

МИНИСТЕРСТВО ПРОСВЕЩЕНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ  
ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ЛУГАНСКИЙ ГОСУДАРСТВЕННЫЙ ПЕДАГОГИЧЕСКИЙ УНИВЕРСИТЕТ»  
(ФГБОУ ВО «ЛПУ»)**

**Структурное подразделение** Институт физико-математического  
образования, информационных и обслуживающих технологий  
**Кафедра** информационных образовательных технологий и систем

**УТВЕРЖДАЮ**

Врио директора ИФМОИОТ

Е.А. Журавлёва

«14» сентября 2026 г.

Приложение к рабочей программе учебной дисциплины

**ФОНД ОЦЕНОЧНЫХ СРЕДСТВ  
для проведения текущего контроля и промежуточной аттестации  
обучающихся по дисциплине  
«Структуры данных и алгоритмы»**

**По направлению подготовки** 09.03.04 Программная инженерия

**Профиль подготовки** Программное обеспечение систем и комплексов

**Квалификация выпускника** – бакалавр

**Форма обучения** очная

**Курс** ОФО – 1 курс

Разработчик  
канд. физ.-мат. наук, доцент кафедры  
информационных образовательных  
технологий и систем Швыров В.В.

заведующий кафедрой  
информационных образовательных  
технологий и систем  
Д.А. Капустин

Протокол

от «14» сентября 2026 г. № 11

Луганск, 2026

# 1. ПАСПОРТ ФОНДА ОЦЕНОЧНЫХ СРЕДСТВ

## 1.1 Область применения

Фонд оценочных средств (ФОС) – неотъемлемая часть рабочей программы дисциплины «Структуры данных и алгоритмы» и предназначен для контроля и оценки образовательных достижений студентов, освоивших программу дисциплины.

## 1.2. Цели и задачи фонда оценочных средств

Цель ФОС – установить соответствие уровня подготовки обучающегося требованиям ФГОС ВО бакалавриат по направлению подготовки 09.03.04 Программная инженерия, утвержденным приказом Министерства образования и науки Российской Федерации от 19 сентября 2017 г. № 920 и Профессиональным стандартом, утвержденным Приказом Министерства труда и социальной защиты Российской Федерации «Об утверждении профессионального стандарта 06.001 «Программист» от 20.07.2022 № 424н.

## 1.3. Перечень компетенций, формируемых в процессе освоения основной образовательной программы

Процесс освоения дисциплины направлен на формирование следующих компетенций и индикаторов их достижения:

Код по ФГОС ВО	Индикатор достижения
Общепрофессиональные	
ОПК-7. Способен применять в практической деятельности основные концепции, принципы, теории и факты, связанные с информатикой	ОПК-7.1. Знает основные языки программирования и работы с базами данных, операционные системы и оболочки, современные программные среды разработки информационных систем и технологий.
	ОПК-7.2. Умеет применять языки программирования и работы с базами данных, современные программные среды разработки информационных систем и технологий для автоматизации бизнес-процессов, решения прикладных задач различных классов, ведения баз данных и информационных хранилищ.
	ОПК-7.3. Владеет навыками программирования, отладки и тестирования прототипов программно-технических комплексов задач.

## 1.4. Этапы формирования компетенций и средства оценивания уровня их сформированности

Этапы формирования компетенций	Компетенции	Контрольно-оценочные средства / способ оценивания
Тема 1. Введение. Оценка сложности алгоритмов.	ОПК-7	Устный опрос

Тема 2. Стандартные и пользовательские типы данных	ОПК-7	Устный опрос. Защита лабораторных работ
Тема 3. Алгоритмы работы с массивами	ОПК-7	Устный опрос. Защита лабораторных работ
Тема 3. Поиск и рекурсия. Сортировки.	ОПК-7	Устный опрос
Тема 4. Динамическая память.	ОПК-7	Устный опрос. Защита лабораторных работ
Тема 5. Динамические структуры.	ОПК-7	Устный опрос. Защита лабораторных работ
Тема 6. Работа с библиотеками: list, queue, dequeue, stack.	ОПК-7	Устный опрос. Защита лабораторных работ
Тема 7. Графы ориентированные. Поиск короткого пути – алгоритм Дейкстры, алгоритм Флойда.	ОПК-7	Устный опрос. Защита лабораторных работ
Тема 8. Графы неориентированные. Алгоритм Прима. Поиск в глубину, поиск в ширину.	ОПК-7	Устный опрос. Защита лабораторных работ
<b>Текущая аттестация</b>	ОПК-7	Индивидуальное задание
<b>Промежуточная аттестация</b>	ОПК-7	Экзамен

### 1.5. Описание показателей формирования компетенций

Код по ФГОС ВО	Результаты обучения по дисциплине
ОПК-7	<p>знать: основные этапы компьютерного решения задач; понятие алгоритма и структуры управления; традиционные структуры данных; основные требования методологии структурного программирования, как технологической основы разработки качественных программных компонентов; понятие статических и динамических данных; примеры базовых структур, данных; подходы процедурного, модульного программирования, реализацию вызова процедур в языках с блочной структурой, рекурсию, математический аппарат, необходимый для оценивания времени выполнения алгоритма.</p> <p>уметь: применять требования методологии структурного программирования при проектировании информационных моделей; разрабатывать и записывать на языке программирования высокого уровня алгоритмы решения классических задач программирования; выбирать оптимальную структуру для представления данных.</p> <p>владеть: методами создания алгоритмов для решения поставленной задачи; принципами структурного программирования; навыками практического программирования конкретных задач в определенной языковой среде.</p>

### 1.6. Критерии оценивания компетенций на разных этапах их формирования

Вид учебной работы	Количество баллов		
	ОФО	О-ЗФО	ЗФО
Устные ответы на семинарских занятиях			
Выполнение и защита практических / лабораторных работ	45		

Самостоятельная работа	<b>10</b>		
Иные виды учебной работы (подготовка презентации, написание реферата, решение задач и др.)	<b>15</b>		
Зачет	<b>30</b>		
<b>Всего</b>	<b>100</b>		

### Накопительная система оценивания по 100-балльной шкале

Четырехбалльная система оценивания экзамена	100-балльная шкала	Буквенная шкала, соответствующая 100-балльной шкале	Система оценивания зачета
Отлично	<b>90-100</b>	<b>А</b> – отлично – теоретическое содержание курса освоено полностью, без пробелов; необходимые практические навыки работы с освоенным материалом сформированы; все предусмотренные программой обучения учебные задания выполнены, качество их выполнения оценено числом баллов, близким к максимальному	Зачтено
Хорошо	<b>83-89</b>	<b>В</b> – очень хорошо – теоретическое содержание курса освоено полностью, без пробелов; необходимые практические навыки работы с освоенным материалом в основном сформированы; все предусмотренные программой обучения учебные задания выполнены, качество выполнения большинства из них оценено числом баллов, близким к максимальному	
Хорошо	<b>75-82</b>	<b>С</b> – хорошо – теоретическое содержание курса освоено полностью; некоторые практические навыки работы с освоенным материалом сформированы недостаточно; все предусмотренные программой обучения учебные задания выполнены, качество выполнения ни одного из них не оценено минимальным числом баллов, некоторые виды заданий выполнены с ошибками	
Удовлетворительно	<b>63-74</b>	<b>Д</b> – удовлетворительно – теоретическое содержание курса освоено частично, но пробелы не носят существенного характера; необходимые практические навыки работы с освоенным материалом в основном сформированы; большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки	
Удовлетворительно	<b>50-62</b>	<b>Е</b> – посредственно – теоретическое содержание курса освоено частично; некоторые практические навыки работы не сформированы, многие предусмотренные учебной программой обучения учебные задания не выполнены либо качество выполненных некоторых из них оценено числом баллов, близким к минимальному	
Неудовлетворительно	<b>21-49</b>	<b>FX</b> – неудовлетворительно – теоретическое содержание курса освоено частично; необходимые практические навыки работы с освоенным материалом не сформированы; большинство предусмотренных учебной программой обучения учебных заданий не выполнено либо качество их выполнения оценено числом баллов, близким к минимальному; при дополнительно самостоятельной работе над материалом курса возможно повышение качества выполнения учебных заданий	Не зачтено
Неудовлетворительно	<b>0-20</b>	<b>F</b> – неудовлетворительно – теоретическое содержание курса не освоено; необходимые	

		практические навыки работы не сформированы; все выполненные учебные задания содержат грубые ошибки; дополнительная самостоятельная работа над материалом курса не приведет к какому-либо значимому повышению качества выполнения учебных заданий	
--	--	---	--

## 2. КОНТРОЛЬНО-ОЦЕНОЧНЫЕ СРЕДСТВА

### 2.1. Оценочные средства текущего контроля

1. Алгоритм и его свойства
2. Виды алгоритмов
3. Оценка сложности алгоритма
4. Типы данных
5. Одномерные массивы. Назначение
6. Двумерные массивы. Назначение
7. Строки. Функции работы со строками.
8. Массивы строк
9. Алгоритмы работы с массивами
10. Структуры. Назначение, синтаксис.
11. Массивы структур.
12. Перечисления
13. Объединения
14. Рекурсия
15. Алгоритмы поиска: линейный, бинарный
16. Сортировки. Методы вставки, выбором, быстрая и др
17. Одномерные динамические массивы
18. Двумерные динамические массивы
19. Динамические структуры: стеки. Добавление, удаление узла, поиск узла.
20. Динамические структуры: очереди. Добавление, удаление узла, поиск узла.
21. Динамические структуры: однонаправленные и двунаправленные списки. Добавление, удаление узла, поиск узла.
22. Деревья бинарные. Добавление, удаление узла, поиск узла.
23. Графы ориентированные и неориентированные.
24. Графы ориентированные. Поиск короткого пути – алгоритм Дейкстры
25. Графы ориентированные. Поиск короткого пути – алгоритм Флойда
26. Графы неориентированные. Алгоритм Прима.
27. Поиск в глубину
28. Поиск в ширину.

Типовые задания по дисциплине

Задание 1.

В каждом четном варианте организовать сортировку данных по убыванию, а в нечетном по возрастанию

№	
1	Создать динамический массив, размер запросить у пользователя. Сделать проверку на возможность выделения заданного количества памяти. Заполнить случайными числами в диапазоне от 0 до 5. Подсчитать количество единиц в массиве.
2	Создать динамический массив, размер запросить у пользователя. Сделать проверку на возможность выделения заданного количества памяти. В массиве найти номер минимального элемента
3	Создать динамический массив, размер запросить у пользователя. Сделать проверку на возможность выделения заданного количества памяти. Вывести максимальный элемент
4	Создать динамический массив, размер запросить у пользователя. Сделать проверку на возможность выделения заданного количества памяти. В массиве подсчитать количество элементов меньших 10

5	Создать динамический массив, размер запросить у пользователя. Сделать проверку на возможность выделения заданного количества памяти. В массиве подсчитать количество элементов больших 100
---	--

#### Задание 2

1. Даны матрица  $A(n \times (n+1))$  и два одномерных массива  $X=(x_1, \dots, x_{n+1})$  и  $Y=(y_1, \dots, y_{n+1})$ , а также натуральные числа  $p, q$ . Образовать новую матрицу размера  $(n+1) \times (n+2)$  вставкой после строки с номером  $p$  матрицы  $A$  новой строки с элементами  $x_1, x_2, \dots, x_{n+1}$  и последующей вставкой после столбца с номером  $q$  нового столбца с элементами  $y_1, y_2, \dots, y_{n+1}$ .
2. Даны массив  $A=(a_1, a_2, \dots, a_{10})$  и матрица  $B(n \times n)$ . Заменить нулями в матрице те элементы с четной суммой индексов, для которых имеются равные среди элементов массива  $A$ .
3. Даны массив  $A=(a_1, a_2, \dots, a_{10})$  и матрица  $B(n \times n)$ . Элементы первой строки матрицы упорядочены по возрастанию. Получить новую матрицу размера  $n \times (n+1)$ , вставив в исходную матрицу новый столбец с элементами массива  $A$  так, чтобы упорядоченность первой строки матрицы не нарушилась.
4. Дана матрица  $A(n \times m)$ . Получить матрицу, получающуюся из данной: перестановкой столбцов: первого с последним, второго с предпоследним и т. д.
5. Дана матрица  $A(n \times m)$  и целые числа  $p$  и  $q$ . Преобразовать матрицу  $A$  так, чтобы строка с исходным номером  $p$  непосредственно следовала за строкой с исходным номером  $q$ , сохранив порядок следования остальных строк.

#### Задание 3

1. Дана строка, состоящая из символов латинского алфавита, разделенных пробелами (одним или несколькими). Преобразовать каждое слово в строке, удалив из него все вхождения первой буквы этого слова (количество пробелов между словами не изменять).
2. Дана строка, состоящая из символов латинского алфавита, разделенных пробелами (одним или несколькими). Определить количество слов, которые начинаются и заканчиваются одной и той же буквой.
3. Дана строка-предложение из символов латинского алфавита. Вывести самое короткое слово в предложении (если таких слов несколько, то вывести первое из них).
4. Дана строка, состоящая из символов латинского алфавита, разделенных пробелами (одним или несколькими). Определить количество слов, которые содержат ровно три буквы 'A'.
5. Дана строка, состоящая из символов латинского алфавита, разделенных пробелами (одним или несколькими). Определить длину самого длинного слова.

#### Задание 4

Решите задачи данной группы, оформив решение в виде функций генерации, вывода и обработки массивов. Предусмотрите в функции генерации массива ввод границ диапазона случайных чисел. Число элементов в массиве задается пользователем.

1. Дан массив, состоящий из  $n$  элементов. Переставить в обратном порядке элементы массива, расположенные между его минимальным и максимальным элементами.
2. Дан массив, состоящий из  $n$  элементов. Назовем *серией* группу подряд идущих одинаковых элементов, а *длиной серии* – количество этих элементов (длина серии

может быть равна 1). Заменить каждую серию, длина которой больше  $k$ , на один наименьший элемент массива. Если таких серий нет, то массив оставить без изменений.

3. Дан массив, состоящий из  $n$  элементов. Назовем *серией* группу подряд идущих одинаковых элементов, а *длиной серии* – количество этих элементов (длина серии может быть равна 1). Преобразовать массив, увеличив серию на 1.
4. Дан массив, состоящий из  $n$  элементов. Назовем *серией* группу подряд идущих одинаковых элементов, а *длиной серии* – количество этих элементов (длина серии может быть равна 1). Вставить перед каждой серией минимальный элемент массива.
5. Дан массив, состоящий из  $n$  элементов. Назовем *серией* группу подряд идущих одинаковых элементов, а *длиной серии* – количество этих элементов (длина серии может быть равна 1). Поменять местами наибольшую первую и  $k$ -ю серии массива. Если таких серий в массиве меньше  $k$ , то вывести массив без изменений.

#### Задание 5

Задания на одномерный вектор

1. Скорректировать массив  $A=(a_1, a_2, \dots, a_n)$ , переписав в начало массива группу, содержащую наибольшее число подряд идущих положительных элементов. Элементы массива вводить с клавиатуры.
2. В массиве  $A=(a_1, a_2, \dots, a_n)$  все элементы, равные нулю, поставить сразу после максимального элемента данного массива. Элементы массива вводить с клавиатуры.
3. В массиве  $A=(a_1, a_2, \dots, a_n)$  все отрицательные элементы отправить в «хвост» массива.
4. В массиве  $A=(a_1, a_2, \dots, a_n)$  удалить последнюю группу положительных элементов. Группой называется подряд идущие элементы одного знака, число которых больше или равно 2.
5. В массиве  $A=(a_1, a_2, \dots, a_n)$  все положительные элементы, стоящие перед минимальным положительным элементом, переслать в «хвост» массива.

#### Задание 6

1. Тип данных, хранящихся в списке: строка переменной длины. Реализовать следующие действия:
  - а) подсчет строк в списке;
  - б) получение  $j$ -го символа  $i$ -й строки (если такого не существует, вернуть 0);
2. Тип данных, хранящихся в списке: строка переменной длины. Реализовать следующие действия:
  - а) добавление после  $i$ -й строки копии  $j$ -й строки;
  - б) удаление  $j$ -й строки из списка;
3. Тип данных, хранящихся в списке: число с плавающей точкой. Реализовать следующие действия:
  - а) проверка пуст ли список;
  - б) подсчет среднего арифметического элементов непустого списка;
4. Тип данных, хранящихся в списке: символ. Реализовать следующие действия:
  - а) проверка пуст ли список;
  - б) замена в списке всех вхождений  $E_1$  на  $E_2$ ;
5. Тип данных, хранящихся в списке: строка из десяти символов. Реализовать следующие действия:
  - а) подсчет количества слов, совпадающих с головой списка;
  - б) поиск в списке заданного слова (вернуть номер элемента списка).

#### Задание 7

Используя стек, вычислить следующее выражение:

№	Выражение

1.	$x^2 + 6xy - 8c$
2.	$12x + 6(x + 2) - 2(c - 1)$
3.	$2x^3 + 2y - 8c + x$
4.	$x^2 + 6xy - 2c$
5.	$x^2 + 2xy + c^3$

#### Задание 8

Организовать работу со стеком или очередью, где каждый элемент структура, организовать поиск по значимому полю (например название игр, фильмов, книг и т.д.)

№	Задание
1	Описать структуру игры, содержащую информацию об играх, которые содержатся в вашей игротке: название, жанр, производитель, дата приобретения. Создать стек таких структур и заполнить его. Вывести в файл все непройденные игры.
2	Создать структуру телефона, которая содержит информацию о мобильных телефонах, продаваемых магазином: производитель, модель, цвет, полифония, наличие фото-видеокамеры и т. п. Создать очередь таких структур и заполнить его. Вывести в файл все телефоны, имеющие фотокамеру.
3	Описать структуру книги, содержащую информацию о книгах, хранимых в библиотеке: название, автор, жанр. Создать стек таких структур и заполнить его. Вывести в файл книги, которые находятся в наличии.
4	Описать структуру музыка, которая содержит информацию о музыкальных дисках, которые продаются в магазине: название альбома, исполнитель, жанр, год создания альбома, цена диска. Создать очередь таких структур и заполнить его. Вывести в файл диски по жанру исполнения: у пользователя запрашивается жанр и выводятся все диски, соответствующие условию.
5	Описать структуру видео, которая содержит информацию о фильмах, показываемых в кинотеатре: название, режиссер, жанр, главные актеры, дата показа. Создать стек таких структур и заполнить его. Вывести в файл все фильмы которые показывались в текущем месяце.

## 2.2 Оценочные средства для промежуточной аттестации

1. Оценка сложности алгоритма
2. Что такое алгоритм в программировании и какие типы алгоритмы вы знаете?
3. Какие свойства алгоритма вы знаете?
4. Что такое линейный алгоритм и как он выполняется?
5. Что такое условный алгоритм и как он выполняется?
6. Что такое циклический алгоритм и как он выполняется?
7. Какие алгоритмы сортировки существуют в C++?
8. Как работает алгоритм сортировки пузырьком в C++?
9. Как работает алгоритм сортировки выбором в C++?
10. Как работает алгоритм сортировки вставками в C++?
11. Какие алгоритмы поиска существуют в C++?
12. Как работает алгоритм линейного поиска в C++?
13. Как работает алгоритм бинарного поиска в C++?

14. Что такое блок-схема (flowchart) в программировании?
15. Какие элементы используются в блок-схемах и как они представляются?
16. Какие типы блоков (элементов) могут быть использованы в блок-схемах?
17. Какие символы используются для представления различных типов блоков в блок-схемах?
18. Какие виды соединений между блоками могут быть использованы в блок-схемах?
19. Какие основные правила следует учитывать при создании блок-схем?
20. Какие преимущества предоставляют блок-схемы при разработке программ?
21. Какие основные шаги нужно выполнить для создания блок-схемы?
22. Как читать и интерпретировать блок-схему?
23. Какие инструменты и программы можно использовать для создания блок-схем?
24. Какие типичные ошибки можно совершить при создании блок-схем и как их избежать?
25. Как блок-схемы связаны с процессом программирования и разработки алгоритмов?
26. Как использование блок-схем может помочь в обучении программированию и алгоритмическому мышлению?
27. Какие типы данных существуют?
28. Как происходит преобразование типов?
29. Что такое переменная в C++?
30. Что такое область видимости?
31. Отличие глобальных и локальных переменных?
32. Что такое одномерный массив в C++ и как его объявить?
33. Как объявить и инициализировать одномерный массив с элементами типа `int` в C++?
34. Как получить доступ к элементу массива в C++?
35. Как объявить и инициализировать двумерный массив (матрицу) в C++?
36. Как перебрать все элементы двумерного массива с помощью вложенных циклов в C++?
37. Что такое структура (структура данных) в C++ и как она объявляется?
38. Как объявить переменную типа структуры и получить доступ к ее членам в C++?
39. Что такое перечисление (enum) в C++ и как оно объявляется?
40. Как использовать перечисление в C++?
41. Что такое объединение (union) в C++ и как оно объявляется?
42. Как использовать объединение в C++?
43. Как объявить и инициализировать строку (массив символов) в C++?
44. Зачем нужна библиотека `string`?
45. Как получить длину строки в C++?
46. Как сконкатенировать две строки в C++?
47. Как скопировать одну строку в другую в C++?
48. Как проверить, содержит ли строка определенный подстроку в C++?
49. Как найти первое вхождение символа или подстроки в строке в C++?
50. Как осуществить поиск всех вхождений символа или подстроки в строке в C++?
51. Что такое динамический массив (dynamic array) в C++ и как он создается?
52. Как добавить элемент в динамический массив в C++?
53. Как удалить элемент из динамического массива в C++?
54. Как изменить размер динамического массива в C++?
55. Что такое вектор (vector) в C++ и как он отличается от обычного массива?
56. Как объявить и инициализировать вектор в C++?
57. Как добавить элемент в конец вектора в C++?
58. Как получить доступ к элементам вектора в C++?

59. Как удалить элемент из вектора в C++?
60. Что такое множество (set) в C++ и как оно работает?
61. Как объявить и инициализировать множество в C++?
62. Как добавить элемент в множество в C++?
63. Как удалить элемент из множества в C++?
64. Что такое ассоциативный массив (map) в C++ и как он работает?
65. Как объявить и инициализировать ассоциативный массив в C++?
66. Как добавить элемент в ассоциативный массив в C++?
67. Как получить доступ к элементам ассоциативного массива в C++?
68. Что такое рекурсия и как она используется в программировании?
69. Рекурсия что это и зачем?
70. Одномерные динамические массивы
71. Двумерные динамические массивы
72. Динамические структуры: стеки. Добавление, удаление узла, поиск узла.
73. Динамические структуры: очереди. Добавление, удаление узла, поиск узла.
74. Динамические структуры: однонаправленные и двунаправленные списки. Добавление, удаление узла, поиск узла.
75. Динамические структуры: однонаправленные и двунаправленные кольцевые списки. Добавление, удаление узла, поиск узла.
76. Динамические структуры: деки. Добавление, удаление узла, поиск узла.
77. Деревья бинарные. Добавление, удаление узла, поиск узла.
78. Зачем нужна библиотека list в C++ и как она работает?
79. Как объявить и инициализировать список с помощью list в C++?
80. Как добавить элемент в список с помощью list в C++?
81. Как удалить элемент из списка с помощью list в C++?
82. Зачем нужна библиотека stack в C++ и как она работает?
83. Как объявить и инициализировать стек с помощью stack в C++?
84. Как добавить элемент в стек с помощью stack в C++?
85. Как удалить элемент из стека с помощью stack в C++?
86. Зачем нужна библиотека queue в C++ и как она работает?
87. Как объявить и инициализировать очередь с помощью queue в C++?
88. Как добавить элемент в очередь с помощью queue в C++?
89. Как удалить элемент из очереди с помощью queue в C++?
90. Зачем нужна библиотека deque в C++ и как она работает?
91. Как объявить и инициализировать дек с помощью deque в C++?
92. Как добавить элемент в дек с помощью deque в C++?
93. Как удалить элемент из дека с помощью deque в C++?
94. Графы ориентированные и неориентированные.
95. Графы ориентированные. Поиск короткого пути – алгоритм Дейкстры
96. Графы ориентированные. Поиск короткого пути – алгоритм Флойда
97. Нахождения минимального остовного дерева. Алгоритм Прима.
98. Поиск в глубину назначение и применение
99. Поиск в ширину назначение и применение
100. Нахождения минимального остовного дерева. Алгоритм Крускала
101. Каким образом можно работать с файлами в C++?
102. Зачем нужна библиотека fstream?
103. Как записывать данные в файл?
104. Как считывать данные из файла?
105. Как открыть и закрыть файл?

## Типовые задания к экзамену

1. Написать программу, используя `<list>`: Тип данных, хранящихся в списке: строка переменной длины. Реализовать следующие действия:
  - а) обмен содержимым двух заданных элементов списка;
  - б) поиск позиции самой длинной строки.
2. Написать программу, используя `<stack>`: Тип данных, хранящихся в списке: строка переменной длины. Реализовать следующие действия:
  - а) подсчет количества элементов в списке;
  - б) создание нового списка типа `<stack>`: из всех элементов списка, которые завершаются точкой
3. Написать программу, используя библиотеку `<set>`: Тип данных, хранящихся в списке: строка переменной длины. Реализовать следующие действия:
  - а) создание нового списка из строк, не содержащих заданную подстроку;
  - б) проверка, есть ли в списке хотя бы два одинаковых элемента.
4. Написать программу, используя библиотеку `<map>`: Тип данных, хранящихся в списке: строка переменной длины (ключ) и число. Реализовать следующие действия:
  - а) удаление по заданному ключу;
  - б) вывод всех элементов списка, в которых присутствует заданная подстрока в ключе
5. Написать программу, используя библиотеку `<set>`: Тип данных, хранящихся в списке: число с плавающей точкой. Реализовать следующие действия:
  - а) вставка нового элемента перед заданным;
  - б) проверка, есть ли в списке хотя бы два одинаковых элемента
6. Написать программу, используя библиотеку `<vector>`: Тип данных, хранящихся в списке: число с плавающей точкой. Реализовать следующие действия:
  - а) создание нового списка, состоящего из обратных значений элементов исходного списка;
  - б) удвоение каждого вхождения заданного элемента в список.
7. Написать программу, используя `<queue>` Тип данных, хранящихся в списке: число с плавающей точкой. Реализовать следующие действия:
  - а) определение, входит ли элемент в список;
  - б) поиск максимального элемента списка
8. Написать программу, используя библиотеку `<map>`: Тип данных, хранящихся в списке: слово (ключ), символ. Реализовать следующие действия:
  - а) поиск по указанному ключу;
  - б) замена группы, идущих подряд равных символов на один.
9. Написать программу, используя `<deque>`: Тип данных, хранящихся в списке: число с плавающей точкой. Реализовать следующие действия:
  - а) деление всех элементов списка на заданное значение;
  - б) поиск заданного элемента.
10. Написать программу, используя библиотеку `<multimap>`: Тип данных, хранящихся в списке: символ (ключ) и целое число. Реализовать следующие действия:
  - а) проверка пуст ли список;
  - б) замена в списке всех значений заданного ключа на удвоенное
11. Написать программу, используя библиотеку `<vector>`: Тип данных, хранящихся в списке: число с плавающей точкой. Реализовать следующие действия:
  - а) отсортировать список;
  - б) подсчет среднего арифметического элементов непустого списка