8 шт.; Системный блок PentiumD 3.0GHz, Монитор Samsung 740N – 3 шт.; Моноблок IRU T2105P – 2 шт.; Системный блок Pentium G3250, Монитор BenQG955 – 1 шт.; Системный блок Pentium G3250, Монитор BenQ GL2250 – 1 шт.; Системный блок Pentium G3250, Монитор Samsung T200 HD – 1 шт.; Системный блок Pentium G3250, Монитор Samsung T190N – 1 шт.; Системный блок tium G3250, Монитор Samsung 740N – 1 шт.; с неограниченным доступом к сети Интернет; Проектор BenQ MX503; экран ScreenVtdiaEcot. Ноутбук Lenovo G580 – 1 шт. С неограниченным доступом к сети Интернет.

- Помещения для хранения и профилактического обслуживания учебного оборудования. Аудитория оборудована: специализированной мебелью на 11 посадочных мест; Шкаф для документов - 3 шт.; Сейф – 1 шт ; Шкаф-купе - 2 шт. ; Принтер цв.Canon LBR-5050 Laser Printer; Принтер Canon LBP-3010; Ноутбук Lenovo G580 – 1 шт.

6.2. Программное обеспечение:

DreamSpark Premium Electronic Software Delivery (3 years) Renewal (Windows 10 Education 32/64-bit (Russian) - Microsoft Imagine, Windows 7 Professional with Service Pack 1 32/64-bit (English) - Microsoft Imagine, Windows Server 2008 Enterprise and Standard without Hyper-V with SP2 32/64-bit (English) - Microsoft Imagine, Access 2016 32/64-bit (Russian) - Microsoft Imagine, Access 2010 32/64-bit (Russian) - Microsoft Imagine). Договор №03-016-14 от 30.10.2014г.

Kaspersky Endpoint Security для бизнеса - Стандартный Russian Edition. 250-499. Форус Контракт №04-114-16 от 14ноября 2016г KES. Счет №РСЦ3-000147 и АКТ от 23ноября 2016г Лиц.№1B08161103014721370444.

Microsoft Office Enterprise 2007 Russian Academic OPEN No Level. Номер Лицензии Microsoft 43364238.

Microsoft Windows XP Professional Russian Upgrade Academic OPEN No Level. Номер Лицензии Microsoft 41059241.

Office 365 профессиональный плюс для учащихся. Номер заказа: 36dde53d-7cdb-4cad-a87f-29b2a19c463e.

6.3. Технические и электронные средства:
Презентации по всем темам курса.

VII. ОБРАЗОВАТЕЛЬНЫЕ ТЕХНОЛОГИИ

При реализации различных видов учебной работы дисциплины используются как стандартные методы обучения, так и интерактивные формы проведения занятий.

Стандартные методы обучения:

1. Информационная лекция.
2. Практические занятия, предназначенные для освоения студентами базовых методов анализа данных и использованию математических методов с помощью методов математического анализа
3. Самостоятельная работа студентов (выполнение домашних заданий, выполнения домашних заданий по тема для самостоятельного изучения, подготовка к экзаменационному тесту).
4. Консультации преподавателя.

Дистанционные образовательные технологии. Под дистанционными образовательными технологиями понимаются образовательные технологии, реализуемые в основном с применением информационно-телекоммуникационных сетей - интернет-технология – задействование образовательного портала ИГУ - educa.isu.ru для предоставления письменных отчетов по домашним работам.

Наименование тем занятий с использованием дистанционных образовательных технологий:

№	Тема занятия	Вид занятия	Форма / Методы интерактивного обучения	Кол-во часов
1	**Тема 1.** Введение в предмет основы программирования, создание проектов и компиляция программ, среда разработки Dev-Cpp 5.1	самостоятельная работа	Загрузка задания для контроля на образовательный портал ИГУ educa.isu.ru	4
2	**Тема 2.** Переменные и линейные алгоритмы в языке программирования C\C++.	самостоятельная работа	Загрузка задания для контроля на образовательный портал ИГУ educa.isu.ru	4
3	**Тема 3.** Ветвящиеся алгоритмы в C и C++,операторы условного перехода, составные операторы.	самостоятельная работа	Загрузка задания для контроля на образовательный портал ИГУ educa.isu.ru	4
4	**Тема 4.** Циклические алгоритмы в языке программирования C\C++.	самостоятельная работа	Загрузка задания для контроля на образовательный портал ИГУ educa.isu.ru	4
5	**Тема 5.** Массивы в языке программирования C/C++.	самостоятельная работа	Загрузка задания для контроля на образовательный	4

			портал ИГУ educa.isu.ru	
6	**Тема 6.** Обработка массивов данных в C\C++.	самостоятельная работа	Загрузка задания для контроля на образовательный портал ИГУ educa.isu.ru	4
7	**Тема 7.** Пользовательские функции в C\C++, структурное программирование.	самостоятельная работа	Загрузка задания для контроля на образовательный портал ИГУ educa.isu.ru	4
8	**Тема 8.** Чтение и запись данных в файл в C\C++.	самостоятельная работа	Загрузка задания для контроля на образовательный портал ИГУ educa.isu.ru	5
9	**Тема 9.** Работа со строковыми переменными в языке программирования C\C++.	самостоятельная работа	Загрузка задания для контроля на образовательный портал ИГУ educa.isu.ru	4
10	**Тема 10.** Структуры в языке программирования C/C++.	самостоятельная работа	Загрузка задания для контроля на образовательный портал ИГУ educa.isu.ru	4
11	**Тема 11.** Обработка векторной графики	самостоятельная работа	Загрузка задания для контроля на образовательный портал ИГУ educa.isu.ru	4
Итого часов				45

VIII. ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ДЛЯ ТЕКУЩЕГО КОНТРОЛЯ И ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ

Входной контроль знаний по данной дисциплине не предусмотрен.

Оценочные материалы текущего контроля

Оценочные материалы текущего контроля формируются в соответствии с ЛНА университета.

В рамках дисциплины «Основы программирования» используются следующие формы текущего контроля:

- письменная работа по решению самостоятельных заданий (все формулировки заданий для самостоятельного решения с необходимыми сопроводительными материалами выложены на образовательном портале ИГУ в темах курса «Основы программирования»);

Перечень письменных работ для самостоятельного выполнения по разделам – темам дисциплины.

Задание по теме 1:

Задание по теме 2:

Задание по теме 3:

Задание по теме 4:

Задание по теме 5:

Задание по теме 6:

Задание по теме 7:

Задание по теме 8:

Задание по теме 9:

Задание по теме 10:

Задание по теме 11:

Оценочные средства для промежуточной аттестации

Промежуточная аттестация проходит в форме экзамена (1 семестр), к которому допускаются студенты, выполнившие в полном объеме аудиторную нагрузку, самостоятельную работу. Студенты, имеющие задолженность, должны выполнить все обязательные виды деятельности.

Фонд оценочных средств для промежуточной аттестации включает:

- тестовые задания для экзамена.

Назначение оценочных средств: выявить сформированность компетенций ОПК-6, ОПК-7 (см. п. III).

Тестовое задание включает два варианта по 20 вопросов по всем темам курса. К тесту допускаются студенты, задавшие все домашние заданий и получившие по каждому заданию зачет.

Критерий оценивания тестового экзаменационного задания

№	Тип задания	Критерии оценки	Результат оценивания
1	Задание закрытого типа на установление соответствия	Считается верным, если правильно установлены все соответствия (позиции одного столбца верно соотнесены с позициями другого столбца)	Полное совпадение с верным ответом – 1 балл Все остальные случаи – 0 баллов
2	Задание закрытого типа на установление последовательности	Считается верным, если правильно указана вся последовательность цифр	Полное совпадение с верным ответом – 1 балл Все остальные случаи – 0 баллов
3	Задание комбинированного типа с выбором одного верного ответа из четырех предложенных и обоснованием выбора	Считается верным, если правильно указана цифра (буква) правильного ответа и приведены корректные аргументы, используемые при выборе ответа	Полное совпадение с верным ответом – 1 балл Все остальные случаи – 0 баллов
4	Задание комбинированного типа с выбором нескольких верных ответов из четырех предложенных и обоснованием выбора	Считается верным, если правильно указаны цифры (буквы) правильного ответа и приведены корректные аргументы, используемые при выборе ответа	Полное совпадение с верным ответом – 1 балл Все остальные случаи – 0 баллов
5	Задание открытого типа с развернутым ответом	Считается верным, если ответ совпадает с эталонным ответом по содержанию и полноте	Полное соответствие эталонному ответу – 1 балл Все остальные случаи – 0 баллов

Система получения баллов за тестирование

Оценка	критерий
отлично	18 и более баллов
хорошо	16 – 17 баллов
удовлетворительно	15 – 13 баллов
неудовлетворительно	12 баллов и менее

Оценочные материалы для промежуточной аттестации (экзамена)

Тестирование (Вариант 1).

Индекс и содержание формируемой компетенции	Индикаторы компетенций	Тестовые задания для промежуточной аттестации
ОПК-6 Способен разрабатывать алгоритмы и компьютерные программы, пригодные для практического применения	*ИДК ОПК-6.1* Знает принципы создания компьютерных программ, используемых в биоинформатике и биоинженерии	**Задание комбинированного типа с выбором одного или нескольких верных ответов из четырех предложенных и аргументацией выбора** ***Вопрос 1.*** Какие из перечисленных типов данных являются целочисленными в C/C++? a) float b) int c) double d) char Варианты ответов: 1. Только b 2. b и d 3. a и c 4. a, b, c и d Правильный ответ: 2 (b и d) Аргументация: • int - это основной целочисленный тип, используемый для хранения целых чисел. • char - хоть и используется для хранения символов, в C/C++ он также является целочисленным типом, так как символы представлены числовыми кодами (обычно ASCII). • float и double - типы данных с плавающей точкой, предназначенные для хранения чисел с десятичной частью. ***Вопрос 2.*** Какие из следующих операторов используются для логического "И" в C/C++? a) & b) && c) | d) || Варианты ответов: 1. Только a 2. Только b 3. a и c 4. b и d Правильный ответ: 2 (Только b) Аргументация: • && - это оператор логического "И". Он возвращает true только если оба операнда имеют значение true. • & - это побитовый оператор "И". Он выполняет побитовую операцию "И" между операндами.
	ИДК ОПК-6.2 Использует современные IT-технологии при сборе, анализе, обработке и представлении информации	
	ИДК ОПК-6.3 Использует навыки создания компьютерных программ, баз данных и иные программных продуктов, используемых в биоинженерии и биоинформатике	
ОПК-7 Способен понимать принципы работы современных	*ИДК ОПК-7.1* Демонстрирует теоретические и практические	

информационных технологий и использовать их для решения задач профессиональной деятельности	навыки использования современных информационных технологий в области профессианальной деятельности *ИДК ОПК-7.1* Использует современные информационные технологии в рамках освоения материала и реализации задач в области профессиональной деятельности	• || - это оператор логического "ИЛИ". • | - это побитовый оператор "ИЛИ". ***Вопрос 3.*** Что из следующего верно относительно указателей в C/C++? a) Указатель хранит адрес переменной. b) Указатель может хранить только адреса целых чисел. c) Указатель должен быть инициализирован при объявлении. d) Можно выполнять арифметические операции с указателями. Варианты ответов: 1. Только a 2. a и d 3. b и c 4. a, b и d Правильный ответ: 2 (a и d) Аргументация: • a - Указатель хранит адрес в памяти, где расположена переменная. • d - Можно выполнять арифметические операции (сложение, вычитание) с указателями для перемещения по памяти. Например, для доступа к элементам массива. • b - Указатель может хранить адрес переменной любого типа данных. • c - Инициализация указателя при объявлении не обязательна, но рекомендуется для избежания ошибок (обычно инициализируют nullptr или NULL). ***Вопрос 4.*** Какие из следующих утверждений верны относительно функций в C/C++? a) Функция может возвращать несколько значений. b) Функция может быть рекурсивной. c) Функция должна обязательно принимать аргументы. d) Функция может быть объявлена внутри другой функции. Варианты ответов: 1. Только b 2. b и d 3. a и c 4. a, b и d Правильный ответ: 1 (Только b) Аргументация: • b - Функция может вызывать саму себя, это называется рекурсией. • a - Функция в C/C++ может возвращать только одно значение (или void, если не возвращает ничего). Для возврата нескольких значений используются структуры, классы, массивы или указатели. • c - Функция может не принимать аргументы (аргументы могут отсутствовать). • d - В C/C++ (до C++11) функции обычно не объявляются внутри других функций, за исключением лямбда-функций (в C++11 и новее).

Вопрос 5.
Какие из перечисленных типов данных гарантированно имеют наименьший размер в памяти (среди перечисленных) в стандарте C?
a) int b) short c) long d) char
Варианты ответов:
1. Только a
2. Только d
3. b и d
4. a, b, c и d

Правильный ответ: 2 (Только d)
Аргументация:
- Стандарт C гарантирует, что char занимает наименьшее количество памяти, достаточное для хранения одного символа.
- Размер int, short и long зависит от конкретной архитектуры и компилятора, но char всегда будет наименьшим.

Вопрос 6.
Что из следующего *не* верно относительно указателей в C?
a) Указатель хранит адрес переменной. b) void* можно использовать для указания на любой тип данных. c) Указатель всегда занимает 4 байта памяти. d) Можно использовать указатели для динамического выделения памяти.
Варианты ответов:
1. Только a
2. Только c
3. b и d
4. a и d

Правильный ответ: 2 (Только c)
Аргументация:
- a - Указатель хранит адрес в памяти, где расположена переменная. Это фундаментальное свойство указателей.
- b - void* - это универсальный указатель, который может указывать на данные любого типа. Требуется приведение типа при использовании void*.
- c - Размер указателя зависит от архитектуры процессора (32-битная или 64-битная). Он может быть 4 байта (на 32-битных системах) или 8 байт (на 64-битных системах). Это утверждение *не* всегда верно.
- d – Оператор new используются для динамического выделения памяти в C, и они возвращают указатели на выделенную область.

Вопрос 7.
Какие из следующих утверждений верны относительно функций в C?
a) Функция может возвращать несколько значений через структуру. b) Функция может быть рекурсивной. c) Функция

		должна обязательно принимать аргументы. d) В C функции могут быть перегружены (иметь одинаковое имя, но разные аргументы). Варианты ответов: 1. Только b 2. a и b 3. a и c 4. a, b и d Правильный ответ: 2 (a и b) Аргументация: • a - Функция в C может вернуть структуру, которая содержит несколько полей, тем самым косвенно возвращая несколько значений. • b - Функция может вызывать саму себя, это называется рекурсией. • c - Функция может не принимать аргументы (аргументы могут отсутствовать). • d - Перегрузка функций *не* поддерживается в C. Это особенность C++. ***Вопрос 8.*** Какой способ выделения памяти позволяет создать массив, размер которого определяется во время выполнения программы? • A) Объявление массива с переменной в качестве размера: int arr[n]; где n - переменная. • B) Использование оператор new для выделения динамической памяти • C) Использование ключевого слова dynamic. • D) Объявление массива как глобальной переменной. Правильные ответы: B Аргументация: • A) Объявление массива с переменной в качестве размера: int arr[n]; где n - переменная. – это особенность, появившаяся в стандарте C99 и называется Variable Length Array (VLA). Не все компиляторы поддерживают VLA, и их использование имеет свои ограничения. • B) Использование оператор new. - Это *правильный* и наиболее распространенный способ динамического выделения памяти для массивов (и вообще, любых структур данных) в C и C++. Размер блока может быть определен во время выполнения. • C) Использование ключевого слова dynamic. - В C нет ключевого слова dynamic для выделения памяти. Для динамического выделения памяти используется оператор new. • D) Объявление массива как глобальной переменной. - Объявление массива как глобальной переменной не позволяет определить размер массива во время выполнения. Размер глобального массива должен быть известен во время компиляции. ***Вопрос 9.*** Что произойдет, если при использовании функции scanf для ввода данных в массив, пользователь введет больше

		элементов, чем размер массива? Например, массив int arr[5] и пользователь вводит 7 чисел.
		• A) scanf автоматически увеличит размер массива.
		• B) scanf запишет только первые 5 элементов, а остальные будут проигнорированы.
		• C) scanf запишет данные за пределы выделенной памяти для массива, что может привести к непредсказуемым последствиям.
		• D) Программа выдаст ошибку компиляции.
		Правильные ответы: C
		Аргументация:
		• A) scanf автоматически увеличит размер массива. - C не предоставляет автоматического расширения массивов.
		• B) scanf запишет только первые 5 элементов, а остальные будут проигнорированы. - К сожалению, scanf *не* имеет встроенной защиты от переполнения буфера. Он будет продолжать записывать данные в память, даже если массив переполнен.
		• C) scanf запишет данные за пределы выделенной памяти для массива, что может привести к непредсказуемым последствиям. - Это *именно то*, что произойдет. scanf запишет данные за пределы массива, что может привести к повреждению данных, ошибкам сегментации, или другому непредсказуемому поведению программы. Это серьезная проблема безопасности, известная как "переполнение буфера".
		• D) Программа выдаст ошибку компиляции. - Компилятор не может предвидеть, сколько данных введет пользователь во время выполнения, поэтому ошибка компиляции не произойдет.
		
		Вопрос 10.
		Какое значение будет содержать arr[2] после выполнения следующего кода?
		int arr[5] = {0};
		arr[2] = arr[0] + 5;
		arr[0] = 10;
		• A) 0
		• B) 5
		• C) 10
		• D) 15
		Правильные ответы: B
		Аргументация:
		• A) 0 - Неверно. Изначально arr[2] имеет значение 0, но потом его значение меняется.
		• B) 5 - Верно. Изначально все элементы массива инициализированы нулями. Затем arr[2] присваивается значение arr[0] + 5, что равно 0 + 5 = 5. Изменение значения arr[0] после этого не влияет на значение arr[2].
		• C) 10 - Неверно. arr[0] присваивается значение 10, но это не влияет на arr[2] после того, как arr[2] было присвоено значение.
		• D) 15 - Неверно. Это было бы верно, если бы arr[2] присваивалось значение *после* изменения arr[0].

		Вопрос 11. В каждом вопросе выберите один или несколько правильных ответов из предложенных вариантов. Обоснуйте свой выбор кратким объяснением. 1. Какой из следующих операторов является оператором цикла с предусловием в C/C++? • A) for • B) while • C) do...while • D) switch Ваш ответ: B) while Обоснование: Цикл while проверяет условие *перед* выполнением блока кода. Если условие ложно с самого начала, блок кода не выполнится ни разу. for цикл может иметь предусловие, но он также включает инициализацию и инкремент/декремент. do...while - это цикл с постусловием. switch - это оператор выбора, а не цикл. ***Вопрос 12.*** Что произойдет, если условие в цикле while всегда истинно? • A) Программа выдаст ошибку компиляции. • B) Программа не запустится. • C) Произойдет бесконечный цикл. • D) Цикл выполнится ровно один раз. Правельный ответ: C) Произойдет бесконечный цикл. Обоснование: Если условие цикла while всегда истинно (например, while(1) или while(true)), то цикл будет выполняться бесконечно, пока не будет прерван вручную или пока не произойдет ошибка, например, переполнение памяти. ***Вопрос 13.*** В каком из следующих случаев наиболее уместно использовать цикл for? • A) Когда количество итераций цикла заранее неизвестно. • B) Когда тело цикла должно выполниться хотя бы один раз. • C) Когда необходимо выполнить блок кода бесконечное количество раз. • D) Когда количество итераций цикла известно заранее или может быть легко вычислено. Правильный ответ: D) Когда количество итераций цикла известно заранее или может быть легко вычислено. Обоснование: Цикл for обычно используется, когда известно количество итераций, так как он включает инициализацию, условие и инкремент/декремент в одной строке, что делает его удобным для таких случаев. while лучше подходит, когда количество итераций заранее неизвестно, а do...while - когда нужно гарантировать хотя бы однократное выполнение. **Задание закрытого типа на установление соответствия**

		Вопрос14.
		Установите соответствие между операторами условного перехода в C/C++ (Левая колонка) и их описаниями (Правая колонка). Каждому оператору соответствует только одно описание.
		Левая колонка (Операторы)
		1. if
		2. if...else
		3. if...else if...else
		4. switch
		Правая колонка (Описания)
		A. Позволяет выбрать один из нескольких блоков кода для выполнения в зависимости от значения целочисленного выражения.
		B. Выполняет блок кода только в том случае, если указанное условие истинно.
		C. Предоставляет альтернативный блок кода для выполнения, если указанное в if условие ложно.
		D. Предоставляет несколько взаимоисключающих условий, позволяя выбрать один из нескольких блоков кода для выполнения. Последний else блок выполняется, если ни одно из предыдущих условий не было истинным.
		Установите соответствие:
		• 1 - ?
		• 2 - ?
		• 3 - ?
		• 4 - ?
		Правильный ответ:
		• 1 - B
		• 2 - C
		• 3 - D
		• 4 - A
		
		Вопрос 15.
		Установите соответствие между примерами кода или описаниями (Левая колонка) и соответствующими особенностями использования операторов цикла в C/C++ (Правая колонка). Каждому элементу левой колонки соответствует только один элемент правой колонки.
		Левая колонка (Примеры кода/Описания)
		1. Использование break для немедленного выхода из цикла.
		2. Использование continue для пропуска текущей итерации и перехода к следующей.
		3. Бесконечный цикл, который должен быть прерван с помощью break при выполнении определенного условия.
		4. Эквивалент for (int i = 0; i < 10; ++i) { ... }, реализованный с использованием цикла while.
		Правая колонка (Особенности/Примеры кода)
		A. c++ int i = 0; while (i < 10) { // ... ++i; }
		B. c++ while (true) { // ... if (condition) { break; } }

C. Цикл с ранним выходом (early exit)
D. Пропуск итерации цикла
Установите соответствие:
- 1 - ?
- 2 - ?
- 3 - ?
- 4 - ?

Правильный ответ:
- 1 - C
- 2 - D
- 3 - B
- 4 - A

Задание закрытого типа на установление последовательности

Вопрос 16.
Расположите следующие действия в правильной последовательности, описывающей основной процесс использования функции в C/C++.
Действия:
A. Вызов функции: Использование имени функции с передачей аргументов (если необходимо) для выполнения ее кода.
B. Определение функции: Предоставление кода, который будет выполнен при вызове функции.
C. Объявление функции: Указание имени функции, возвращаемого типа и типов аргументов (если есть).
Установите последовательность:
? -> ? -> ?
Правильный ответ:
C -> B -> A

Вопрос 17.
Расположите следующие действия в правильной последовательности при работе со структурой в C/C++, которая хранит числовое и текстовое значение.
Действия:
A. Присвоение значений членам структуры: Заполнение числового и текстового полей конкретного экземпляра структуры.
B. Объявление структуры: Определение структуры с объявлением члена для хранения числового значения (например, int) и члена для хранения текстового значения (например, std::string или char[]).
C. Создание экземпляра структуры: Выделение памяти для хранения одного объекта (экземпляра) определенной структуры.

D. Использование значений членов структуры: Обращение к числовому и текстовому полям для чтения, вывода или других операций.
Установите последовательность (например: A -> B -> C -> D):
? -> ? -> ? -> ?
Правильный ответ:
B -> C -> A -> D

Задание открытого типа с развернутым ответом

Вопрос 18.
Напишите функцию на языке C/C++, которая принимает на вход вещественный массив (массив типа float или double) из 20 элементов и возвращает минимальное значение, содержащееся в этом массиве. Если массив содержит следующие элементы:
{5.2, 12.8, 8.1, 23.5, 9.7, 1.9, 15.3, 3.6, 18.4, 2.0, 7.5, 11.2, 19.9, 4.3, 21.6, 6.4, 14.1, 10.7, 16.5, 25.0}

Правильный ответ:

```
double findMin(double arr[20]) {
  double minVal = arr[0]; // Инициализируем значением элемента массива

  for (int i = 0; i < 20; ++i) {
    if (arr[i] < minVal) {
      minVal = arr[i];
    }
  }

  return minVal;
}

//использование функции
double myArray[20] = {5.2, 12.8, 8.1, 23.5, 9.7, 1.9, 15.3, 3.6, 18.4, 2.0, 7.5, 11.2, 19.9, 4.3, 21.6, 6.4, 14.1, 10.7, 16.5, 25.0};
  double minValue = findMin(myArray);
  std::cout << "Минимальное значение: " << minValue << std::endl; // Вывод: Минимальное значение: 1.9
```

Вопрос 19.
Напишите функцию на языке C/C++, которая принимает на вход целочисленный массив из 20 элементов и сортирует его по возрастанию. Используйте любой известный вам алгоритм сортировки (например, сортировку пузырьком, сортировку вставками или сортировку выбором). Функция должна изменять сам переданный массив. Если массив содержит следующие элементы:

		{5, 12, 8, 23, 9, 1, 15, 3, 18, 2, 7, 11, 19, 4, 21, 6, 14, 10, 16, 25}
		
		Правильный ответ:
		// Сортировка пузырьком
		int sortArray(int arr[20]) {
		  for (int i = 0; i < 19; ++i) {
		  for (int j = 0; j < 19 - i; ++j) {
		  if (arr[j] > arr[j + 1]) {
		  // Обмен элементов
		  int temp = arr[j];
		  arr[j] = arr[j + 1];
		  arr[j + 1] = temp;
		  }
		  }
		  }
		}
		
		//использование функции
		int myArray[20] = {5, 12, 8, 23, 9, 1, 15, 3, 18, 2, 7, 11, 19, 4, 21, 6, 14, 10, 16, 25};
		
		  std::cout << "Исходный массив: ";
		  for (int i = 0; i < 20; ++i) {
		  std::cout << myArray[i] << " ";
		  }
		  std::cout << std::endl;
		
		  sortArray(myArray);
		
		  std::cout << "Отсортированный массив: ";
		  for (int i = 0; i < 20; ++i) {
		  std::cout << myArray[i] << " ";
		  }
		  std::cout << std::endl;
		
		Вопрос 20.
		Перечислите основные встроенные (базовые) типы данных, доступные в языках программирования C и C++. Для каждого типа укажите его название, краткое описание и типичный размер в памяти (в байтах).

		Правильный ответ: В языках C и C++ есть следующие основные встроенные типы данных: • **int**: Целочисленный тип. Используется для представления целых чисел (как положительных, так и отрицательных) без дробной части. Типичный размер: 4 байта. • **char**: Символьный тип. Используется для представления отдельных символов, таких как буквы, цифры и знаки препинания. На самом деле char хранит целое число, представляющее символ в соответствии с кодировкой (например, ASCII). Типичный размер: 1 байт. • **float**: Вещественный тип с одинарной точностью. Используется для представления чисел с плавающей точкой (то есть чисел с дробной частью). Типичный размер: 4 байта. • **double**: Вещественный тип с двойной точностью. Используется для представления чисел с плавающей точкой с большей точностью, чем float. Типичный размер: 8 байт. • **bool** (только в C++): Булевский (логический) тип. Используется для представления логических значений: true (истина) или false (ложь). В C для этих целей обычно используется int (где 0 означает false, а любое ненулевое значение - true). Типичный размер: 1 байт. • **void**: Пустой тип. Обозначает отсутствие значения. Используется для: o Указания, что функция не возвращает никакого значения. o Объявления универсальных указателей (указателей на данные неизвестного типа). Дополнительно, существуют модификаторы, которые можно применять к целочисленным типам для изменения их размера и диапазона значений: • **short**: Сокращенный int. Типичный размер: 2 байта. • **long**: Увеличенный int. Типичный размер: 4 байта (часто, как обычный int) или 8 байт (на 64-битных системах). • **long long** (начиная со стандарта C++11): Расширенный long int. Типичный размер: 8 байт. • **unsigned**: Означает, что целочисленный тип не может быть отрицательным, что позволяет расширить диапазон положительных значений. Например, unsigned int.

Тестирование (Вариант 2).

Индекс и содержание формируемой компетенции	**Индикаторы компетенций**	**Тестовые задания для промежуточной аттестации**
ОПК-6 Способен разрабатывать алгоритмы и компьютерные программы, пригодные	*ИДК ОПК-6.1* Знает принципы создания компьютерных программ, используемых в биоинформатике и	**Задание комбинированного типа с выбором одного или нескольких верных ответов из четырех предложенных и аргументацией выбора** ***Вопрос 1.***

для практического применения	биоинженерии	Что из перечисленного является корректным способом объявления целочисленной переменной в C? • A) int x; • B) integer x; • C) x : integer; • D) define x int; Правильные ответы: A Аргументация: • A) int x; - Это стандартный и правильный способ объявления целочисленной переменной в C. Сначала указывается тип данных (int), затем имя переменной (x), и в конце ставится точка с запятой. • B) integer x; - В C нет типа данных integer. Правильный тип для целых чисел - int. • C) x : integer; - Это синтаксис, используемый в некоторых других языках программирования (например, Pascal), но не в C. • D) define x int; - Это препроцессорная директива, используемая для определения макросов. Хотя это можно использовать для создания псевдонима int, это не стандартный способ объявления переменной и может привести к путанице. Кроме того, даже если бы это работало, переменная не была бы объявлена *фактически* до ее использования. ***Вопрос 2.*** Что из перечисленного верно относительно указателей в C? • A) Указатель хранит адрес переменной в памяти. • B) Указатель может указывать только на переменные типа int. • C) Указатель объявляется с помощью символа *. • D) NULL - это специальное значение, которое может быть присвоено указателю, чтобы обозначить, что он не указывает ни на какой объект. Правильные ответы: A, C, D Аргументация: • A) Указатель хранит адрес переменной в памяти. - Это фундаментальное определение указателя. Он "указывает" на место в памяти, где хранится значение переменной. • B) Указатель может указывать только на переменные типа int. - Это неверно. Указатели могут указывать на переменные любого типа данных (int, char, float, структуры, и т.д.). Важно, чтобы тип указателя соответствовал типу переменной, на которую он указывает. • C) Указатель объявляется с помощью символа *. - Действительно, символ * используется при объявлении указателя. Например, int *ptr; объявляет указатель ptr на целое число. • D) NULL - это специальное значение, которое может быть присвоено указателю, чтобы обозначить, что он не указывает ни на какой объект. - NULL (или 0) используется для обозначения "нулевого" указателя, то есть указателя, который в данный момент не указывает ни на какой допустимый адрес в памяти. Это полезно для избежания ошибок разыменования (попытки доступа к памяти по недействительному адресу).
	ИДК ОПК-6.2 Использует современные IT-технологии при сборе, анализе, обработке и представлении информации	
ОПК-7 Способен понимать принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности	*ИДК ОПК-6.3* Использует навыки создания компьютерных программ, баз данных и иные программных продуктов, используемых в биоинженерии и биоинформатике	
	ИДК ОПК-7.1 Демонстрирует теоретические и практические навыки использования современных информационных технологий в области профессианальной деятельности	
	ИДК ОПК-7.2 Использует современные информационные технологии в рамках освоения материала и реализации задач в области профессиональной деятельности	

		Вопрос 3. Какие из следующих функций являются частью стандартной библиотеки ввода/вывода C (stdio.h)? • A) printf • B) scanf • C) sqrt • D) fopen Правильные ответы: A, B, D Аргументация: • A) printf - Функция printf используется для форматированного вывода данных на стандартный вывод (обычно экран). Она является ключевой частью библиотеки stdio.h. • B) scanf - Функция scanf используется для форматированного ввода данных со стандартного ввода (обычно клавиатуры). Она также входит в stdio.h. • C) sqrt - Функция sqrt вычисляет квадратный корень числа. Она определена в библиотеке math.h, а не в stdio.h. • D) fopen - Функция fopen используется для открытия файла. Она является частью библиотеки stdio.h и позволяет работать с файлами для чтения и записи. ***Вопрос 4.*** Какие из следующих утверждений о массивах в C являются верными? • A) Индексы элементов массива начинаются с 1. • B) Все элементы массива должны быть одного типа данных. • C) Размер массива должен быть известен во время компиляции, если массив объявлен локально (внутри функции). • D) При передаче массива в функцию, передается копия массива. Правильные ответы: B, C Аргументация: • A) Индексы элементов массива начинаются с 1. - Неверно. В C (как и во многих других языках), индексы массивов начинаются с 0. Первый элемент массива имеет индекс 0, второй - 1, и так далее. • B) Все элементы массива должны быть одного типа данных. - Верно. Массив в C - это упорядоченный набор элементов *одного и того же* типа данных (например, int, float, char, или даже структуры). • C) Размер массива должен быть известен во время компиляции, если массив объявлен локально (внутри функции). - Верно, для массивов, объявленных локально, размер должен быть константой времени компиляции. Можно использовать динамическое выделение памяти (malloc), если размер массива заранее неизвестен. • D) При передаче массива в функцию, передается копия массива. - Неверно. При передаче массива в функцию, передается *указатель* на первый элемент массива. Это означает, что функция может изменять

		исходный массив. Копия массива *не* создается. ***Вопрос 5.*** Какой из следующих способов инициализации массива int arr[5] является корректным? • A) int arr[5] = {1, 2, 3, 4, 5}; • B) int arr[5] = {1, 2, 3}; • C) int arr[5]; arr = {1, 2, 3, 4, 5}; • D) int arr[] = {1, 2, 3, 4, 5}; Правильные ответы: A, B, D Аргументация: • A) int arr[5] = {1, 2, 3, 4, 5}; - Это правильный способ инициализации массива при объявлении. Все 5 элементов инициализируются заданными значениями. • B) int arr[5] = {1, 2, 3}; - Это тоже правильный способ. В этом случае, первые три элемента инициализируются значениями 1, 2 и 3, а остальные элементы (в данном случае, arr[3] и arr[4]) инициализируются нулями по умолчанию. • C) int arr[5]; arr = {1, 2, 3, 4, 5}; - Это *не* корректно. Вы не можете присвоить массив таким образом после объявления. Оператор присваивания {1, 2, 3, 4, 5} можно использовать только во время объявления массива. В этом случае потребуется использовать цикл, чтобы присвоить значения каждому элементу по отдельности. • D) int arr[] = {1, 2, 3, 4, 5}; - Это также правильно. Если вы инициализируете массив при объявлении, опуская размерность, компилятор определит размер массива на основе количества предоставленных инициализаторов. **Вопрос 6.** Что произойдет, если вы попытаетесь обратиться к элементу массива int arr[10] с индексом 10 (например, arr[10])? • A) Программа выдаст ошибку компиляции. • B) Программа выдаст ошибку времени при выполнении. • C) Программа продолжит работу, но результат будет непредсказуемым. • D) Программа автоматически увеличит размер массива. Правильные ответы: B, C Аргументация: • A) Программа выдаст ошибку компиляции. - Компилятор обычно *не* обнаруживает выход за границы массива во время компиляции (особенно если индекс является переменной). Поэтому ошибки компиляции, скорее всего, не будет. • B) Программа выдаст ошибку времени выполнения (segmentation fault или подобное). - Это *один из* возможных результатов. Обращение к памяти за пределами выделенной области может привести к ошибке

		сегментации (segmentation fault) или другому типу ошибки времени выполнения, которая завершит программу. • C) Программа продолжит работу, но результат будет непредсказуемым. - Это *другой* возможный результат. В C нет автоматической проверки границ массива. Обращение к arr[10] может привести к чтению или записи в область памяти, которая не принадлежит массиву. Это может привести к непредсказуемым результатам, повреждению данных, зависанию программы или другим неожиданным проблемам. Это один из самых опасных типов ошибок в C, поскольку их трудно отлавливать. • D) Программа автоматически увеличит размер массива. - Это *абсолютно неверно*. C не предоставляет автоматического расширения массивов. Размер массива фиксируется при его объявлении. ***Вопрос 7.*** Какие из следующих утверждений верны при передаче массива в функцию в C? • A) Внутри функции можно узнать размер массива с помощью оператора sizeof. • B) Функция получает копию массива, поэтому изменения внутри функции не повлияют на исходный массив. • C) Функция получает указатель на первый элемент массива. • D) Размер массива необходимо передавать в функцию отдельным параметром (если требуется знать его размер). Правильные ответы: C, D Аргументация: • A) Внутри функции можно узнать размер массива с помощью оператора sizeof. - Это *неверно*. Внутри функции, когда массив передается как аргумент, он "преобразуется" в указатель на свой первый элемент. Оператор sizeof вернет размер указателя, а не размер исходного массива. • B) Функция получает копию массива, поэтому изменения внутри функции не повлияют на исходный массив. - Это *неверно*. Как уже говорилось, функция получает указатель на первый элемент массива. Следовательно, изменения, сделанные внутри функции (например, изменение значения элемента массива), *повлияют* на исходный массив. • C) Функция получает указатель на первый элемент массива. - Это *верно*. Это основной механизм передачи массивов в функции в C. • D) Размер массива необходимо передавать в функцию отдельным параметром (если требуется знать его размер). - Это *верно* и является хорошей практикой. Поскольку sizeof не работает корректно внутри функции (см. пункт A), если функции нужно знать размер массива, его нужно передать как отдельный аргумент. Это особенно важно, если функция должна обрабатывать все элементы массива. ***Вопрос 8.*** В каком из следующих циклов гарантируется хотя бы однократное выполнение тела цикла? • A) for • B) while

		• C) do...while • D) Ни в одном из перечисленных Ваш ответ: C) do...while Обоснование: Цикл do...while сначала выполняет блок кода, а затем проверяет условие. Таким образом, тело цикла всегда выполняется хотя бы один раз. Циклы for и while проверяют условие перед выполнением, поэтому тело цикла может не выполниться ни разу. ***Вопрос 9.*** Какой из следующих фрагментов кода выведет числа от 1 до 10 включительно? • A) for (int i = 0; i < 10; i++) { printf("%d ", i); } • B) for (int i = 1; i <= 10; i++) { printf("%d ", i); } • C) int i = 1; while (i <= 10) { printf("%d ", i); i++; } • D) int i = 1; do { printf("%d ", i); i++; } while (i < 10); Правельный ответ: B) for (int i = 1; i <= 10; i++) { printf("%d ", i); } и C) int i = 1; while (i <= 10) { printf("%d ", i); i++; } Обоснование: • A) Выведет числа от 0 до 9. Условие i < 10 останавливает цикл при i = 10, не включая его. • B) Выведет числа от 1 до 10, как и требуется. Условие i <= 10 позволяет включить 10. • C) Выведет числа от 1 до 10, как и требуется. Условие i <= 10 позволяет включить 10. • D) Выведет числа от 1 до 9. Условие i < 10 останавливает цикл при i = 10, не включая его, хотя тело цикла выполняется минимум один раз. ***Вопрос 10.*** Какие ключевые слова используются для прерывания цикла в C/C++? (Выберите все подходящие варианты) • A) break • B) continue • C) return • D) exit Правильный ответ: A) break и B) continue Обоснование: • break: Прерывает выполнение цикла и передает управление оператору, следующему за циклом. • continue: Прерывает текущую итерацию цикла и переходит к следующей итерации. • return: Выходит из текущей функции, в которой находится цикл. • exit: Завершает выполнение всей программы. ***Вопрос 11.*** Какой оператор используется для вывода данных в консоль в C++?

		• A) scanf • B) printf • C) std::cout • D) std::cin Правельный ответ: C) std::cout Обоснование: std::cout - это объект класса ostream, представляющий стандартный поток вывода (обычно консоль) в C++. std::cin используется для ввода. scanf и printf - это функции из C, которые также можно использовать в C++, но std::cout и std::cin являются частью стандартной библиотеки C++ и более типобезопасны. ***Вопрос 12.*** Какой объект класса используется для записи данных в файл в C++? • A) std::cin • B) std::cout • C) std::ofstream • D) std::ifstream Ваш ответ: C) std::ofstream Обоснование: std::ofstream (output file stream) - это класс для записи данных в файл. std::ifstream (input file stream) используется для чтения данных из файла. std::cin и std::cout предназначены для работы с консолью. ***Вопрос13.*** Какой заголовочный файл необходимо подключить для работы с файлами в C++? • A) iostream • B) fstream • C) stdio.h • D) iostream.h Ваш ответ: B) fstream Обоснование: Заголовочный файл <fstream> содержит определения классов ifstream, ofstream и fstream, которые используются для работы с файлами. iostream необходим для std::cin и std::cout. stdio.h - это заголовочный файл из C, предоставляющий функции для работы с файлами, но fstream предпочтительнее в C++. iostream.h - устаревший заголовочный файл. **Задание закрытого типа на установление соответствия** ***Вопрос14.*** Установите соответствие между примерами кода или описаниями (Левая колонка) и соответствующими концепциями/особенностями использования операторов if и else в C/C++ (Правая колонка). Каждому элементу

		левой колонки соответствует только один элемент правой колонки. Левая колонка (Примеры кода/Описания) 1. Код, который проверяет, является ли число положительным, отрицательным или нулем, и выводит соответствующее сообщение. 2. Упрощенная запись условного присваивания значения переменной, избегая полной конструкции if...else. 3. Ситуация, когда вложенный оператор if не имеет блока else, и может возникнуть неоднозначность в том, к какому if относится else. 4. Пример кода: c++ if (x > 0) if (y > 0) std::cout << "Both positive"; else std::cout << "x is not positive"; Правая колонка (Концепции/Особенности) A. Dangling else problem B. Тернарный оператор C. if...else if...else конструкция D. else относится к ближайшему незакрытому if Установите соответствие: • 1 - ? • 2 - ? • 3 - ? • 4 - ? Правильный ответ: • 1 - C • 2 - B • 3 - A • 4 - D ***Вопрос 15.*** Установите соответствие между описаниями (Левая колонка) и соответствующими операторами цикла в C/C++ (Правая колонка). Каждому описанию соответствует только один оператор. Левая колонка (Описания) 1. Цикл, который гарантированно выполнится хотя бы один раз. 2. Бесконечный цикл. 3. Цикл, который позволяет выполнять блок кода до тех пор, пока заданное условие истинно. 4. Цикл, который требует явного указания инициализации, условия и обновления счетчика (инкремента или декремента). Правая колонка (Операторы) A. for B. while C. do...while

		D. for(;;) Установите соответствие: • 1 - ? • 2 - ? • 3 - ? • 4 - ? Правильный ответ: • 1 - C • 2 - D • 3 - B • 4 - A **Задание закрытого типа на установление последовательности** ***Вопрос 16.*** Расположите следующие действия в правильной последовательности, описывающей процесс использования функции в C/C++. Действия: A. Вызов функции: Использование имени функции с передачей необходимых аргументов для выполнения её кода. B. Объявление функции (Function Declaration/Prototype): Указание имени функции, типа возвращаемого значения и типов аргументов. C. Определение функции (Function Definition): Предоставление фактического кода, который будет выполнен при вызове функции. Установите последовательность (например: A -> B -> C): ? -> ? -> ? -> ? Правильный ответ: B -> C -> A ***Вопрос 17.*** Расположите следующие действия в правильной последовательности, описывающей процесс использования структуры в C/C++. Действия: A. Доступ к члену структуры: Использование оператора . (точка) или -> (стрелка) для получения или изменения значения одного из полей (членов) структуры. B. Объявление структуры (Structure Declaration): Определение нового типа данных, состоящего из нескольких переменных (членов) различных типов.

C. Создание экземпляра структуры (Structure Instantiation): Выделение памяти для переменной типа структуры.
D. Инициализация членов структуры (Structure Initialization): Присвоение начальных значений членам структуры при создании экземпляра.
Установите последовательность (например: A -> B -> C -> D):
? -> ? -> ? -> ?
Правильный ответ:
B -> C -> D -> A

Задание открытого типа с развернутым ответом

Вопрос 18.
Напишите функцию на языке C/C++, которая принимает на вход целочисленный массив из 20 элементов и возвращает максимальное значение, содержащееся в этом массиве. Если массив содержит следующие элементы:
{5, 12, 8, 23, 9, 1, 15, 3, 18, 2, 7, 11, 19, 4, 21, 6, 14, 10, 16, 25}

Правильный ответ:

```
int findMax(int arr[20]) {
  int maxVal = arr[0]; // Инициализируем значением элемента массива

  for (int i = 0; i < 20; ++i) {
    if (arr[i] > maxVal) {
      maxVal = arr[i];
    }
  }

  return maxVal;
}

//использование функции
int myArray[20] = {5, 12, 8, 23, 9, 1, 15, 3, 18, 2, 7, 11, 19, 4, 21, 6, 14, 10, 16, 25};
  int maxValue = findMax(myArray);
  std::cout << "Максимальное значение: " << maxValue << std::endl; // Вывод: Максимальное значение: 25
```

Вопрос 19.
Напишите функцию на языке C/C++, которая принимает на вход целочисленный массив из 20 элементов и сортирует его по убыванию. Используйте любой известный вам алгоритм сортировки (например, сортировку пузырьком, сортировку вставками или сортировку выбором). Функция должна изменять сам переданный

		массив. Если массив содержит следующие элементы:
		{5, 12, 8, 23, 9, 1, 15, 3, 18, 2, 7, 11, 19, 4, 21, 6, 14, 10, 16, 25}
		
		Правильный ответ:
		// Сортировка пузырьком (модифицированная для убывания)
		int sortArrayDescending(int arr[20]) {
		  for (int i = 0; i < 19; ++i) {
		  for (int j = 0; j < 19 - i; ++j) {
		  if (arr[j] < arr[j + 1]) { // Изменено условие для убывания
		  // Обмен элементов
		  int temp = arr[j];
		  arr[j] = arr[j + 1];
		  arr[j + 1] = temp;
		  }
		  }
		  }
		}
		
		//Использование функции
		int myArray[20] = {5, 12, 8, 23, 9, 1, 15, 3, 18, 2, 7, 11, 19, 4, 21, 6, 14, 10, 16, 25};
		
		  std::cout << "Исходный массив: ";
		  for (int i = 0; i < 20; ++i) {
		  std::cout << myArray[i] << " ";
		  }
		  std::cout << std::endl;
		
		  sortArrayDescending(myArray);
		
		  std::cout << "Отсортированный массив (по убыванию): ";
		  for (int i = 0; i < 20; ++i) {
		  std::cout << myArray[i] << " ";
		  }
		  std::cout << std::endl;
		
		Вопрос 20.
		Перечислите основные методы и функции для работы со строками, предоставляемые классом std::string в языке программирования C++. Для каждого метода/функции укажите его название, краткое описание и пример

		использования. Правильный ответ: Класс std::string из библиотеки <string> предоставляет мощные и удобные средства для работы со строками в C++. Вот некоторые из основных методов и функций: operator= (присваивание): Присваивает одной строке значение другой. Описание: Присваивает строке новое значение. Может принимать другую строку, строку C-style (массив char) или отдельный символ. append(): Добавляет текст в конец строки. Описание: Добавляет символы в конец строки. Может принимать другую строку, строку C-style или указанное количество символов из другой строки. insert(): Вставляет текст в строку в указанную позицию. Описание: Вставляет символы в строку по указанному индексу. replace(): Заменяет часть строки другой строкой. Описание: Заменяет указанное количество символов в строке другой строкой. substr(): Создает подстроку. Описание: Возвращает новую строку, содержащую подстроку исходной строки. find(): Находит первое вхождение подстроки. Описание: Ищет первое вхождение указанной подстроки в строке. Возвращает индекс первого символа найденной подстроки или std::string::npos, если подстрока не найдена. rfind(): Находит последнее вхождение подстроки. Описание: Ищет последнее вхождение указанной подстроки в строке. Возвращает индекс первого символа найденной подстроки или std::string::npos, если подстрока не найдена. length() / size(): Возвращает длину строки. Описание: Возвращает количество символов в строке (не включая завершающий нулевой символ, которого в std::string и так нет). length() и size() делают одно и то же. clear(): Очищает строку (делает ее пустой). Описание: Удаляет все символы из строки, делая ее пустой.

		compare(): Сравнивает две строки. Описание: Сравнивает строку с другой строкой. Возвращает: Отрицательное значение, если строка меньше другой строки. Ноль, если строки равны. Положительное значение, если строка больше другой строки.

Разработчик:

_____ доцент Букин Ю.С.

(подпись)

Программа составлена в соответствии с требованиями ФГОС ВО по направлению 06.05.01 «Биоинженерия и биоинформатика».

Программа рассмотрена на заседании кафедры физико-химической биологии, биоинженерии и биоинформатики 17.04.2024 г. протокол № 15.

Зав. кафедрой, д.б.н., профессор В.П. Саловарова _____