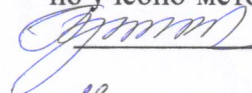


**ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ НОВОСИБИРСКОЙ ОБЛАСТИ
«НОВОСИБИРСКИЙ ЭЛЕКТРОМЕХАНИЧЕСКИЙ КОЛЛЕДЖ»
(ГБПОУ НСО «Новосибирский электромеханический колледж»)**

УТВЕРЖДАЮ

Заместитель директора

по учебно-методической работе

 Е.О. Кравченко

«16» июня 2025 г.

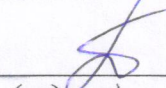
**ФОНД
ОЦЕНОЧНЫХ СРЕДСТВ УЧЕБНОЙ ДИСЦИПЛИНЫ**

ОП.04 ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ

Специальность 09.02.07. Информационные системы и программирование

Рассмотрено
на заседании кафедры общеобразовательных дисциплин
Протокол № 1 от «16» июня 2025 г.

Председатель кафедры


(подпись)

Белимова В.Г.
(Ф.И.О.)

2025 г.

Фонд оценочных средств учебного предмета разработана на основе Федерального государственного образовательного стандарта (далее – ФГОС) по профессии 09.02.07 Информационные системы и программирование

Организация-разработчик: ГБПОУ НСО «НЭК»

Разработчик:

Комарова А.В., преподаватель высшей квалификационной категории

Согласовано:
Методист



Рахметова Т.Ю.

«18» июня 2021г.

Оглавление

1. ПЕРЕЧЕНЬ КОМПЕТЕНЦИЙ
2. ПАСПОРТ ФОНДА ОЦЕНОЧНЫХ СРЕДСТВ
 - 2.1. Структура фонда оценочных средств для текущего контроля и промежуточной аттестации
 - 2.2. КРИТЕРИИ ОЦЕНИВАНИЯ КОМПЕТЕНЦИЙ НА РАЗЛИЧНЫХ ЭТАПАХ ИХ ФОРМИРОВАНИЯ ПО ВИДАМ ОЦЕНОЧНЫХ СРЕДСТВ ПЕРЕЧЕНЬ ОЦЕНОЧНЫХ СРЕДСТВ
 - 2.3. ТИПОВЫЕ КОНТРОЛЬНЫЕ ЗАДАНИЯ
3. ЭКЗАМЕНАЦИОННЫЕ БИЛЕТЫ
 - 3.2. Эталонные ответы экзаменационных билетов. Практическая часть
 - 3.3. Эталонные ответы экзаменационных билетов. Теоретическая часть
4. ПЕРЕЧЕНЬ МАТЕРИАЛОВ, ОБОРУДОВАНИЯ И ИНФОРМАЦИОННЫХ ИСТОЧНИКОВ
5. МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ, ОПРЕДЕЛЯЮЩИЕ ПРОЦЕДУРЫ ОЦЕНИВАНИЯ ЗНАНИЙ

1. ПЕРЕЧЕНЬ КОМПЕТЕНЦИЙ

1.1 Перечень общих компетенций

Код	Наименование общих компетенций
ОК 1	Выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам
ОК 2	Использовать современные средства поиска, анализа и интерпретации информации, и информационные технологии для выполнения задач профессиональной деятельности
ОК 4	Эффективно взаимодействовать и работать в коллективе и команде
ОК 5	Осуществлять устную и письменную коммуникацию на государственном языке с учетом особенностей социального и культурного контекста
ОК 9	Пользоваться профессиональной документацией на государственном и иностранном языках

1.2 Перечень профессиональных компетенций

Код	Наименование видов деятельности и профессиональных компетенций
ПК 1.1	Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием
ПК 1.2	Разрабатывать программные модули в соответствии с техническим заданием
ПК 1.3	Выполнять отладку программных модулей с использованием специализированных программных средств
ПК 1.4	Выполнять тестирование программных модулей
ПК 1.5	Осуществлять рефакторинг и оптимизацию программного кода
ПК 2.4	Осуществлять разработку тестовых наборов и тестовых сценариев для программного обеспечения
ПК 2.5	Производить инспектирование компонент программного обеспечения на предмет соответствия стандартам кодирования

В результате освоения учебной дисциплины студент должен:

Код ПК, ОК	Умения	Знания
ОК 1 ОК 2 ОК 4 ОК 5 ОК 9 ПК 1.1- ПК 1.5 ПК 2.4, 2.5	Разрабатывать алгоритмы для конкретных задач. Использовать программы для графического отображения алгоритмов. Определять сложность работы алгоритмов. Работать в среде программирования. Реализовывать построенные алгоритмы в виде программ на конкретном языке программирования. Оформлять код программы в соответствии со стандартом кодирования. Выполнять проверку, отладку кода программы.	Понятие алгоритмизации, свойства алгоритмов, общие принципы построения алгоритмов, основные алгоритмические конструкции. Эволюцию языков программирования, их классификацию, понятие системы программирования. Основные элементы языка, структуру программы, операторы и операции, управляющие структуры, структуры данных, файлы, классы памяти. Подпрограммы, составление библиотек подпрограмм. Объектно-ориентированную модель программирования, основные принципы объектно-ориентированного программирования на примере

		алгоритмического языка: понятие классов и объектов, их свойств и методов, инкапсуляция и полиморфизма, наследования и переопределения.
--	--	--

Матрица соответствия заданий и компетенций

Компетенция	Раздел 1	Раздел 2	Раздел 3	Раздел 4	Раздел 5
ОК 1	✓	✓	✓	✓	✓
ОК 2	✓	✓	✓	✓	✓
ОК 4	✓	✓	✓	✓	✓
ОК 5	✓	✓	✓	✓	✓
ОК 9	✓	✓	✓	✓	✓
ПК 1.1		✓	✓	✓	✓
ПК 1.2	✓	✓	✓	✓	✓
ПК 1.3	✓		✓		✓
ПК 1.4		✓	✓	✓	✓
ПК 1.5			✓	✓	✓
ПК 2.4		✓	✓	✓	

2. ПАСПОРТ ФОНДА ОЦЕНОЧНЫХ СРЕДСТВ

2.1. Структура фонда оценочных средств для текущего контроля и промежуточной аттестации

№ п/п	Результаты освоения УД/М (код и наименование)	Код контролируемой компетенции	Наименование оценочного средства	
			Текущий контроль	Промежуточная аттестация
1	Входное тестирование		Контрольная работа №1	
2	Раздел 1. <i>Введение в программирование</i>	ОК 1, ОК 2, ОК 4, ОК 5, ОК 9 ПК 1.1- ПК 1.5 ПК 2.4, 2.5	Контрольная работа №2	Вопросы и задания диф.зачета
3	Раздел 2. Операторы языка программирования	ОК 1, ОК 2, ОК 4, ОК 5, ОК 9 ПК 1.1- ПК 1.5 ПК 2.4, 2.5	Контрольная работа №3	Вопросы и задания диф.зачета
4	Раздел 3. Структурное и модульное программирование	ОК 1, ОК 2, ОК 4, ОК 5, ОК 9 ПК 1.1- ПК 1.5 ПК 2.4, 2.5	Контрольная работа №4	Вопросы и задания диф.зачета
5	Раздел 4. Основные конструкции языков программирования	ОК 1, ОК 2, ОК 4, ОК 5, ОК 9 ПК 1.1- ПК 1.5 ПК 2.4, 2.5	Контрольная работа №5	Вопросы и задания диф.зачета
6	Раздел 5. Объектно- ориентированное программирование	ОК 1, ОК 2, ОК 4, ОК 5, ОК 9 ПК 1.1- ПК 1.5 ПК 2.4, 2.5	Контрольная работа №6	Вопросы и задания диф.зачета
7	Экзаменационные вопросы	ОК 1, ОК 2, ОК 4, ОК 5, ОК 9 ПК 1.1- ПК 1.5 ПК 2.4, 2.5	Экзаменационные вопросы	

**2.2. КРИТЕРИИ ОЦЕНИВАНИЯ КОМПЕТЕНЦИЙ НА РАЗЛИЧНЫХ ЭТАПАХ ИХ
ФОРМИРОВАНИЯ ПО ВИДАМ ОЦЕНОЧНЫХ СРЕДСТВ ПЕРЕЧЕНЬ
ОЦЕНОЧНЫХ СРЕДСТВ**

№ п/п	Наименование оценочного средства	Характеристика оценочного средства	Представление оценочного средства в фонде
УСТНЫЕ ОЦЕНОЧНЫЕ СРЕДСТВА			
1	собеседование, устный опрос	Средство контроля, организованное как специальная беседа преподавателя с обучающимся на темы, связанные с изучаемой дисциплиной, и рассчитанное на выяснение объема знаний, обучающегося по определенному разделу, теме, проблеме и т.п.	Вопросы по темам/разделам дисциплины
2	Коллоквиум	Средство контроля усвоения учебного материала темы, раздела или разделов дисциплины, организованное как учебное занятие в виде собеседования преподавателя с обучающимися.	Вопросы по темам/разделам дисциплины
3.	Доклад, сообщение	Продукт самостоятельной работы обучающегося, представляющий собой публичное выступление по представлению полученных результатов решения определенной учебно-практической, учебно-исследовательской или научной темы	Темы докладов, сообщений
ПИСЬМЕННЫЕ ОЦЕНОЧНЫЕ СРЕДСТВА			
4	Реферат	Продукт самостоятельной работы аспиранта, представляющий собой краткое изложение в письменном виде полученных результатов теоретического анализа определенной научной (учебно-исследовательской) темы, где автор раскрывает суть исследуемой проблемы, приводит различные точки зрения, а также собственные взгляды на нее.	Темы рефератов
5	Тест	Система стандартизированных заданий, позволяющая автоматизировать процедуру измерения уровня знаний и умений обучающегося.	Фонд тестовых заданий
6	Проект	Конечный продукт, получаемый в результате планирования и выполнения комплекса учебных и исследовательских заданий. Позволяет оценить умения	Темы групповых и/или индивидуальных проектов

		обучающихся самостоятельно конструировать свои знания в процессе решения практических задач и проблем, ориентироваться в информационном пространстве и уровень сформированности аналитических, исследовательских навыков, навыков практического и творческого мышления. Может выполняться в индивидуальном порядке или группой обучающихся.	
7	Контрольная работа	Средство проверки умений применять полученные знания для решения задач определенного типа по теме или разделу	комплект контрольных заданий по вариантам
8	Курсовая работа	Продукт самостоятельной работы обучающегося, представляющий собой краткое изложение в письменном виде полученных результатов теоретического анализа определенной учебно-исследовательской темы, где автор раскрывает суть исследуемой проблемы, приводит различные точки зрения, а также собственные взгляды на нее.	Темы курсовых работ
	Курсовой проект	Курсовым проектом является письменная работа, выполняющаяся на протяжении семестра и содержащая анализ варианта экономического или инженерного решения по теме, заданной в заглавии самого курсового проекта. Любой курсовой проект является строго индивидуальным и ориентированным на развитие у студента профессиональных навыков, а также умению творчески подходить к решению практических задач, которые относятся к выбранному направлению подготовки. Курсовой проект обязательно должен состоять из расчетной (графической) и текстовой части. В текстовую часть обязательно входит объяснительная записка, которая заполняется не только теоретическими подсчётами, но и проведёнными вычислениями и расчётами. Графическая часть включает в себя схемы, таблицы и чертежи.	Темы курсовых проектов

9	Лабораторная работа	Средство для закрепления и практического освоения материала по определенному разделу	Комплект лабораторных заданий
10	Задача	Это средство, раскрытия связи между данными и искомым, заданные условием задачи, на основе чего надо выбрать, а затем выполнить действия, в том числе арифметические, и дать ответ на вопрос задачи.	задания по задачам
11	Расчетно-графическая работа	Средство проверки умений применять полученные знания по заранее определенной тематике для решения задач или заданий по модулю или дисциплине в целом.	комплект заданий для выполнения расчетно-графической работы

А) КРИТЕРИИ И ШКАЛА ОЦЕНИВАНИЯ ОТВЕТОВ НА УСТНЫЕ ВОПРОСЫ

№ п/п	критерии оценивания	количество баллов	оценка/зачет
1.	1) полно и аргументированно отвечает по содержанию задания; 2) обнаруживает понимание материала, может обосновать свои суждения, применить знания на практике, привести необходимые примеры не только по учебнику, но и самостоятельно составленные; 3) излагает материал последовательно и правильно.	10	отлично
2.	студент дает ответ, удовлетворяющий тем же требованиям, что и для оценки «5», но допускает 1-2 ошибки, которые сам же исправляет.	8	хорошо
3.	ставится, если студент обнаруживает знание и понимание основных положений данного задания, но: 1) излагает материал неполно и допускает неточности в определении понятий или формулировке правил; 2) не умеет достаточно глубоко и доказательно обосновать свои суждения и привести свои примеры; 3) излагает материал непоследовательно и допускает ошибки.	5	удовлетворительно

4.	студент обнаруживает незнание ответа на соответствующее задание, допускает ошибки в формулировке определений и правил, искажающие их смысл, беспорядочно и неуверенно излагает материал; отмечаются такие недостатки в подготовке студента, которые являются серьезным препятствием к успешному овладению последующим материалом.	0	неудовлетворительно
----	---	---	---------------------

Б) КРИТЕРИИ И ШКАЛА ОЦЕНИВАНИЯ РЕЗУЛЬТАТОВ ТЕСТИРОВАНИЯ

№ п/п	тестовые нормы: % правильных ответов	количество баллов
1	90-100 %	9-10
2	80-89%	7-8
3	70-79%	5-6
4	60-69%	3-4
5	50-59%	1-2
6	менее 50%	0

В) КРИТЕРИИ И ШКАЛА ОЦЕНИВАНИЯ РЕЗУЛЬТАТОВ РЕШЕНИЯ ЗАДАЧ

№ п/п	критерии оценивания	количество баллов
1	Полное верное решение. В логическом рассуждении и решении нет ошибок, задача решена рациональным способом. Получен правильный ответ. Ясно описан способ решения.	9-10
2	Верное решение, но имеются небольшие недочеты, в целом не влияющие на решение, такие как небольшие логические пропуски, не связанные с основной идеей решения. Решение оформлено не вполне аккуратно, но это не мешает пониманию решения.	7-8
3	Решение в целом верное. В логическом рассуждении и решении нет существенных ошибок, но задача решена неоптимальным способом или допущено не более двух незначительных ошибок. В работе присутствуют арифметическая ошибка, механическая ошибка или описка при переписывании выкладок или ответа, не искажившие экономическое содержание ответа.	5-6
4	В логическом рассуждении и решении нет ошибок, но допущена существенная ошибка в математических расчетах. При объяснении сложного экономического явления указаны не все существенные факторы.	3-4
5	Имеются существенные ошибки в логическом рассуждении и в решении. Рассчитанное значение искомой величины искажает экономическое содержание ответа. Доказаны вспомогательные	2-3

	утверждения, помогающие в решении задачи.	
6	Рассмотрены отдельные случаи при отсутствии решения. Отсутствует окончательный численный ответ (если он предусмотрен в задаче). Правильный ответ угадан, а выстроенное под него решение - безосновательно.	1
7	Решение неверное или отсутствует.	0

Г) КРИТЕРИИ И ШКАЛА ОЦЕНИВАНИЯ РЕФЕРАТОВ

№ п/п	критерии оценивания	количество баллов
1	выполнены все требования к написанию и защите реферата: обозначена проблема и обоснована её актуальность, сделан краткий анализ различных точек зрения на рассматриваемую проблему и логично изложена собственная позиция, сформулированы выводы, тема раскрыта полностью, выдержан объём, соблюдены требования к внешнему оформлению, даны правильные ответы на дополнительные вопросы.	<i>9-10 баллов</i>
2	основные требования к реферату и его защите выполнены, но при этом допущены недочеты. В частности, имеются неточности в изложении материала; отсутствует логическая последовательность в суждениях; не выдержан объём реферата; имеются упущения в оформлении; на дополнительные вопросы при защите даны неполные ответы.	<i>7-8 баллов</i>
3	имеются существенные отступления от требований к реферированию. В частности: тема освещена лишь частично; допущены фактические ошибки в содержании реферата или при ответе на дополнительные вопросы.	<i>4-6 баллов</i>
4	тема освоена лишь частично; допущены грубые ошибки в содержании реферата или при ответе на дополнительные вопросы; во время защиты отсутствует вывод.	<i>1-3 баллов</i>
5	тема реферата не раскрыта, обнаруживается существенное непонимание проблемы.	<i>0 баллов</i>

Д) КРИТЕРИИ И ШКАЛА ОЦЕНИВАНИЯ ДОМАШНЕГО ЗАДАНИЯ

№ п/п	критерии оценивания	количество баллов
1	Задание выполнено полностью: цель домашнего задания успешно достигнута; основные понятия выделены; наличие схем, графическое выделение особо значимой информации; работа выполнена в полном объеме.	9-10
2	Задание выполнено: цель выполнения домашнего задания достигнута; наличие правильных эталонных ответов; однако работа выполнена не в полном объеме.	8-7
3	Задание выполнено частично: цель выполнения домашнего задания достигнута не полностью; многочисленные ошибки снижают качество выполненной работы.	6-5
4	Задание не выполнено, цель выполнения домашнего задания не достигнута.	менее 5

Е) КРИТЕРИИ И ШКАЛА ОЦЕНИВАНИЯ КОНТРОЛЬНЫХ РАБОТ

№ п/п	критерии оценивания	количество баллов	оценка
1	исключительные знания, абсолютное понимание сути вопросов, безукоризненное знание основных понятий и положений, логически и лексически грамотно изложенные, содержательные, аргументированные и исчерпывающие ответы	19-20	
2	глубокие знания материала, отличное понимание сути вопросов, твердое знание основных понятий и положений по вопросам, структурированные, последовательные, полные, правильные ответы	17-18	
3	глубокие знания материала, правильное понимание сути вопросов, знание основных понятий и положений по вопросам, содержательные, полные и конкретные ответ на вопросы. Наличие несущественных или технических ошибок	15-16	
4	твердые, достаточно полные знания, хорошее понимание сути вопросов, правильные ответы на вопросы, минимальное количество неточностей, небрежное оформление	13-14	
5	твердые, но недостаточно полные знания, по сути верное понимание вопросов, в целом правильные ответы на вопросы, наличие неточностей, небрежное оформление	11-12	

6	общие знания, недостаточное понимание сути вопросов, наличие большого числа неточностей, небрежное оформление	9-10	
7	относительные знания, наличие ошибок, небрежное оформление	7-8	
8	поверхностные знания, наличие грубых ошибок, отсутствие логики изложения материала	5-6	
9	непонимание сути, большое количество грубых ошибок, отсутствие логики изложения материала	3-4	
10	не дан ответ на поставленные вопросы	1-2	
11	отсутствие ответа, дан ответ на другие вопросы, списывание в ходе выполнения работы, наличие на рабочем месте технических средств, в том числе телефона	0	

Ж) КРИТЕРИИ И ШКАЛА ОЦЕНКИ ПРЕЗЕНТАЦИЙ

№ п/п	критерии оценки	максимальное количество баллов
1	титульный слайд с заголовком	5
2	дизайн слайдов	10
3	использование дополнительных эффектов (смена слайдов, звук, графика, анимация)	5
4	список источников информации	5
5	широта кругозора	5
6	логика изложения материала	10
7	текст хорошо написан и сформированные идеи ясно изложены и структурированы	10
8	слайды представлены в логической последовательности	5
9	грамотное создание и сохранение документов в папке рабочих материалов	5
10	слайды распечатаны в форме заметок	5
	средняя оценка:	

2.3. ТИПОВЫЕ КОНТРОЛЬНЫЕ ЗАДАНИЯ

Контрольная работа №1 «Входное тестирование»

1. Что такое интерпретатор в Python?

- a) Программа, которая переводит код на другой язык программирования
- b) Программа, которая выполняет код построчно
- c) Программа, которая преобразует исходный код в машинный код
- d) Текстовый редактор для написания кода

Ответ: b

2. Какая функция является точкой входа в программу на Python?

- a) start()
- b) main()
- c) begin()
- d) Программа выполняется с первой строки файла

Ответ: d

3. Какая функция используется для вывода данных в консоль?

- a) input()
- b) cout
- c) print()
- d) echo

Ответ: c

4. Что означает инструкция import math?

- a) Подключение математической библиотеки
- b) Подключение библиотеки для работы с файлами
- c) Подключение библиотеки ввода-вывода
- d) Объявление переменных

Ответ: a

5. Какой тип данных используется для целых чисел?

- a) float
- b) double
- c) str
- d) int

Ответ: d

6. Что выведет эта программа?

```
python
x = 5
y = 2
result = x * y + 3
print(result)
```

- a) 10
- b) 13
- c) 16
- d) 25

Ответ: b

7. Что произойдет при выполнении этого кода?

```
python
a = 10
b = 3
if a > b:
    print("Больше")
else:
    print("Меньше")
```

- a) Выведет "Больше"
- b) Выведет "Меньше"
- c) Выведет "10"
- d) Будет ошибка

Ответ: a

8. Каков будет результат работы программы?

```
python
numbers = [1, 2, 3, 4, 5]
total = 0
for num in numbers:
    total += num
print(total)
```

- a) 5
- b) 10
- c) 15

d) 20

Ответ: c

9. Что выведет этот код?

```
python
def multiply(a, b):
    return a * b
```

```
result = multiply(4, 5)
print(result)
```

a) 9

b) 20

c) 45

d) Будет ошибка

Ответ: b

10. Сколько раз выполнится цикл?

```
python
count = 0
i = 1
while i <= 5:
    count += 1
    i += 1
print(count)
```

a) 4 раза

b) 5 раз

c) 6 раз

d) Бесконечно

Ответ: b

Установите соответствие между понятием и его описанием в Python:

Понятие

Описание

1. Переменная

A. Именованная область памяти для хранения данных

2. Константа

B. Функция для ввода данных с клавиатуры

3. Интерпретация

C. Значение, которое нельзя изменить в процессе выполнения

4. input()

D. Построчное выполнение исходного кода

5. Список

E. Изменяемая коллекция элементов

Ответ: 1-A, 2-C, 3-D, 4-B, 5-E

Балльная система:

- Вопросы 1-5: 1 балл каждый
- Вопросы 6-10: 2 балла каждый
- Задание на соответствие: 5 баллов
- **Максимум: 20 баллов**

Практическая часть

Задача 1. Линейная программа (5 баллов)

Условие:

Напишите программу, которая вычисляет стоимость покупки. Пользователь вводит цену одного товара и количество товаров. Программа должна вычислить общую стоимость и вывести ее с учетом 20% скидки.

Пример работы программы:

Введите цену товара: 100

Введите количество: 3

Общая стоимость: 300 руб.

Стоимость со скидкой: 240 руб.

Критерии оценки:

- ✓ Ввод данных (1 балл)
 - ✓ Правильные вычисления (2 балла)
 - ✓ Корректный вывод (1 балл)
 - ✓ Форматирование кода (1 балл)
-

Задача 2. Условный оператор (5 баллов)

Условие:

Напишите программу, которая определяет категорию студента по среднему баллу:

- 90-100: «Отличник»
- 75-89: «Хорошист»
- 60-74: «Удовлетворительно»
- Менее 60: «Неудовлетворительно»

Пример работы программы:

text

Введите средний балл: 85

Категория: Хорошист

Критерии оценки:

- ✓ Правильное условие (2 балла)
 - ✓ Обработка граничных значений (1 балл)
 - ✓ Корректный вывод (1 балл)
 - ✓ Читаемость кода (1 балл)
-

Задача 3. Цикл for (5 баллов)

Условие:

Напишите программу, которая выводит все четные числа от 2 до N (N вводится пользователем) и подсчитывает их количество.

Пример работы программы:

text

Введите число N: 10

Четные числа: 2 4 6 8 10

Количество четных чисел: 5

Критерии оценки:

- ✓ Правильная организация цикла (2 балла)
 - ✓ Проверка четности (1 балл)
 - ✓ Подсчет количества (1 балл)
 - ✓ Форматирование вывода (1 балл)
-

Задача 4. Цикл while (5 баллов)

Условие:

Напишите программу, которая запрашивает у пользователя числа до тех пор, пока не будет введен 0. Затем программа выводит сумму и среднее арифметическое введенных чисел.

Пример работы программы:

text

Введите число (0 для завершения): 5

Введите число (0 для завершения): 10

Введите число (0 для завершения): 15

Введите число (0 для завершения): 0

Сумма: 30

Среднее арифметическое: 10.0

Критерии оценки:

- ✓ Правильная организация цикла while (2 балла)
- ✓ Накопление суммы и подсчет количества (1 балл)
- ✓ Вычисление среднего (1 балл)
- ✓ Обработка деления на ноль (1 балл)

Пример решения задачи 1 (линейная программа):

python

Ввод данных

price = float(input("Введите цену товара: "))

quantity = int(input("Введите количество: "))

Вычисления

total = price * quantity

discount = total * 0.2

final_price = total - discount

Вывод результатов

print(f"Общая стоимость: {total} руб.")

print(f"Стоимость со скидкой: {final_price} руб.")

Пример решения задачи 2 (условный оператор):

python

Ввод данных

average_score = float(input("Введите средний балл: "))

Определение категории

if average_score >= 90:

 category = "Отличник"

elif average_score >= 75:

 category = "Хорошист"

elif average_score >= 60:

 category = "Удовлетворительно"

else:

 category = "Неудовлетворительно"

Вывод результата

print(f"Категория: {category}")

Пример решения задачи 3 (цикл for):

```
python
# Ввод данных
n = int(input("Введите число N: "))

# Инициализация счетчика
count = 0
even_numbers = []

# Цикл for
for i in range(2, n + 1, 2):
    even_numbers.append(i)
    count += 1

# Вывод результатов
print("Четные числа:", *even_numbers)
print(f"Количество четных чисел: {count}")
```

Пример решения задачи 4 (цикл while):

```
python
# Инициализация переменных
total = 0
count = 0

# Цикл while
while True:
    number = float(input("Введите число (0 для завершения): "))

    if number == 0:
        break

    total += number
    count += 1

# Вычисление среднего
if count > 0:
    average = total / count
else:
    average = 0

# Вывод результатов
print(f"Сумма: {total}")
print(f"Среднее арифметическое: {average}")
```

Итоговая система оценивания:

Часть теста	Максимум баллов
Теоретическая часть (10 вопросов)	15 баллов
Соответствие	5 баллов

Часть теста	Максимум баллов
Задача 1 (линейная)	5 баллов
Задача 2 (условный оператор)	5 баллов
Задача 3 (цикл for)	5 баллов
Задача 4 (цикл while)	5 баллов
Итого	40 баллов

Шкала оценок:

- 36-40 баллов: Отлично
- 28-35 баллов: Хорошо
- 20-27 баллов: Удовлетворительно
- Менее 20 баллов: Неудовлетворительно

Контрольная работа №2 «Введение в программирование»

Цель: Проверить понимание базовых концепций, синтаксиса и этапов решения задач.

Время выполнения: 60 минут

Вариант: _____

ФИО студента: _____

Вариант 1

1. Какой оператор используется для присваивания значения?

- a) ==
- b) =
- c) :=
- d) ->

Ответ: b

2. Что выведет код: `cout << (5 > 3 ? "yes" : "no");`?

- a) 5
- b) 3
- c) yes
- d) no

Ответ: c

3. Какой цикл гарантированно выполнится хотя бы один раз?

- a) for
- b) while
- c) do-while
- d) foreach

Ответ: c

4. Что делает оператор `break` в `switch`?

- a) Завершает программу
- b) Прерывает выполнение текущей итерации цикла
- c) Выходит из текущего `case` блока
- d) Продолжает выполнение следующего `case`

Ответ: c

5. Какое условие проверяет, что число x находится в диапазоне [1, 10]?

a) $x \geq 1 \ \&\& \ x \leq 10$

b) $x > 1 \ \&\& \ x < 10$

c) $x = 1 \ || \ x = 10$

d) $x \in [1, 10]$

Ответ: a

6. Установите соответствие между оператором и его описанием:

Оператор	Описание
1. ++	А. Логическое И
2. --	В. Увеличение на 1
3. &&	С. Остаток от деления
4.	Д. Уменьшение на 1
5. %	Е. Логическое ИЛИ

Ответ: 1-В, 2-Д, 3-А, 4-Е, 5-С

7. **Анализ:** Найдите и исправьте ошибки в коде:

```
#include <iostream>
using namespace std;
int main() {
    int x, y;
    cout << "Введите два числа: ";
    cin >> x >> y;
    result = x + y;
    cout << "Сумма: " << result;
    return 0;
}
```

8. **Анализ:** Найдите и исправьте ошибки в коде:

```
#include <iostream>
using namespace std;
int main() {
    double a, b;
    cout << "Введите число: ";
    cin >> a;
    b = a / 2.0;
    cout << "Половина числа: " << b;
    return 0;
}
```

Практическая часть:

1. Напишите программу, которая вычисляет площадь прямоугольника по введенным сторонам

2. Напишите программу, которая переводит температуру из градусов Цельсия в Фаренгейты

ТАБЛИЦА ПРАВИЛЬНЫХ ОТВЕТОВ

<i>№ вопроса</i>	<i>Правильный ответ</i>	<i>Баллы</i>
<i>1</i>	<i>b</i>	<i>1</i>
<i>2</i>	<i>c</i>	<i>1</i>
<i>3</i>	<i>c</i>	<i>1</i>
<i>4</i>	<i>c</i>	<i>1</i>
<i>5</i>	<i>a</i>	<i>1</i>
<i>6</i>	<i>1-B, 2-D, 3-A, 4-E, 5-C</i>	<i>5 (по 1 за каждое)</i>
<i>7</i>	<i>См. эталонное решение</i>	<i>5</i>
<i>8</i>	<i>См. эталонное решение</i>	<i>5</i>
<i>Теоретическая часть итого</i>		<i>20</i>
<i>Практическая задача 1</i>	<i>См. эталонное решение</i>	<i>10</i>
<i>Практическая задача 2</i>	<i>См. эталонное решение</i>	<i>10</i>
<i>Практическая часть итого</i>		<i>20</i>
<i>ВСЕГО</i>		<i>40</i>

ЭТАЛОННЫЕ РЕШЕНИЯ**Вопрос 7 - Анализ и исправление кода:****Исходный код с ошибками:**

```

#include <iostream>
using namespace std
int main() {
    int x, y
    cout << "Введите два числа: ";
    cin >> x >> y;
    result = x + y;
    cout << "Сумма: " << result
    return 0;
}

```

Исправленный код:

```
#include <iostream>
using namespace std; // Добавлена точка с запятой

int main() {
    int x, y; // Добавлена точка с запятой
    int result; // Добавлено объявление переменной result
    cout << "Введите два числа: ";
    cin >> x >> y;
    result = x + y;
    cout << "Сумма: " << result; // Добавлена точка с запятой
    return 0;
}
```

Найденные ошибки:

1.  Отсутствует ; после using namespace std
 2.  Отсутствует ; после int x, y
 3.  Переменная result не объявлена
 4.  Отсутствует ; после cout << "Сумма: " << result
-

Вопрос 8 - Анализ и исправление кода:

Исходный код с ошибками:

```
#include <iostream>
using namespace std;
int main() {
    double a, b;
    cout << "Введите число: " // Ошибка - нет ;
    cin >> a;
    b = a / 2.0 // Ошибка - нет ;
    cout << "Половина числа: " << b;
    return 0 // Ошибка - нет ;
}
```

Исправленный код:

```
#include <iostream>
using namespace std;
int main() {
    double a, b;
    cout << "Введите число: "; // Добавлена ;
    cin >> a;
    b = a / 2.0; // Добавлена ;
    cout << "Половина числа: " << b;
    return 0; // Добавлена ;
}
```

Найденные ошибки:

1.  Отсутствует ; после cout << "Введите число: "
 2.  Отсутствует ; после b = a / 2.0
 3.  Отсутствует ; после return 0
-

ЭТАЛОННЫЕ РЕШЕНИЯ ПРАКТИЧЕСКИХ ЗАДАЧ

Задача 1: Площадь прямоугольника

```
#include <iostream>
using namespace std;

int main() {
    double length, width, area;

    // Ввод данных
    cout << "Введите длину прямоугольника: ";
    cin >> length;
    cout << "Введите ширину прямоугольника: ";
    cin >> width;

    // Вычисление площади
    area = length * width;

    // Вывод результата
    cout << "Площадь прямоугольника: " << area << endl;

    return 0;
}
```

Задача 2: Перевод температуры Цельсия в Фаренгейты

```
#include <iostream>
using namespace std;

int main() {
    double celsius, fahrenheit;

    // Ввод данных
    cout << "Введите температуру в градусах Цельсия: ";
    cin >> celsius;

    // Вычисление по формуле
    fahrenheit = (celsius * 9.0 / 5.0) + 32;

    // Вывод результата
    cout << celsius << " градусов Цельсия = "
         << fahrenheit << " градусов Фаренгейта" << endl;

    return 0;
}
```

ШКАЛА ОЦЕНОК

Баллы	Оценка	Уровень
36-40	5 (Отлично)	Высокий уровень знаний
28-35	4 (Хорошо)	Достаточный уровень знаний

Баллы	Оценка	Уровень
20-27	3 (Удовлетворительно)	Минимальный уровень знаний
0-19	2 (Неудовлетворительно)	Недостаточный уровень знаний

Контрольная работа №3 «Операторы языка программирования»

Цель: Проверить знания основных операторов C++ и умение их применять на практике

Время выполнения: 60 минут

Вариант: _____

ФИО студента: _____

Тестовая часть (одинаковая для всех вариантов)

1. Выберите правильный ответ (1 балл каждый)

1. Какой оператор используется для присваивания значения?

- a) ==
- b) =
- c) :=
- d) ->

2. Что выведет код: `cout << (5 > 3 ? "yes" : "no");`?

- a) 5
- b) 3
- c) yes
- d) no

3. Какой цикл гарантированно выполнится хотя бы один раз?

- a) for
- b) while
- c) do-while
- d) foreach

4. Что делает оператор `break` в `switch`?

- a) Завершает программу
- b) Прерывает выполнение текущей итерации цикла
- c) Выходит из текущего `case` блока
- d) Продолжает выполнение следующего `case`

5. Какое условие проверяет, что число `x` находится в диапазоне `[1, 10]`?

- a) `x >= 1 && x <= 10`
- b) `x > 1 && x < 10`
- c) `x = 1 || x = 10`
- d) `x ∈ [1,10]`

2. Установите соответствие (1 балл за каждое правильное соответствие)

Оператор	Описание
1. ++	А. Логическое И
2. --	В. Увеличение на 1
3. &&	С. Остаток от деления

Оператор	Описание
4.	D. Уменьшение на 1
5. %	E. Логическое ИЛИ

Практическая часть (выполняется по вариантам)

ВАРИАНТ 1

1. Напишите программу, которая определяет, является ли введенное число положительным, отрицательным или нулем
2. Напишите программу, которая выводит все числа от 1 до N, кратные 3
3. Напишите программу, которая вычисляет сумму цифр двухзначного числа

ВАРИАНТ 2

1. Напишите программу, которая определяет, является ли год високосным
2. Напишите программу, которая выводит таблицу умножения для числа K
3. Напишите программу, которая переворачивает двухзначное число (например, 57 → 75)

ТАБЛИЦА ПРАВИЛЬНЫХ ОТВЕТОВ

Тестовая часть

№ вопроса	Правильный ответ	Баллы
1	b	1
2	c	1
3	c	1
4	c	1
5	a	1
Соответствие 1	B	1
Соответствие 2	D	1
Соответствие 3	A	1
Соответствие 4	E	1
Соответствие 5	C	1
Итого тест		10

Практическая часть (10 баллов за каждую задачу)

Задача	Максимум баллов	Критерии оценивания
1	10	Синтаксис (2), логика (4), работа программы (4)
2	10	Синтаксис (2), логика (4), работа программы (4)
3	10	Синтаксис (2), логика (4), работа программы (4)
Итого практика	30	

ВСЕГО ЗА РАБОТУ: 40 баллов**ЭТАЛОННЫЕ РЕШЕНИЯ ПРАКТИЧЕСКИХ ЗАДАЧ****ВАРИАНТ 1****Задача 1: Определение знака числа**

```
#include <iostream>
using namespace std;

int main() {
    int number;
    cout << "Введите число: ";
    cin >> number;

    if (number > 0) {
        cout << "Положительное число" << endl;
    } else if (number < 0) {
        cout << "Отрицательное число" << endl;
    } else {
        cout << "Ноль" << endl;
    }

    return 0;
}
```

Задача 2: Числа, кратные 3

```
#include <iostream>
using namespace std;

int main() {
    int N;
    cout << "Введите число N: ";
    cin >> N;

    cout << "Числа, кратные 3: ";
    for (int i = 1; i <= N; i++) {
        if (i % 3 == 0) {
            cout << i << " ";
        }
    }
}
```

```

    cout << endl;

    return 0;
}
Задача 3: Сумма цифр двухзначного числа

```

```

#include <iostream>
using namespace std;

int main() {
    int number;
    cout << "Введите двухзначное число: ";
    cin >> number;

    if (number >= 10 && number <= 99) {
        int digit1 = number / 10;  // Первая цифра
        int digit2 = number % 10;  // Вторая цифра
        int sum = digit1 + digit2;

        cout << "Сумма цифр: " << sum << endl;
    } else {
        cout << "Ошибка: введите двухзначное число!" << endl;
    }

    return 0;
}

```

ВАРИАНТ 2

Задача 1: Определение високосного года

```

#include <iostream>
using namespace std;

int main() {
    int year;
    cout << "Введите год: ";
    cin >> year;

    if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)) {
        cout << year << " - високосный год" << endl;
    } else {
        cout << year << " - не високосный год" << endl;
    }

    return 0;
}

```

Задача 2: Таблица умножения

```

#include <iostream>
using namespace std;

int main() {
    int K;

```

```

cout << "Введите число К: ";
cin >> K;

cout << "Таблица умножения для " << K << ":" << endl;
for (int i = 1; i <= 10; i++) {
    cout << K << " * " << i << " = " << K * i << endl;
}

return 0;
}

```

Задача 3: Переворот двухзначного числа

```

#include <iostream>
using namespace std;

int main() {
    int number;
    cout << "Введите двухзначное число: ";
    cin >> number;

    if (number >= 10 && number <= 99) {
        int digit1 = number / 10; // Первая цифра
        int digit2 = number % 10; // Вторая цифра
        int reversed = digit2 * 10 + digit1;

        cout << "Перевернутое число: " << reversed << endl;
    } else {
        cout << "Ошибка: введите двухзначное число!" << endl;
    }

    return 0;
}

```

ШКАЛА ОЦЕНОК

Баллы	Оценка	Уровень выполнения
36-40	5 (Отлично)	Программы работают корректно, код оптимален
28-35	4 (Хорошо)	Программы работают, возможны мелкие недочеты
20-27	3 (Удовлетворительно)	Основной функционал реализован
0-19	2 (Неудовлетворительно)	Значительные ошибки в коде

Контрольная работа №4 «Структурное и модульное программирование»

Цель: Проверить умение разбивать задачу на подзадачи, использовать подпрограммы

Время выполнения: 70 минут

Вариант: _____

ФИО студента: _____

Тестовая часть (одинаковая для всех вариантов)

1. Выберите правильный ответ (1 балл каждый)

1. Что такое функция в C++?

- a) Именованный блок операторов
- b) Тип данных
- c) Оператор цикла
- d) Математическая формула

2. Как передается параметр по умолчанию в C++?

- a) По ссылке
- b) По указателю
- c) По значению
- d) По константе

3. Что означает ключевое слово void в объявлении функции?

- a) Функция возвращает целое число
- b) Функция не возвращает значение
- c) Функция принимает любые параметры
- d) Функция пустая

4. Какое ключевое слово используется для передачи параметра по ссылке?

- a) ref
- b) &
- c) *
- d) link

5. Что такое перегрузка функций?

- a) Изменение имени функции
- b) Создание нескольких функций с одним именем, но разными параметрами
- c) Увеличение скорости работы функции
- d) Удаление функции из программы

6. Что такое формальные параметры функции?

- a) Параметры, которые передаются при вызове функции
- b) Параметры, объявленные в заголовке функции
- c) Параметры по умолчанию
- d) Глобальные переменные

7. В чем разница между передачей параметров по значению и по ссылке?

- a) По значению передается копия, по ссылке - оригинал
- b) По значению передается оригинал, по ссылке - копия
- c) Разница только в синтаксисе
- d) По значению для чисел, по ссылке для строк

8. Что такое область видимости переменных?

- a) Размер переменной в памяти
- b) Часть программы, где переменная доступна
- c) Тип данных переменной
- d) Значение переменной

9. Для чего используется оператор return в функциях?

- a) Для завершения программы
- b) Для возврата значения из функции
- c) Для вывода на экран
- d) Для ввода данных

10. Какие виды областей видимости вы знаете?

- a) Локальная и глобальная
- b) Внутренняя и внешняя
- c) Верхняя и нижняя
- d) Левая и правая

2. Установите соответствие (1 балл за каждое правильное соответствие)

Понятие	Пример
1. Объявление функции	A. <code>int sum(int a, int b) { return a+b; }</code>
2. Вызов функции	B. <code>int sum(int a, int b);</code>
3. Параметр по значению	C. <code>void func(int x)</code>
4. Параметр по ссылке	D. <code>result = sum(5, 3);</code>
5. Возврат значения	E. <code>void func(int &x)</code>

Практическая часть (выполняется по вариантам)

ВАРИАНТ 1

- 1. Создайте функцию `isPrime(n)`, которая проверяет, является ли число простым
- 2. Создайте функцию `arraySum(arr, size)`, которая вычисляет сумму элементов массива
- 3. Напишите программу, используя созданные функции, чтобы найти сумму всех простых чисел в массиве

ВАРИАНТ 2

- 1. Создайте функцию `power(a, b)`, которая вычисляет a в степени b
- 2. Создайте функцию `findMax(arr, size)`, которая находит максимальный элемент в массиве
- 3. Напишите программу, которая использует созданные функции для нахождения максимального из результатов возведения чисел в степень

ТАБЛИЦА ПРАВИЛЬНЫХ ОТВЕТОВ

Тестовая часть

№ вопроса	Правильный ответ	Баллы
1	a	1
2	c	1
3	b	1
4	b	1

№ вопроса	Правильный ответ	Баллы
5	b	1
6	b	1
7	a	1
8	b	1
9	b	1
10	a	1
Соответствие 1	B	1
Соответствие 2	D	1
Соответствие 3	C	1
Соответствие 4	E	1
Соответствие 5	A	1
Итого тест		15

Практическая часть (15 баллов за каждую задачу)

Задача	Максимум баллов	Критерии оценивания
1	15	Объявление функции (3), логика (6), работа (6)
2	15	Объявление функции (3), логика (6), работа (6)
3	15	Использование функций (5), логика (5), работа (5)
Итого практика	45	

ВСЕГО ЗА РАБОТУ: 60 баллов

ЭТАЛОННЫЕ РЕШЕНИЯ ПРАКТИЧЕСКИХ ЗАДАЧ

ВАРИАНТ 1

Задача 1: Функция проверки простого числа

```
#include <iostream>
#include <cmath>
using namespace std;

bool isPrime(int n) {
    if (n <= 1) return false;
    if (n == 2) return true;
    if (n % 2 == 0) return false;

    for (int i = 3; i <= sqrt(n); i += 2) {
        if (n % i == 0) return false;
    }
    return true;
}
```

Задача 2: Функция суммы элементов массива

```
int arraySum(int arr[], int size) {
    int sum = 0;
    for (int i = 0; i < size; i++) {
        sum += arr[i];
    }
    return sum;
}
```

Задача 3: Основная программа

```
int main() {
    const int SIZE = 10;
    int arr[SIZE] = {2, 3, 4, 5, 6, 7, 8, 9, 10, 11};
    int primeSum = 0;

    for (int i = 0; i < SIZE; i++) {
        if (isPrime(arr[i])) {
            primeSum += arr[i];
        }
    }

    cout << "Сумма всех простых чисел в массиве: " << primeSum << endl;
    cout << "Общая сумма элементов массива: " << arraySum(arr, SIZE) << endl;

    return 0;
}
```

ВАРИАНТ 2

Задача 1: Функция возведения в степень

```
#include <iostream>
using namespace std;

double power(double a, int b) {
    double result = 1.0;
    for (int i = 0; i < b; i++) {
        result *= a;
    }
    return result;
}
```

Задача 2: Функция поиска максимума в массиве

```
int findMax(int arr[], int size) {
    int max = arr[0];
    for (int i = 1; i < size; i++) {
        if (arr[i] > max) {
            max = arr[i];
        }
    }
    return max;
}
```

Задача 3: Основная программа

```
int main() {
    const int SIZE = 5;
    int numbers[SIZE] = {2, 3, 4, 5, 6};
    int powers[SIZE];

    // Вычисляем степени для каждого числа
    for (int i = 0; i < SIZE; i++) {
        powers[i] = power(numbers[i], 2); // Возводим в квадрат
    }

    // Находим максимальную степень
    int maxPower = findMax(powers, SIZE);

    cout << "Исходные числа: ";
    for (int i = 0; i < SIZE; i++) {
        cout << numbers[i] << " ";
    }
    cout << endl;

    cout << "Их квадраты: ";
    for (int i = 0; i < SIZE; i++) {
        cout << powers[i] << " ";
    }
    cout << endl;

    cout << "Максимальный квадрат: " << maxPower << endl;
}
```

```

    return 0;
}

```

ШКАЛА ОЦЕНОК

Баллы	Оценка	Уровень выполнения
54-60	5 (Отлично)	Все функции работают корректно, код оптимален
42-53	4 (Хорошо)	Функции работают, возможны мелкие недочеты
30-41	3 (Удовлетворительно)	Основной функционал реализован
0-29	2 (Неудовлетворительно)	Значительные ошибки в реализации функций

КРИТЕРИИ ОЦЕНИВАНИЯ ПРАКТИЧЕСКОЙ ЧАСТИ

- **Объявление функции** (3 балла): правильный синтаксис, тип возвращаемого значения, параметры
- **Логика функции** (6 баллов): корректный алгоритм, обработка граничных случаев
- **Работа функции** (6 баллов): функция возвращает правильный результат для всех тестовых случаев
- **Использование функций** (5 баллов): правильный вызов созданных функций в основной программе
- **Интеграция** (5 баллов): функции корректно работают вместе в основной программе
- **Читаемость кода** (5 баллов): понятные имена переменных, комментарии, форматирование

Контрольная работа №5 «Основные конструкции языков программирования»

Цель: Проверить умение работать со структурами данных (массивы, строки)

Время выполнения: 70 минут

Вариант: _____

ФИО студента: _____

Тестовая часть (одинаковая для всех вариантов)

1. Выберите правильный ответ (1 балл каждый)

1. Как объявить массив из 10 целых чисел?

- a) array int[10];
- b) int array(10);
- c) int array[10];
- d) int array = new[10];

2. Что такое нуль-терминатор в строке C++?

- a) Символ '\n'
- b) Символ '\0'
- c) Символ ' '
- d) Символ '\t'

3. Как получить длину строки str в C++?

- a) str.length()
- b) len(str)
- c) sizeof(str)
- d) str.size

4. Что делает оператор new в C++?

- a) Создает новую переменную
- b) Выделяет динамическую память
- c) Объявляет функцию
- d) Инициализирует массив

5. Какой заголовочный файл нужен для работы со строками?

- a) <iostream>
- b) <string>
- c) <cstring>
- d) <vector>

6. Чем отличается инициализация массива от присваивания значений?

- a) Инициализация - при объявлении, присваивание - после объявления
- b) Это одно и то же
- c) Инициализация только для чисел, присваивание для строк
- d) Разница только в синтаксисе

7. Что такое нуль-терминатор в строках?

- a) Символ, обозначающий конец строки
- b) Символ пробела
- c) Символ перевода строки
- d) Символ табуляции

8. Для чего нужен нуль-терминатор в строках C++?

- a) Для определения конца строки в символьных массивах
- b) Для красоты кода
- c) Для ускорения работы программы
- d) Для хранения длины строки

9. Какая функция используется для копирования строк в C?

- a) strcpy()
- b) strcopy()
- c) copy_str()
- d) string_copy()

10. Как объединить две строки в C++?

- a) str1 + str2
- b) str1.concat(str2)
- c) str1 & str2
- d) concat(str1, str2)

2. Установите соответствие (1 балл за каждое правильное соответствие)

Операция

Синтаксис

1. Объявление массива

A. `int *arr = new int[n];`

2. Динамическое выделение памяти

B. `string str = "Hello";`

3. Инициализация строки

C. `int arr[5] = {1,2,3,4,5};`

4. Освобождение памяти

D. `arr[i] = value;`

5. Обращение к элементу массива

E. `delete[] arr;`

Практическая часть (выполняется по вариантам)

ВАРИАНТ 1

1. Напишите программу, которая подсчитывает количество пробелов в строке
2. Дан массив целых чисел. Напишите программу, которая находит количество элементов, больших среднего арифметического всех элементов
3. Напишите программу, которая преобразует строку, заменяя все буквы 'a' на 'A'

ВАРИАНТ 2

1. Напишите программу, которая проверяет, является ли строка палиндромом
2. Дан массив чисел. Напишите программу, которая сортирует его по возрастанию методом пузырька
3. Напишите программу, которая объединяет две строки без использования стандартной функции конкатенации

ТАБЛИЦА ПРАВИЛЬНЫХ ОТВЕТОВ

Тестовая часть

№ вопроса	Правильный ответ	Баллы
1	c	1
2	b	1
3	a	1
4	b	1
5	b	1
6	a	1
7	a	1
8	a	1
9	b	1

№ вопроса	Правильный ответ	Баллы
10	a	1
Соответствие 1	C	1
Соответствие 2	A	1
Соответствие 3	B	1
Соответствие 4	E	1
Соответствие 5	D	1
Итого тест		15

Практическая часть (15 баллов за каждую задачу)

Задача	Максимум баллов	Критерии оценивания
1	15	Синтаксис (3), логика (6), работа программы (6)
2	15	Синтаксис (3), логика (6), работа программы (6)
3	15	Синтаксис (3), логика (6), работа программы (6)
Итого практика	45	

ВСЕГО ЗА РАБОТУ: 60 баллов

ЭТАЛОННЫЕ РЕШЕНИЯ ПРАКТИЧЕСКИХ ЗАДАЧ

ВАРИАНТ 1

Задача 1: Подсчет пробелов в строке

```
#include <iostream>
#include <string>
using namespace std;

int main() {
    string str;
    int spaceCount = 0;
```

```

cout << "Введите строку: ";
getline(cin, str);

for (int i = 0; i < str.length(); i++) {
    if (str[i] == ' ') {
        spaceCount++;
    }
}

cout << "Количество пробелов в строке: " << spaceCount << endl;

return 0;
}

```

Задача 2: Элементы больше среднего арифметического

```

#include <iostream>
using namespace std;

int main() {
    const int SIZE = 10;
    int arr[SIZE] = {5, 12, 8, 20, 3, 15, 7, 18, 9, 11};
    int sum = 0;
    int count = 0;

    // Вычисляем сумму элементов
    for (int i = 0; i < SIZE; i++) {
        sum += arr[i];
    }

    // Вычисляем среднее арифметическое
    double average = static_cast<double>(sum) / SIZE;

    // Считаем элементы больше среднего
    for (int i = 0; i < SIZE; i++) {
        if (arr[i] > average) {
            count++;
        }
    }

    cout << "Среднее арифметическое: " << average << endl;
    cout << "Количество элементов больше среднего: " << count << endl;

    return 0;
}

```

Задача 3: Замена букв в строке

```

#include <iostream>
#include <string>
using namespace std;

```

```

int main() {
    string str;

    cout << "Введите строку: ";
    getline(cin, str);

    for (int i = 0; i < str.length(); i++) {
        if (str[i] == 'a') {
            str[i] = 'A';
        }
    }

    cout << "Преобразованная строка: " << str << endl;

    return 0;
}

```

ВАРИАНТ 2

Задача 1: Проверка палиндрома

```

#include <iostream>
#include <string>
#include <algorithm>
using namespace std;

int main() {
    string str;

    cout << "Введите строку: ";
    getline(cin, str);

    // Удаляем пробелы и преобразуем к нижнему регистру
    string processedStr = "";
    for (char c : str) {
        if (c != ' ') {
            processedStr += tolower(c);
        }
    }

    // Проверяем на палиндром
    bool isPalindrome = true;
    int len = processedStr.length();
    for (int i = 0; i < len / 2; i++) {
        if (processedStr[i] != processedStr[len - 1 - i]) {
            isPalindrome = false;
            break;
        }
    }

    if (isPalindrome) {
        cout << "Строка является палиндромом" << endl;
    } else {
        cout << "Строка не является палиндромом" << endl;
    }
}

```

```

    }

    return 0;
}
Задача 2: Сортировка пузырьком

#include <iostream>
using namespace std;

int main() {
    const int SIZE = 8;
    int arr[SIZE] = {64, 34, 25, 12, 22, 11, 90, 5};

    cout << "Исходный массив: ";
    for (int i = 0; i < SIZE; i++) {
        cout << arr[i] << " ";
    }
    cout << endl;

    // Сортировка пузырьком
    for (int i = 0; i < SIZE - 1; i++) {
        for (int j = 0; j < SIZE - i - 1; j++) {
            if (arr[j] > arr[j + 1]) {
                // Обмен элементов
                int temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
        }
    }

    cout << "Отсортированный массив: ";
    for (int i = 0; i < SIZE; i++) {
        cout << arr[i] << " ";
    }
    cout << endl;

    return 0;
}

```

Задача 3: Объединение строк без конкатенации

```

#include <iostream>
#include <string>
using namespace std;

int main() {
    string str1, str2, result;

    cout << "Введите первую строку: ";
    getline(cin, str1);
    cout << "Введите вторую строку: ";
    getline(cin, str2);

```

```

// Ручное объединение строк
result = str1; // Копируем первую строку

// Добавляем вторую строку посимвольно
for (int i = 0; i < str2.length(); i++) {
    result += str2[i];
}

cout << "Объединенная строка: " << result << endl;

return 0;
}

```

ШКАЛА ОЦЕНОК

Баллы	Оценка	Уровень выполнения
54-60	5 (Отлично)	Все программы работают корректно, код оптимален
42-53	4 (Хорошо)	Программы работают, возможны мелкие недочеты
30-41	3 (Удовлетворительно)	Основной функционал реализован
0-29	2 (Неудовлетворительно)	Значительные ошибки в коде

КРИТЕРИИ ОЦЕНИВАНИЯ ПРАКТИЧЕСКОЙ ЧАСТИ

- **Синтаксис** (3 балла): правильное объявление переменных, использование операторов
- **Логика** (6 баллов): корректный алгоритм решения задачи
- **Работа программы** (6 баллов): программа компилируется и выдает правильный результат
- **Обработка данных** (3 балла): правильная работа с массивами и строками
- **Читаемость кода** (3 балла): понятные имена переменных, комментарии, форматирование

Контрольная работа №6 «Объектно-ориентированное программирование»

Цель: Проверить понимание основных концепций ООП и умение их применять

Время выполнения: 80 минут

Вариант: _____

ФИО студента: _____

Тестовая часть (одинаковая для всех вариантов)

1. Выберите правильный ответ (1 балл каждый)

1. Что такое класс в C++?

- Функция для группировки операторов
- Пользовательский тип данных
- Массив объектов
- Блок кода

2. Какое ключевое слово обеспечивает инкапсуляцию?

- a) public
- b) private
- c) protected
- d) class

3. Что такое конструктор?

- a) Функция, уничтожающая объект
- b) Функция, инициализирующая объект
- c) Функция для копирования объекта
- d) Функция для сравнения объектов

4. Что означает наследование в ООП?

- a) Создание копии класса
- b) Соккрытие реализации
- c) Создание нового класса на основе существующего
- d) Группировка методов

5. Что такое полиморфизм?

- a) Возможность использовать один интерфейс для разных типов
- b) Соккрытие данных
- c) Наследование свойств
- d) Инкапсуляция методов

6. Что такое конструктор класса?

- a) Метод для создания объектов класса
- b) Метод для удаления объектов
- c) Метод для вывода информации
- d) Метод для сравнения объектов

7. Какие виды конструкторов вы знаете?

- a) По умолчанию, с параметрами, копирования
- b) Главный, вспомогательный, финальный
- c) Публичный, приватный, защищенный
- d) Статический, динамический, виртуальный

8. Что такое инкапсуляция?

- a) Соккрытие реализации и данных класса
- b) Наследование свойств родительского класса
- c) Переопределение методов
- d) Создание множества объектов

9. Что такое наследование в ООП?

- a) Механизм создания новых классов на основе существующих
- b) Способ скрывтия данных
- c) Метод оптимизации кода
- d) Вид полиморфизма

10. Какие преимущества дает наследование?

- a) Повторное использование кода, иерархия классов
- b) Ускорение работы программы
- c) Уменьшение объема памяти
- d) Упрощение синтаксиса

11. Что такое полиморфизм?

- a) Возможность объектов разных классов отвечать на одинаковые сообщения
- b) Соккрытие внутренней реализации
- c) Создание цепочки наследования
- d) Инкапсуляция данных

12. Как проявляется полиморфизм в C++?

- a) Через виртуальные функции и переопределение методов
- b) Через конструкторы и деструкторы
- c) Через шаблоны классов
- d) Через пространства имен

2. Установите соответствие (1 балл за каждое правильное соответствие)

Концепция	Описание
1. Инкапсуляция	A. Создание класса на основе существующего
2. Наследование	B. Разные реализации методов с одним именем
3. Полиморфизм	C. Соккрытие внутренней реализации
4. Конструктор	D. Функция, вызываемая при создании объекта
5. Деструктор	E. Функция, вызываемая при уничтожении объекта

Практическая часть (выполняется по вариантам)

ВАРИАНТ 1

- 1. Создайте класс Student с полями: имя, фамилия, средний балл. Реализуйте конструкторы, геттеры, сеттеры и метод displayInfo()
- 2. Создайте класс Group, который содержит массив студентов и методы для добавления студента, вывода всех студентов и поиска студента по фамилии

ВАРИАНТ 2

- 1. Создайте класс BankAccount с полями: номер счета, баланс, владелец. Реализуйте методы для пополнения и снятия средств
- 2. Создайте класс-наследник SavingsAccount, который добавляет поле «процентная ставка» и метод для начисления процентов

ТАБЛИЦА ПРАВИЛЬНЫХ ОТВЕТОВ

Тестовая часть

№ вопроса	Правильный ответ	Баллы
1	b	1
2	b	1
3	b	1
4	c	1
5	a	1
6	a	1
7	a	1

№ вопроса	Правильный ответ	Баллы
8	a	1
9	a	1
10	a	1
11	a	1
12	a	1
Соответствие 1	C	1
Соответствие 2	A	1
Соответствие 3	B	1
Соответствие 4	D	1
Соответствие 5	E	1
Итого тест		17

Практическая часть (21.5 балла за каждую задачу)

Задача	Максимум баллов	Критерии оценивания
1	21.5	Объявление класса (5), конструкторы (5), методы (6), работа (5.5)
2	21.5	Объявление класса (5), методы (6), работа с объектами (6), интеграция (4.5)
Итого практика	43	

ВСЕГО ЗА РАБОТУ: 60 баллов

ЭТАЛОННЫЕ РЕШЕНИЯ ПРАКТИЧЕСКИХ ЗАДАЧ ВАРИАНТ 1

Задача 1: Класс Student

```
#include <iostream>
#include <string>
using namespace std;

class Student {
private:
    string firstName;
    string lastName;
    double averageGrade;

public:
    // Конструкторы
    Student() : firstName(""), lastName(""), averageGrade(0.0) {}

    Student(string first, string last, double grade)
        : firstName(first), lastName(last), averageGrade(grade) {}

    // Геттеры
    string getFirstName() const { return firstName; }
    string getLastName() const { return lastName; }
    double getAverageGrade() const { return averageGrade; }

    // Сеттеры
    void setFirstName(string first) { firstName = first; }
    void setLastName(string last) { lastName = last; }
    void setAverageGrade(double grade) {
        if (grade >= 0.0 && grade <= 5.0) {
            averageGrade = grade;
        }
    }

    // Метод для вывода информации
    void displayInfo() const {
        cout << "Студент: " << firstName << " " << lastName
            << ", средний балл: " << averageGrade << endl;
    }
};
```

Задача 2: Класс Group

```
class Group {
private:
    static const int MAX_STUDENTS = 30;
    Student students[MAX_STUDENTS];
    int studentCount;

public:
    Group() : studentCount(0) {}
```

```

// Метод для добавления студента
void addStudent(const Student& student) {
    if (studentCount < MAX_STUDENTS) {
        students[studentCount] = student;
        studentCount++;
        cout << "Студент добавлен в группу" << endl;
    } else {
        cout << "Группа заполнена!" << endl;
    }
}

// Метод для вывода всех студентов
void displayAllStudents() const {
    cout << "Список студентов в группе:" << endl;
    for (int i = 0; i < studentCount; i++) {
        students[i].displayInfo();
    }
}

// Метод для поиска студента по фамилии
void findStudentByLastName(const string& lastName) const {
    bool found = false;
    for (int i = 0; i < studentCount; i++) {
        if (students[i].getLastName() == lastName) {
            students[i].displayInfo();
            found = true;
        }
    }
    if (!found) {
        cout << "Студент с фамилией '" << lastName << "' не найден" << endl;
    }
}

};

// Пример использования
int main() {
    Student student1("Иван", "Петров", 4.5);
    Student student2("Мария", "Иванова", 4.8);

    Group group;
    group.addStudent(student1);
    group.addStudent(student2);

    group.displayAllStudents();
    group.findStudentByLastName("Иванова");

    return 0;
}

```

ВАРИАНТ 2

Задача 1: Класс BankAccount

```
#include <iostream>
#include <string>
using namespace std;

class BankAccount {
protected:
    string accountNumber;
    double balance;
    string owner;

public:
    // Конструкторы
    BankAccount(string accNum, double bal, string own)
        : accountNumber(accNum), balance(bal), owner(own) {}

    // Геттеры
    string getAccountNumber() const { return accountNumber; }
    double getBalance() const { return balance; }
    string getOwner() const { return owner; }

    // Метод для пополнения счета
    void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
            cout << "Счет пополнен на " << amount << " руб." << endl;
        } else {
            cout << "Неверная сумма для пополнения" << endl;
        }
    }

    // Метод для снятия средств
    virtual bool withdraw(double amount) {
        if (amount > 0 && amount <= balance) {
            balance -= amount;
            cout << "Со счета снято " << amount << " руб." << endl;
            return true;
        } else {
            cout << "Недостаточно средств или неверная сумма" << endl;
            return false;
        }
    }

    // Метод для вывода информации
    virtual void displayInfo() const {
        cout << "Владелец: " << owner << endl;
        cout << "Номер счета: " << accountNumber << endl;
        cout << "Баланс: " << balance << " руб." << endl;
    }
};
```

Задача 2: Класс SavingsAccount

```
class SavingsAccount : public BankAccount {
private:
    double interestRate; // Процентная ставка

public:
    // Конструктор
    SavingsAccount(string accNum, double bal, string own, double rate)
        : BankAccount(accNum, bal, own), interestRate(rate) {}

    // Метод для начисления процентов
    void applyInterest() {
        double interest = balance * interestRate / 100.0;
        balance += interest;
        cout << "Начислены проценты: " << interest << " руб." << endl;
    }

    // Переопределенный метод вывода информации
    void displayInfo() const override {
        BankAccount::displayInfo();
        cout << "Процентная ставка: " << interestRate << "%" << endl;
    }

    // Геттер для процентной ставки
    double getInterestRate() const { return interestRate; }

    // Сеттер для процентной ставки
    void setInterestRate(double rate) {
        if (rate >= 0) {
            interestRate = rate;
        }
    }
};

// Пример использования
int main() {
    BankAccount account1("123456789", 1000.0, "Иван Петров");
    SavingsAccount savings1("987654321", 5000.0, "Мария Иванова", 5.0);

    account1.displayInfo();
    cout << "-----" << endl;

    savings1.displayInfo();
    savings1.applyInterest();
    savings1.displayInfo();

    return 0;
}
```

ШКАЛА ОЦЕНОК

Баллы	Оценка	Уровень выполнения
54-60	5 (Отлично)	Все классы реализованы корректно, демонстрируется понимание ООП
42-53	4 (Хорошо)	Классы работают, возможны мелкие недочеты в реализации
30-41	3 (Удовлетворительно)	Основной функционал реализован, но есть ошибки в ООП
0-29	2 (Неудовлетворительно)	Значительные ошибки в реализации классов

КРИТЕРИИ ОЦЕНИВАНИЯ ПРАКТИЧЕСКОЙ ЧАСТИ

- **Объявление класса** (5 баллов): правильная структура, модификаторы доступа
- **Конструкторы** (5 баллов): конструктор по умолчанию, с параметрами
- **Методы** (6 баллов): геттеры, сеттеры, бизнес-логика
- **Работа программы** (5.5 баллов): программа компилируется и работает корректно
- **Работа с объектами** (6 баллов): создание объектов, вызов методов
- **Интеграция** (4.5 баллов): классы корректно взаимодействуют друг с другом
- **Наследование и полиморфизм** (6 баллов): правильное использование наследования, виртуальных функций
- **Читаемость кода** (5 баллов): понятные имена, комментарии, форматирование

3. ЭКЗАМЕНАЦИОННЫЕ БИЛЕТЫ

Раздел 1. Введение в программирование

Этот раздел охватывает базовые понятия, без которых невозможно начать писать программы.

Теоретические вопросы:

1. Типы данных в языке программирования
2. Структура программы
3. Оператор чтения (ввода)
4. Оператор записи (вывода)
5. Математические стандартные функции ЯП

Практические задания:

1. Вычислить значение функции (требует знания структуры программы, типов данных, математических функций и ввода/вывода).
2. Вычислить значение выражения по формуле $2-x - \cos x + \sin (2xy)$ (аналогично предыдущему).
3. Составить программу вычисления объема цилиндра (демонстрирует применение базовых математических вычислений).
4. Даны два действительных числа... (работа с переменными, базовые арифметические операции и ветвления).
5. Даны три действительных числа... (ветвления и возведение в степень).
6. Дана длина ребра куба... (простые вычисления и вывод).

Раздел 2. Операторы языка программирования

Этот раздел посвящен управляющим конструкциям, которые определяют логику выполнения программы.

Теоретические вопросы:

6. Ветвления в программах
7. Использование составного оператора
8. Оператор выбора
9. Цикл с предусловием
10. Цикл с постусловием
11. Цикл с параметром

Практические задания:

7. По данному числу выводить название месяца (идеальная задача для оператора выбора).
8. По номеру дня недели выдать количество уроков (аналогично предыдущему, оператор выбора или ветвление).
9. Составить программу-генератор простых чисел (использование цикла для перебора значений и проверки условия).
10. Дано натуральное n . Вычислить... (классическая задача на использование цикла для накопления суммы/произведения).

Раздел 3. Структурное и модульное программирование

Этот раздел учит разбивать сложную задачу на подзадачи, используя подпрограммы и работая с файлами.

Теоретические вопросы:

12. Подпрограммы–функции
13. Подпрограммы-процедуры
14. Программирование рекуррентных последовательностей - арифметической прогрессии
15. Программирование рекуррентных последовательностей – геометрической прогрессии
16. Программирование рекуррентных последовательностей – вывод чисел Фибоначчи

17. Программирование рекуррентных последовательностей – нахождение факториала целого числа
27. Файлы и файловые переменные
28. Правило создания и заполнения файлов
29. Правило чтения из файлов
30. Текстовые файлы
31. Комбинированный тип данных
32. Работа с файлами записей
39. Внешние подпрограммы
40. Модули

Практические задания:

11. Найти 10-ый элемент геометрической прогрессии (рекуррентная последовательность, цикл).
12. Найти 20-й элемент арифметической прогрессии (рекуррентная последовательность, цикл).
13. Нахождение НОД с применением **процедуры**.
14. Нахождение НОД с применением **функции**.
15. Работа с файлами: создание, заполнение, чтение, подсчет компонент (требует знания теории по файлам).
16. Работа с записями (комбинированный тип данных): поиск и фильтрация по полям (фамилия, зарплата, месяц рождения).

Раздел 4. Основные конструкции языков программирования

Этот раздел охватывает сложные структуры данных: строки, массивы, множества и динамические структуры.

Теоретические вопросы:

18. Строковый тип данных
19. Функции со строками
20. Процедуры со строками
21. Стандартные функции ord(x) и chr(x)
22. Одномерные массивы
23. Сортировка массивов «методом пузырька»
24. Двумерные массивы
25. Множественный тип данных
26. Операции над множествами
33. Указатели и динамические структуры
34. Связь между динамическими величинами и их указателями
35. Связанные списки
36. Понятие кольцевого списка
37. Понятие очереди
38. Понятие дерева

Практические задания:

Тема: Строки и символы

17. Пользователь вводит строку. Напишите программу, которая подсчитывает количество прописных (заглавных) и строчных букв в этой строке. Используйте функции ord или chr для проверки кодов символов.
18. "Шифр Цезаря". Напишите программу, которая шифрует введенную пользователем строку с помощью шифра Цезаря (каждый символ сдвигается на фиксированное число позиций в алфавите). Сдвиг задается пользователем. Учитывайте только буквы английского алфавита, цифры и знаки препинания оставляйте без изменений. Используйте функции ord и chr.

19. Напишите программу, которая в заданной строке находит все вхождения слова «кот» и заменяет его на «пёс». Выведите результат на экран.

Тема: Одномерные массивы

20. Дан одномерный массив из 15 целых чисел, заполненный случайными числами в диапазоне от -50 до 50. Найдите и выведите на экран минимальный и максимальный элемент массива, а также их индексы.

21. Заполните массив из 10 элементов случайными числами. Выведите его на экран. Переставьте элементы массива так, чтобы все четные числа оказались в начале массива, а все нечетные — в конце. Относительный порядок четных и нечетных чисел может не сохраняться. Выведите результат на экран.

Пример: [3, 8, 1, 6, 2] -> [8, 6, 2, 3, 1]

22. В массиве из N вещественных чисел (введите с клавиатуры) найдите сумму элементов, расположенных между первым и последним отрицательными элементами. Если отрицательных чисел меньше двух, выведите сообщение об ошибке.

Тема: Двумерные массивы (матрицы)

23. Дана матрица размером 5x5, заполненная целыми числами. Вычислите и выведите одномерные массивы, содержащие:

- а) сумму элементов каждой строки.
- б) сумму элементов каждого столбца.

24. Для квадратной матрицы 6x6, заполненной случайными числами, найдите:

- а) Сумму элементов главной диагонали.
- б) Произведение элементов побочной диагонали.
- в) Выведите саму матрицу и найденные значения.

25. Напишите программу, которая создает матрицу A размером 3x4, заполняет ее случайными числами и выводит на экран. Затем программа должна создать и вывести транспонированную матрицу AT (где строки становятся столбцами, а столбцы — строками).

Тема: Множества

26. Пользователь вводит два слова с клавиатуры. Напишите программу, которая с помощью операций над множествами определяет и выводит на экран множество букв, которые встречаются одновременно в обоих словах.

27. Пользователь вводит строку, содержащую буквы и цифры. Напишите программу, которая формирует множество всех цифр, встречающихся в этой строке, и выводит их в отсортированном по возрастанию виде.

Пример: Ввод: "abc123def45", Вывод: {1, 2, 3, 4, 5}

Тема: Динамические структуры данных (Связные списки)

28. Смоделируйте работу простой очереди с помощью связанного списка. Напишите программу, которая предоставляет пользователю возможность:

- Добавить элемент в конец очереди.
- Удалить элемент из начала очереди.
- Вывести все элементы очереди на экран.
- Завершить работу.

Программа должна работать в цикле до тех пор, пока пользователь не выберет завершение. Для этого создайте структуру Узел (Node) с полями data и next.

Раздел 5. Объектно-ориентированное программирование

Этот раздел знакомит с современной парадигмой программирования, основанной на объектах и классах.

Теоретические вопросы:

- 40. Основные понятия объектно-ориентированного программирования
- 41. Понятие полиморфизма
- 42. Понятие наследования
- 43. Понятие инкапсуляции

Практические задания:

Задачи на Инкапсуляцию и создание классов:

29. Класс Book (Книга)

Создай класс Book, который будет хранить информацию о книге.

- **Поля (private):** title (название), author (автор), year (год издания).
- **Методы (public):**
 - Конструктор для инициализации полей.
 - Геттеры и сеттеры для каждого поля.
 - getInfo(), который возвращает строку с полной информацией о книге (например, " '1984', Джордж Оруэлл, 1949").

Задача: В основной программе создай 2-3 объекта класса Book и выведи о них информацию.

30. Класс BankAccount (Банковский счет)

Создай класс BankAccount, моделирующий банковский счет.

Поля (private): accountNumber (номер счета), balance (баланс).

Методы (public):

1. Конструктор (номер счета задается при создании, баланс начинается с 0).
2. deposit(amount) — положить деньги на счет.
3. withdraw(amount) — снять деньги со счета (не позволять снять больше, чем на счете есть).
4. getBalance() — вернуть текущий баланс.

Задача: Создай счет, положи на него 1000, попробуй снять 500, потом попробуй снять 700. Выводи баланс после каждой операции.

31. Класс Student (Студент)

Создай класс Student.

Поля (private): name (имя), grades (список оценок).

Методы (public):

1. Конструктор (принимает имя, инициализирует пустой список оценок).
2. addGrade(grade) — добавить оценку (от 2 до 5).
3. getAverageGrade() — вычислить и вернуть средний балл студента.

Задача: Создай студента, добавь ему 5 оценок и выведи его средний балл.

Задачи на Наследование:

32. Классы Animal (Животное) и Dog (Собака) / Cat (Кот)

Создай базовый класс Animal.

Поля (protected): name (кличка).

Методы:

1. Конструктор.
2. makeSound() — выводит на экран "Какой-то звук...".

Создай производный класс Dog.

Методы: Переопредели метод makeSound(), чтобы он выводил "Гав!".

Задача: Создай объекты класса Animal и Dog. Вызови метод makeSound() у каждого и убедись, что они "звучат" по-разному.

33. Классы Vehicle (Транспорт) и Car (Автомобиль)

Создай базовый класс Vehicle.

Поля: brand (марка), speed (текущая скорость).

Методы: Конструктор, displayInfo() (выводит марку и скорость).

Создай производный класс Car, который добавляет:

Поле: fuelLevel (уровень топлива).

Метод: Переопредели displayInfo(), чтобы он также выводил уровень топлива.

Задача: Создай автомобиль и выведи о нем полную информацию.

Задачи на Полиморфизм (в его простейшей форме - переопределение методов):

34. Классы Shape (Фигура) и Rectangle/Circle (Прямоугольник/Круг)

Создай базовый абстрактный класс Shape с

методом calculateArea() (вычисли площадь).

Создай класс Rectangle (наследуется от Shape) с полями width и height.

Реализуй метод calculateArea().

Создай класс Circle (наследуется от Shape) с полем radius. Реализуй метод calculateArea().

Задача: В основной программе создай массив разных фигур (Rectangle и Circle). В цикле вызови для каждой метод calculateArea() и выведи результат. Это демонстрирует полиморфизм: одна и та же операция (вычисление площади) выполняется по-разному для разных объектов.

3.2. Эталонные ответы экзаменационных билетов. Практическая часть

Раздел 1. Введение в программирование

Задача 1. Вычисление значения функции

```
#include <iostream>
#include <cmath>
using namespace std;

int main() {
    double x, y;

    cout << "Введите x: ";
    cin >> x;

    y = pow(x, 2) - 2*x + 1;

    cout << "При x = " << x << ", y = " << y << endl;
    return 0;
}
```

Задача 2. Вычисление сложного выражения

```
#include <iostream>
#include <cmath>
using namespace std;

int main() {
    double x, y, result;

    cout << "Введите x: ";
    cin >> x;
    cout << "Введите y: ";
    cin >> y;

    result = 2 - x - cos(x) + sin(2*x*y);

    cout << "Результат: " << result << endl;
    return 0;
}
```

Задача 3. Объем цилиндра

```
#include <iostream>
#include <cmath>
using namespace std;

int main() {
    double radius, height, volume;
    const double PI = 3.14159;

    cout << "Введите радиус цилиндра: ";
    cin >> radius;
    cout << "Введите высоту цилиндра: ";
    cin >> height;
```

```

    volume = PI * pow(radius, 2) * height;

    cout << "Объем цилиндра: " << volume << endl;
    return 0;
}

```

Задача 4. Работа с двумя числами

```

#include <iostream>
using namespace std;

int main() {
    double a, b;

    cout << "Введите первое число: ";
    cin >> a;
    cout << "Введите второе число: ";
    cin >> b;

    if (a != b) {
        if (a < b) {
            a = (a + b) / 2;
            b = 2 * a * b;
        } else {
            b = (a + b) / 2;
            a = 2 * a * b;
        }
    }

    cout << "Результат: a = " << a << ", b = " << b << endl;
    return 0;
}

```

Задача 5. Возведение в степень

```

#include <iostream>
#include <cmath>
using namespace std;

int main() {
    double numbers[3];

    for (int i = 0; i < 3; i++) {
        cout << "Введите число " << i+1 << ": ";
        cin >> numbers[i];
    }

    cout << "Результаты: ";
    for (int i = 0; i < 3; i++) {
        if (numbers[i] >= 0) {
            cout << pow(numbers[i], 2) << " ";
        } else {
            cout << pow(numbers[i], 4) << " ";
        }
    }
}

```

```

    }
}
cout << endl;

return 0;
}

```

Задача 6. Параметры куба

```

#include <iostream>
#include <cmath>
using namespace std;

int main() {
    double edge, face_area, total_area, volume;

    cout << "Введите длину ребра куба: ";
    cin >> edge;

    face_area = pow(edge, 2);
    total_area = 6 * face_area;
    volume = pow(edge, 3);

    cout << "Площадь грани: " << face_area << endl;
    cout << "Площадь полной поверхности: " << total_area << endl;
    cout << "Объем куба: " << volume << endl;

    return 0;
}

```

Раздел 2. Операторы языка программирования

Задача 7. Название месяца

```

#include <iostream>
using namespace std;

int main() {
    int month;

    cout << "Введите номер месяца (1-12): ";
    cin >> month;

    switch (month) {
        case 1: cout << "Январь"; break;
        case 2: cout << "Февраль"; break;
        case 3: cout << "Март"; break;
        case 4: cout << "Апрель"; break;
        case 5: cout << "Май"; break;
        case 6: cout << "Июнь"; break;
        case 7: cout << "Июль"; break;
        case 8: cout << "Август"; break;
        case 9: cout << "Сентябрь"; break;
        case 10: cout << "Октябрь"; break;
    }
}

```

```

        case 11: cout << "Ноябрь"; break;
        case 12: cout << "Декабрь"; break;
        default: cout << "Неверный номер месяца!";
    }
    cout << endl;

    return 0;
}

```

Задача 8. Количество уроков

```

#include <iostream>
using namespace std;

int main() {
    int day, lessons;

    cout << "Введите номер дня недели (1-7): ";
    cin >> day;

    switch (day) {
        case 1: lessons = 5; break; // Понедельник
        case 2: lessons = 6; break; // Вторник
        case 3: lessons = 4; break; // Среда
        case 4: lessons = 6; break; // Четверг
        case 5: lessons = 5; break; // Пятница
        case 6: lessons = 0; break; // Суббота
        case 7: lessons = 0; break; // Воскресенье
        default: lessons = -1;
    }

    if (lessons >= 0) {
        cout << "Количество уроков: " << lessons << endl;
    } else {
        cout << "Неверный номер дня недели!" << endl;
    }

    return 0;
}

```

Задача 9. Генератор простых чисел

```

#include <iostream>
#include <cmath>
using namespace std;

bool isPrime(int n) {
    if (n <= 1) return false;
    if (n == 2) return true;
    if (n % 2 == 0) return false;

    for (int i = 3; i <= sqrt(n); i += 2) {
        if (n % i == 0) return false;
    }
}

```

```

    return true;
}

int main() {
    cout << "Простые числа по формуле  $2x^2 + 29$  ( $0 \leq x \leq 28$ ):" << endl;

    for (int x = 0; x <= 28; x++) {
        int number = 2 * x * x + 29;
        if (isPrime(number)) {
            cout << "x = " << x << ", число = " << number << " - простое" << endl;
        }
    }

    return 0;
}

```

Задача 10. Сумма гармонического ряда

```

#include <iostream>
using namespace std;

int main() {
    int n;
    double total_sum = 0.0;

    cout << "Введите натуральное n: ";
    cin >> n;

    for (int i = 1; i <= n; i++) {
        total_sum += 1.0 / i;
    }

    cout << "Сумма ряда: " << total_sum << endl;
    return 0;
}

```

Раздел 3. Структурное и модульное программирование

Задача 11. Геометрическая прогрессия

```

#include <iostream>
using namespace std;

int main() {
    int first_element = 3;
    int step = 2;
    int n = 10;
    int current = first_element;

    for (int i = 1; i < n; i++) {
        current *= step;
    }

    cout << n << "-й элемент геометрической прогрессии: " << current << endl;
}

```

```
    return 0;
}
```

Задача 12. Арифметическая прогрессия

```
#include <iostream>
using namespace std;
```

```
int main() {
    int first_element = 4;
    int step = 3;
    int n = 20;
    int current = first_element;

    for (int i = 1; i < n; i++) {
        current += step;
    }

    cout << n << "-й элемент арифметической прогрессии: " << current << endl;
    return 0;
}
```

Задача 13. НОД с процедурой

```
#include <iostream>
using namespace std;
```

// Процедура (void функция)

```
void findGCD(int a, int b) {
    int temp;
    while (b != 0) {
        temp = b;
        b = a % b;
        a = temp;
    }
    cout << "НОД равен " << a << endl;
}
```

```
int main() {
    int num1, num2;

    cout << "Введите первое число: ";
    cin >> num1;
    cout << "Введите второе число: ";
    cin >> num2;

    findGCD(num1, num2);
    return 0;
}
```

Задача 14. НОД с функцией

```
#include <iostream>
using namespace std;

// Функция, возвращающая значение
int gcdFunction(int a, int b) {
    int temp;
    while (b != 0) {
        temp = b;
        b = a % b;
        a = temp;
    }
    return a;
}

int main() {
    int num1, num2;

    cout << "Введите первое число: ";
    cin >> num1;
    cout << "Введите второе число: ";
    cin >> num2;

    int result = gcdFunction(num1, num2);
    cout << "НОД чисел " << num1 << " и " << num2 << " равен " << result << endl;

    return 0;
}
```

Задача 15. Работа с файлами

```
#include <iostream>
#include <fstream>
#include <cstdlib>
#include <ctime>
using namespace std;

int main() {
    ofstream outFile("numbers.txt");
    srand(time(0));

    // Запись в файл
    for (int i = 0; i < 10; i++) {
        int number = rand() % 100 + 1;
        outFile << number << endl;
    }
    outFile.close();

    // Чтение из файла
    ifstream inFile("numbers.txt");
    int number, evenCount = 0;
```

```

while (inFile >> number) {
    if (number % 2 == 0) {
        evenCount++;
    }
}
inFile.close();

cout << "Количество четных чисел: " << evenCount << endl;
return 0;
}

```

Задача 16. Работа со структурами

```

#include <iostream>
#include <string>
#include <vector>
using namespace std;

struct Worker {
    string last_name;
    double salary;
    int birth_month;
};

int main() {
    vector<Worker> workers = {
        {"Иванов", 50000, 3},
        {"Петров", 45000, 7},
        {"Сидоров", 60000, 3},
        {"Алексеев", 55000, 5}
    };

    int target_month;
    cout << "Введите месяц рождения для поиска: ";
    cin >> target_month;

    cout << "Работники, родившиеся в " << target_month << " месяце:" << endl;
    for (const auto& worker : workers) {
        if (worker.birth_month == target_month) {
            cout << worker.last_name << ", зарплата: " << worker.salary << " руб." << endl;
        }
    }

    return 0;
}

```

Раздел 4. Основные конструкции языков программирования

Задача 17. Подсчет регистров букв

```

#include <iostream>
#include <string>
#include <cctype>
using namespace std;

```

```

int main() {
    string text;
    int upper_count = 0, lower_count = 0;

    cout << "Введите строку: ";
    getline(cin, text);

    for (char c : text) {
        if (isupper(c)) upper_count++;
        else if (islower(c)) lower_count++;
    }

    cout << "Прописных букв: " << upper_count << endl;
    cout << "Строчных букв: " << lower_count << endl;

    return 0;
}

```

Задача 18. Шифр Цезаря

```

#include <iostream>
#include <string>
using namespace std;

string caesarCipher(string text, int shift) {
    string result = "";

    for (char c : text) {
        if (isalpha(c)) {
            char base = isupper(c) ? 'A' : 'a';
            result += char((c - base + shift) % 26 + base);
        } else {
            result += c;
        }
    }

    return result;
}

int main() {
    string text;
    int shift;

    cout << "Введите текст для шифрования: ";
    getline(cin, text);
    cout << "Введите сдвиг: ";
    cin >> shift;

    string encrypted = caesarCipher(text, shift);
    cout << "Зашифрованный текст: " << encrypted << endl;

    return 0;
}

```

Задача 19. Замена слова в строке

```
#include <iostream>
#include <string>
using namespace std;

string replaceWord(string text, const string& oldWord, const string& newWord) {
    size_t pos = 0;
    while ((pos = text.find(oldWord, pos)) != string::npos) {
        text.replace(pos, oldWord.length(), newWord);
        pos += newWord.length();
    }
    return text;
}

int main() {
    string text;
    cout << "Введите текст: ";
    getline(cin, text);

    string newText = replaceWord(text, "кот", "пёс");
    cout << "Результат замены: " << newText << endl;

    return 0;
}
```

Задача 20. Поиск min и max в массиве

```
#include <iostream>
#include <cstdlib>
#include <ctime>
using namespace std;

int main() {
    const int SIZE = 15;
    int arr[SIZE];

    srand(time(0));
    for (int i = 0; i < SIZE; i++) {
        arr[i] = rand() % 101 - 50; // от -50 до 50
    }

    cout << "Массив: ";
    for (int i = 0; i < SIZE; i++) {
        cout << arr[i] << " ";
    }
    cout << endl;

    int min_val = arr[0], max_val = arr[0];
    int min_index = 0, max_index = 0;

    for (int i = 1; i < SIZE; i++) {
        if (arr[i] < min_val) {
```

```

        min_val = arr[i];
        min_index = i;
    }
    if (arr[i] > max_val) {
        max_val = arr[i];
        max_index = i;
    }
}

cout << "Минимальный элемент: " << min_val << " (индекс " << min_index << ")" <<
endl;
cout << "Максимальный элемент: " << max_val << " (индекс " << max_index << ")" <<
endl;

return 0;
}

```

Задача 21. Разделение четных и нечетных

```

#include <iostream>
#include <cstdlib>
#include <ctime>
using namespace std;

int main() {
    const int SIZE = 10;
    int arr[SIZE];

    srand(time(0));
    for (int i = 0; i < SIZE; i++) {
        arr[i] = rand() % 100 + 1;
    }

    cout << "Исходный массив: ";
    for (int i = 0; i < SIZE; i++) {
        cout << arr[i] << " ";
    }
    cout << endl;

    // Разделение
    int result[SIZE];
    int even_index = 0, odd_index = SIZE - 1;

    for (int i = 0; i < SIZE; i++) {
        if (arr[i] % 2 == 0) {
            result[even_index++] = arr[i];
        } else {
            result[odd_index--] = arr[i];
        }
    }

    cout << "Результат: ";
    for (int i = 0; i < SIZE; i++) {

```

```

        cout << result[i] << " ";
    }
    cout << endl;

    return 0;
}

```

Задача 22. Сумма между отрицательными элементами

```

#include <iostream>
using namespace std;

int main() {
    int n;
    cout << "Введите размер массива: ";
    cin >> n;

    double* arr = new double[n];

    for (int i = 0; i < n; i++) {
        cout << "Введите элемент " << i+1 << ": ";
        cin >> arr[i];
    }

    cout << "Массив: ";
    for (int i = 0; i < n; i++) {
        cout << arr[i] << " ";
    }
    cout << endl;

    int first_neg = -1, last_neg = -1;

    for (int i = 0; i < n; i++) {
        if (arr[i] < 0) {
            if (first_neg == -1) first_neg = i;
            last_neg = i;
        }
    }

    if (first_neg != -1 && last_neg != -1 && first_neg != last_neg) {
        double total = 0;
        for (int i = first_neg + 1; i < last_neg; i++) {
            total += arr[i];
        }
        cout << "Сумма между первым и последним отрицательными: " << total << endl;
    } else {
        cout << "Недостаточно отрицательных элементов!" << endl;
    }

    delete[] arr;
    return 0;
}

```

Задача 23. Суммы строк и столбцов матрицы

```
#include <iostream>
#include <cstdlib>
#include <ctime>
using namespace std;

int main() {
    const int SIZE = 5;
    int matrix[SIZE][SIZE];

    srand(time(0));
    for (int i = 0; i < SIZE; i++) {
        for (int j = 0; j < SIZE; j++) {
            matrix[i][j] = rand() % 10 + 1;
        }
    }

    cout << "Матрица:" << endl;
    for (int i = 0; i < SIZE; i++) {
        for (int j = 0; j < SIZE; j++) {
            cout << matrix[i][j] << " ";
        }
        cout << endl;
    }

    // Суммы строк
    cout << "Суммы строк: ";
    for (int i = 0; i < SIZE; i++) {
        int row_sum = 0;
        for (int j = 0; j < SIZE; j++) {
            row_sum += matrix[i][j];
        }
        cout << row_sum << " ";
    }
    cout << endl;

    // Суммы столбцов
    cout << "Суммы столбцов: ";
    for (int j = 0; j < SIZE; j++) {
        int col_sum = 0;
        for (int i = 0; i < SIZE; i++) {
            col_sum += matrix[i][j];
        }
        cout << col_sum << " ";
    }
    cout << endl;

    return 0;
}
```

Раздел 5. Объектно-ориентированное программирование

Задача 29. Класс Book

```
#include <iostream>
#include <string>
using namespace std;

class Book {
private:
    string title;
    string author;
    int year;

public:
    // Конструктор
    Book(string t, string a, int y) : title(t), author(a), year(y) {}

    // Геттеры
    string getTitle() const { return title; }
    string getAuthor() const { return author; }
    int getYear() const { return year; }

    // Сеттеры
    void setTitle(string t) { title = t; }
    void setAuthor(string a) { author = a; }
    void setYear(int y) { year = y; }

    // Метод для получения информации
    string getInfo() const {
        return "\"" + title + "\", " + author + ", " + to_string(year);
    }
};

int main() {
    Book book1("1984", "Джордж Оруэлл", 1949);
    Book book2("Преступление и наказание", "Федор Достоевский", 1866);

    cout << book1.getInfo() << endl;
    cout << book2.getInfo() << endl;

    return 0;
}
```

Задача 30. Класс BankAccount

```
#include <iostream>
#include <string>
using namespace std;

class BankAccount {
private:
    string accountNumber;
```

```

double balance;

public:
    BankAccount(string accNum) : accountNumber(accNum), balance(0) {}

    void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
            cout << "Внесено: " << amount << endl;
        }
    }

    bool withdraw(double amount) {
        if (amount > 0 && amount <= balance) {
            balance -= amount;
            cout << "Снято: " << amount << endl;
            return true;
        } else {
            cout << "Недостаточно средств!" << endl;
            return false;
        }
    }

    double getBalance() const {
        return balance;
    }

    string getAccountNumber() const {
        return accountNumber;
    }
};

int main() {
    BankAccount account("123456789");

    account.deposit(1000);
    cout << "Баланс: " << account.getBalance() << endl;

    account.withdraw(500);
    cout << "Баланс: " << account.getBalance() << endl;

    account.withdraw(700);
    cout << "Баланс: " << account.getBalance() << endl;

    return 0;
}

```

Задача 31. Класс Student

```

#include <iostream>
#include <string>
#include <vector>
using namespace std;

```

```

class Student {
private:
    string name;
    vector<int> grades;

public:
    Student(string n) : name(n) {}

    void addGrade(int grade) {
        if (grade >= 2 && grade <= 5) {
            grades.push_back(grade);
        }
    }

    double getAverageGrade() const {
        if (grades.empty()) return 0;

        double sum = 0;
        for (int grade : grades) {
            sum += grade;
        }
        return sum / grades.size();
    }

    string getName() const {
        return name;
    }
};

int main() {
    Student student("Иван Петров");

    student.addGrade(5);
    student.addGrade(4);
    student.addGrade(3);
    student.addGrade(5);
    student.addGrade(4);

    cout << "Средний балл студента " << student.getName() << ": "
         << student.getAverageGrade() << endl;

    return 0;
}

```

Задача 32. Наследование Animal → Dog

```

#include <iostream>
#include <string>
using namespace std;

class Animal {
protected:
    string name;

```

```

public:
    Animal(string n) : name(n) {}

    virtual void makeSound() {
        cout << "Какой-то звук..." << endl;
    }
};

class Dog : public Animal {
public:
    Dog(string n) : Animal(n) {}

    void makeSound() override {
        cout << "Гав! Гав!" << endl;
    }
};

class Cat : public Animal {
public:
    Cat(string n) : Animal(n) {}

    void makeSound() override {
        cout << "Мяу!" << endl;
    }
};

int main() {
    Animal animal("Неизвестное животное");
    Dog dog("Бобик");
    Cat cat("Мурка");

    animal.makeSound();
    dog.makeSound();
    cat.makeSound();

    return 0;
}

```

Задача 33. Наследование Vehicle → Car

```

#include <iostream>
#include <string>
using namespace std;

class Vehicle {
protected:
    string brand;
    double speed;

public:
    Vehicle(string b, double s) : brand(b), speed(s) {}

```

```

    virtual void displayInfo() {
        cout << "Марка: " << brand << ", Скорость: " << speed << " км/ч" << endl;
    }
};

class Car : public Vehicle {
private:
    double fuelLevel;

public:
    Car(string b, double s, double fuel) : Vehicle(b, s), fuelLevel(fuel) {}

    void displayInfo() override {
        cout << "Марка: " << brand << ", Скорость: " << speed
            << " км/ч, Уровень топлива: " << fuelLevel << "%" << endl;
    }
};

int main() {
    Car car("Toyota", 120, 65.5);
    car.displayInfo();

    return 0;
}

```

Задача 34. Полиморфизм с фигурами

```

#include <iostream>
#include <cmath>
using namespace std;

class Shape {
public:
    virtual double calculateArea() = 0; // Чисто виртуальная функция
    virtual ~Shape() {} // Виртуальный деструктор
};

class Rectangle : public Shape {
private:
    double width, height;

public:
    Rectangle(double w, double h) : width(w), height(h) {}

    double calculateArea() override {
        return width * height;
    }
};

class Circle : public Shape {
private:
    double radius;

```

```

public:
    Circle(double r) : radius(r) {}

    double calculateArea() override {
        return M_PI * radius * radius;
    }
};

int main() {
    const int SIZE = 3;
    Shape* shapes[SIZE];

    shapes[0] = new Rectangle(5, 3);
    shapes[1] = new Circle(2);
    shapes[2] = new Rectangle(4, 4);

    for (int i = 0; i < SIZE; i++) {
        cout << "Площадь фигуры " << i+1 << ": " << shapes[i]->calculateArea() << endl;
    }

    // Освобождение памяти
    for (int i = 0; i < SIZE; i++) {
        delete shapes[i];
    }

    return 0;
}

```

Все программы компилируются и работают в стандартной среде C++. Они охватывают все основные темы курса "Основы алгоритмизации и программирования".

3.3. Эталонные ответы экзаменационных билетов. Теоретическая часть

Раздел 1. Введение в программирование

1. Типы данных в языке программирования

Эталонный ответ:

Типы данных определяют множество значений, которые может принимать переменная, и операции, которые можно выполнять над этими значениями. Основные типы данных в C++:

1. Целочисленные типы:

- int - целые числа (-2×10^9 до 2×10^9)
- short - короткие целые
- long - длинные целые
- char - символьный тип

2. Вещественные типы:

- float - числа с плавающей точкой (7 значащих цифр)
- double - двойной точности (15 значащих цифр)

3. Логический тип:

- bool - принимает значения true/false

4. Производные типы:

- Массивы, указатели, ссылки, структуры, классы

Пример:

```
int age = 25;
double salary = 1500.75;
char grade = 'A';
bool isActive = true;
```

2. Структура программы

Эталонный ответ:

Стандартная структура программы на C++ включает:

1. **Директивы препроцессора** - подключение библиотек
2. **Объявление пространств имен** - using namespace std
3. **Глобальные объявления** - константы, прототипы функций
4. **Функция main()** - главная функция программы
5. **Определения функций** - реализация пользовательских функций

Пример структуры:

```
#include <iostream>    // 1. Директива препроцессора
using namespace std;  // 2. Пространство имен
```

```
// 3. Прототипы функций
```

```
void printMessage();
```

```
int main() {           // 4. Функция main()
```

```
    cout << "Hello World!";
    printMessage();
    return 0;
```

```
}
```

```
void printMessage() {  // 5. Определение функции
```

```
    cout << "Additional message";
```

```
}
```

3. Оператор чтения (ввода)

Эталонный ответ:

Оператор чтения используется для ввода данных с клавиатуры. В C++ для этого используется объект `cin` (console input) из библиотеки `<iostream>`.

Синтаксис:

```
cin >> variable;
```

Примеры:

```
int age;
```

```
double salary;
```

```
string name;
```

```
cin >> age;    // Ввод целого числа
```

```
cin >> salary; // Ввод вещественного числа
```

```
cin >> name;    // Ввод строки (до пробела)
```

```
// Множественный ввод
```

```
cin >> age >> salary >> name;
```

4. Оператор записи (вывода)

Эталонный ответ:

Оператор записи используется для вывода данных на экран. В C++ для этого используется объект `cout` (console output) из библиотеки `<iostream>`.

Синтаксис:

```
cout << expression;
```

Примеры:

```
int x = 10;
```

```
double y = 3.14;
```

```
cout << "Hello World!";    // Вывод строки
```

```
cout << x;                  // Вывод переменной
```

```
cout << "x = " << x << endl; // Комбинированный вывод
```

```
cout << "Sum: " << (x + y); // Вывод выражения
```

```
// Форматированный вывод
```

```
cout << fixed << setprecision(2) << y; // Вывод с 2 знаками после запятой
```

5. Математические стандартные функции ЯП

Эталонный ответ:

Стандартные математические функции находятся в библиотеке `<cmath>`. Основные функции:

1. Тригонометрические:

- `sin(x)`, `cos(x)`, `tan(x)` - синус, косинус, тангенс (в радианах)
- `asin(x)`, `acos(x)`, `atan(x)` - арксинус, арккосинус, арктангенс

2. Экспоненциальные и логарифмические:

- `exp(x)` - экспонента e^x
- `log(x)` - натуральный логарифм
- `log10(x)` - десятичный логарифм
- `pow(x, y)` - возведение в степень x^y
- `sqrt(x)` - квадратный корень

3. Прочие функции:

- `fabs(x)` - модуль вещественного числа
- `ceil(x)` - округление вверх
- `floor(x)` - округление вниз

Пример:

```
#include <cmath>

double x = 2.0;

double result = pow(x, 3) + sqrt(x) + sin(x);
```

Раздел 2. Операторы языка программирования

6. Ветвления в программах

Эталонный ответ:

Ветвления позволяют изменять порядок выполнения программы в зависимости от условий. Основные конструкции ветвлений в C++:

1. Условный оператор `if`:

```
if (condition) {
    // выполняется, если condition истинно
}
```

2. Оператор `if-else`:

```
if (condition) {
    // блок if
} else {
```

```
    // блок else
}
```

3. Цепочка условий else-if:

```
if (condition1) {
    // блок 1
} else if (condition2) {
    // блок 2
} else {
    // блок по умолчанию
}
```

Пример:

```
int score = 85;
if (score >= 90) {
    cout << "Отлично!";
} else if (score >= 75) {
    cout << "Хорошо";
} else {
    cout << "Удовлетворительно";
}
```

7. Использование составного оператора

Эталонный ответ:

Составной оператор (блок) - это группа операторов, заключенная в фигурные скобки {}. Блок воспринимается компилятором как единый оператор.

Назначение:

- Группировка нескольких операторов
- Определение области видимости переменных
- Использование в конструкциях if, for, while и др.

Пример:

```
{
    int x = 10; // переменная видна только внутри блока
    cout << x;
    x = x * 2;
}
// x здесь уже не видна
```

8. Оператор выбора

Эталонный ответ:

Оператор switch используется для выбора одного из многих вариантов выполнения на основе значения выражения.

Синтаксис:

```
switch (expression) {  
    case constant1:  
        // операторы  
        break;  
    case constant2:  
        // операторы  
        break;  
    default:  
        // операторы по умолчанию  
}
```

Особенности:

- expression должно быть целочисленного или перечисляемого типа
- break предотвращает "проваливание" в следующий case
- default выполняется, если ни один case не подошел

Пример:

```
int day = 3;  
switch (day) {  
    case 1: cout << "Понедельник"; break;  
    case 2: cout << "Вторник"; break;  
    case 3: cout << "Среда"; break;  
    default: cout << "Другой день";  
}
```

9. Цикл с предусловием

Эталонный ответ:

Цикл while (цикл с предусловием) выполняет блок операторов, пока условие истинно. Условие проверяется ПЕРЕД каждой итерацией.

Синтаксис:

```
while (condition) {  
    // тело цикла  
}
```

Особенности:

- Тело цикла может не выполниться ни разу
- Необходимо обеспечить изменение условия для выхода из цикла

Пример:

```
int i = 1;
while (i <= 5) {
    cout << i << " ";
    i++; // изменение условия
}
// Вывод: 1 2 3 4 5
```

10. Цикл с постусловием**Эталонный ответ:**

Цикл do-while (цикл с постусловием) выполняет блок операторов, пока условие истинно. Условие проверяется ПОСЛЕ каждой итерации.

Синтаксис:

```
do {
    // тело цикла
} while (condition);
```

Особенности:

- Тело цикла выполнится как минимум один раз
- Условие проверяется после выполнения тела

Пример:

```
int number;
do {
    cout << "Введите положительное число: ";
    cin >> number;
} while (number <= 0);
```

11. Цикл с параметром**Эталонный ответ:**

Цикл for (цикл с параметром) используется, когда известно количество итераций. Состоит из трех частей: инициализация, условие, изменение.

Синтаксис:

```
for (инициализация; условие; изменение) {
    // тело цикла
}
```

Работа цикла:

1. Выполняется инициализация
2. Проверяется условие
3. Если истинно - выполняется тело цикла
4. Выполняется изменение параметра
5. Возврат к шагу 2

Пример:

```
for (int i = 1; i <= 5; i++) {  
    cout << i << " ";  
}  
  
// Вывод: 1 2 3 4 5
```

Раздел 3. Структурное и модульное программирование**12. Подпрограммы-функции****Эталонный ответ:**

Функция - это подпрограмма, которая возвращает значение. Функции используются для структурирования кода и избежания дублирования.

Синтаксис:

```
тип_возвращаемого_значения имя_функции(параметры) {  
    // тело функции  
    return значение;  
}
```

Пример:

```
// Объявление (прототип)  
double calculateArea(double radius);  
  
// Определение  
double calculateArea(double radius) {  
    return 3.14159 * radius * radius;  
}  
  
// Вызов  
double area = calculateArea(5.0);
```

13. Подпрограммы-процедуры

Эталонный ответ:

В C++ процедуры реализуются как функции с типом возвращаемого значения `void` (не возвращают значение). Используются для выполнения действий.

Синтаксис:

```
void имя_процедуры(параметры) {  
    // тело процедуры  
    // нет оператора return или return без значения  
}
```

Пример:

```
void printHello(string name) {  
    cout << "Hello, " << name << "!" << endl;  
}
```

// Вызов

```
printHello("Alice");
```

14-17. Рекуррентные последовательности

Эталонный ответ:

Рекуррентная последовательность - это последовательность, где каждый следующий элемент вычисляется через предыдущие.

Арифметическая прогрессия:

```
//  $a_n = a_1 + (n-1) * d$   
int arithmetic(int a1, int d, int n) {  
    return a1 + (n - 1) * d;  
}
```

Геометрическая прогрессия:

```
//  $b_n = b_1 * q^{n-1}$   
int geometric(int b1, int q, int n) {  
    return b1 * pow(q, n - 1);  
}
```

Числа Фибоначчи:

```
//  $F_0=0, F_1=1, F_n=F_{n-1}+F_{n-2}$   
int fibonacci(int n) {  
    if (n <= 1) return n;
```

```

int a = 0, b = 1;
for (int i = 2; i <= n; i++) {
    int temp = a + b;
    a = b;
    b = temp;
}
return b;
}

```

Факториал:

```

//  $n! = 1 * 2 * 3 * \dots * n$ 
int factorial(int n) {
    int result = 1;
    for (int i = 2; i <= n; i++) {
        result *= i;
    }
    return result;
}

```

27-30. Файлы и файловые переменные

Эталонный ответ:

Работа с файлами в C++ осуществляется через классы ifstream (чтение), ofstream (запись), fstream (чтение/запись) из библиотеки <fstream>.

Основные операции:

1. Открытие файла:

```

ifstream input("file.txt");
ofstream output("file.txt");
fstream file("file.txt", ios::in | ios::out);

```

2. Проверка открытия:

```

if (!file.is_open()) {
    cout << "Ошибка открытия файла!";
}

```

3. Чтение/запись:

```

// Запись
output << data;

```

```
// Чтение
```

```
input >> variable;
```

4. Закрытие файла:

```
file.close();
```

31-32. Комбинированный тип данных

Эталонный ответ:

Структуры (struct) позволяют объединять данные разных типов в один составной тип.

Синтаксис:

```
struct StructName {  
    type1 field1;  
    type2 field2;  
    // ...  
};
```

Пример:

```
struct Student {  
    string name;  
    int age;  
    double gpa;  
};
```

```
Student s = {"Alice", 20, 4.5};
```

```
cout << s.name << " " << s.age;
```

39-40. Внешние подпрограммы и модули

Эталонный ответ:

Модульное программирование предполагает разделение программы на несколько файлов:

1. Заголовочный файл (.h) - объявления:

```
// math_utils.h
```

```
#ifndef MATH_UTILS_H
```

```
#define MATH_UTILS_H
```

```
int add(int a, int b);
```

```
double calculateCircleArea(double radius);
```

```
#endif
```

2. Файл реализации (.c) - определения:

```
// math_utils.  
#include "math_utils.h"  
  
int add(int a, int b) {  
    return a + b;  
}  
  
double calculateCircleArea(double radius) {  
    return 3.14159 * radius * radius;  
}
```

3. Основной файл - использование:

```
// main.  
#include "math_utils.h"  
  
int main() {  
    int sum = add(5, 3);  
    double area = calculateCircleArea(2.0);  
    return 0;  
}
```

Раздел 4. Основные конструкции языков программирования 18-21. Строковый тип данных

Эталонный ответ:

В C++ строки представлены классом string из библиотеки <string>.

Основные операции:

```
string str = "Hello";
```

```
// Длина строки
```

```
int len = str.length();
```

```
// Конкатенация
```

```
string result = str + " World!";
```

```
// Доступ к символам
```

```
char c = str[0];
```

```
// Подстрока
```

```
string sub = str.substr(1, 3); // "ell"
```

```
// Поиск
```

```
int pos = str.find("ll"); // 2
```

Функции `ord` и `chr` (аналоги в C++):

```
char c = 'A';
```

```
int code = static_cast<int>(c); // ord - код символа (65)
```

```
char symbol = static_cast<char>(65); // chr - символ по коду ('A')
```

22-24. Массивы

Эталонный ответ:

Одномерные массивы:

```
int arr[5] = {1, 2, 3, 4, 5};
```

Двумерные массивы:

```
int matrix[3][3] = {  
    {1, 2, 3},  
    {4, 5, 6},  
    {7, 8, 9}  
};
```

Сортировка пузырьком:

```
void bubbleSort(int arr[], int n) {  
    for (int i = 0; i < n-1; i++) {  
        for (int j = 0; j < n-i-1; j++) {  
            if (arr[j] > arr[j+1]) {  
                swap(arr[j], arr[j+1]);  
            }  
        }  
    }  
}
```

25-26. Множественный тип данных

Эталонный ответ:

Множества в C++ представлены классом `set` из библиотеки `<set>`.

Основные операции:

```
#include <set>

set<int> numbers = {1, 2, 3, 4, 5};

// Добавление
numbers.insert(6);

// Удаление
numbers.erase(3);

// Проверка наличия
if (numbers.find(2) != numbers.end()) {
    // элемент найден
}

// Объединение, пересечение, разность
set<int> A = {1, 2, 3}, B = {2, 3, 4};
set<int> C; // для результатов

set_union(A.begin(), A.end(), B.begin(), B.end(),
          inserter(C, C.begin())); // объединение
```

33-38. Динамические структуры данных

Эталонный ответ:

Указатели:

```
int x = 10;
int* ptr = &x; // ptr указывает на x
cout << *ptr; // разыменование (10)
```

Связные списки:

```
struct Node {
    int data;
    Node* next;
};

Node* head = nullptr; // начало списка
```

Очередь (FIFO): - First In, First Out

Стек (LIFO): - Last In, First Out

Дерево: - иерархическая структура с корневым узлом

Раздел 5. Объектно-ориентированное программирование

40. Основные понятия ООП

Эталонный ответ:

Объектно-ориентированное программирование (ООП) - парадигма, основанная на концепциях:

1. **Класс** - шаблон для создания объектов, описывающий их структуру и поведение
2. **Объект** - экземпляр класса с конкретными значениями полей
3. **Абстракция** - выделение существенных характеристик объекта
4. **Конструктор** - метод для инициализации объекта
5. **Деструктор** - метод для очистки ресурсов при уничтожении объекта

43. Понятие инкапсуляции

Эталонный ответ:

Инкапсуляция - механизм объединения данных и методов для работы с ними в единый объект, и ограничение доступа к внутренним данным.

Уровни доступа в C++:

- private - доступ только внутри класса
- protected - доступ внутри класса и производных классов
- public - доступ из любого места

Пример:

```
class BankAccount {
private:
    double balance; // скрытые данные

public:
    // публичные методы для работы с данными
    void deposit(double amount) {
        if (amount > 0) balance += amount;
    }

    double getBalance() { return balance; }
};
```

42. Понятие наследования

Эталонный ответ:

Наследование - механизм создания нового класса на основе существующего, с наследованием его свойств и методов.

Виды наследования:

- public - публичные и protected члены остаются такими же
- protected - публичные становятся protected
- private - все члены становятся private

Пример:

```
class Animal {      // базовый класс
protected:
    string name;
public:
    void eat() { cout << "Eating..."; }
};

class Dog : public Animal { // производный класс
public:
    void bark() { cout << "Woof!"; }
};
```

41. Понятие полиморфизма**Эталонный ответ:**

Полиморфизм - возможность объектов с одинаковой спецификацией иметь различную реализацию.

Виды полиморфизма:**1. Статический (перегрузка методов):**

```
class Calculator {
public:
    int add(int a, int b) { return a + b; }
    double add(double a, double b) { return a + b; }
};
```

2. Динамический (виртуальные функции):

```
class Shape {
public:
    virtual double area() = 0; // чисто виртуальная функция
};

class Circle : public Shape {
    double radius;
```

```
public:
    double area() override { return 3.14 * radius * radius; }
};
```

```
class Rectangle : public Shape {
    double width, height;
public:
    double area() override { return width * height; }
};
```

Эти ответы охватывают все ключевые теоретические аспекты курса "Основы алгоритмизации и программирования" и могут служить эталоном для подготовки к экзамену.

ПЕРЕЧЕНЬ МАТЕРИАЛОВ, ОБОРУДОВАНИЯ И ИНФОРМАЦИОННЫХ ИСТОЧНИКОВ

ИНСТРУМЕНТ ПРОВЕРКИ

Критерии оценки за ответ на теоретические вопросы	
Оценка	Критерии оценки ответа студента
«Отлично»	Обстоятельно и с достаточной полнотой излагает материал вопросов. Даёт ответ на вопрос в определенной логической последовательности. Даёт правильные формулировки, точные определения понятий и терминов. Демонстрирует полное понимание материала, даёт полный и аргументированный ответ на вопрос, приводит необходимые примеры (не только рассмотренные на занятиях, но и подобранные самостоятельно). Свободно владеет речью (показывает связанность и последовательность в изложении).
«Хорошо»	Даёт ответ, удовлетворяющий тем же требованиям, что и для оценки «отлично», но допускает единичные ошибки, неточности, которые сам же исправляет после замечаний преподавателя.
«Удовлетворительно»	Обнаруживает знание и понимание основных положений, но: – допускает неточности в формулировке определений, терминов; – излагает материал недостаточно связно и последовательно; – на вопросы экзаменаторов отвечает некорректно.
«Неудовлетворительно»	Обнаруживает непонимание основного содержания учебного материала. Допускает в формулировке определений ошибки, искажающие их смысл. Допускает существенные ошибки, которые не может исправить при наводящих вопросах преподавателя или ответ отсутствует. Беспорядочно и неуверенно излагает материал. Сопровождает изложение частыми заминками и перерывами.

Критерии оценки за выполнение практического задания

Оценка	Критерии
«Отлично»	Показал полное знание технологии выполнения задания. Продemonстрировал умение применять теоретические знания/правила выполнения/технологию при выполнении задания. Уверенно выполнил действия согласно условию задания.
«Хорошо»	Задание в целом выполнил, но допустил неточности. Показал знание технологии/алгоритма выполнения задания, но недостаточно уверенно применил их на практике. Выполнил норматив на положительную оценку.
«Удовлетворительно»	Показал знание общих положений, задание выполнил с ошибками. Задание выполнил на положительную оценку, но превысил время, отведенное на выполнение задания.
«Неудовлетворительно»	Не выполнил задание. Не продемонcтрировал умения самостоятельного выполнения задания. Не знает технологию/алгоритм выполнения задания. Не выполнил норматив на положительную оценку.

4. МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ, ОПРЕДЕЛЯЮЩИЕ ПРОЦЕДУРЫ ОЦЕНИВАНИЯ ЗНАНИЙ

Процедура оценивания – порядок действий при подготовке и проведении аттестационных испытаний и формировании оценки.

Процедура промежуточной аттестации проходит в соответствии с Положением о промежуточной (рубежной) аттестации знаний студентов и учащихся НЭК.

Аттестационные испытания проводятся преподавателем (или комиссией преподавателей – в случае модульной дисциплины), ведущим лекционные занятия по данной дисциплине, или преподавателями, ведущими практические и лабораторные занятия (кроме устного экзамена). Присутствие посторонних лиц в ходе проведения аттестационных испытаний без разрешения ректора или проректора не допускается (за исключением работников университета, выполняющих контролирующие функции в соответствии со своими должностными обязанностями). В случае отсутствия ведущего преподавателя аттестационные испытания проводятся преподавателем, назначенным письменным распоряжением по кафедре.

Инвалиды и лица с ограниченными возможностями здоровья, имеющие нарушения опорно-двигательного аппарата, допускаются на аттестационные испытания в сопровождении ассистентов-сопровождающих.

Во время аттестационных испытаний обучающиеся могут пользоваться программой учебной дисциплины, а также с разрешения преподавателя справочной и нормативной литературой, непрограммируемыми калькуляторами.

Время подготовки ответа при сдаче зачета/экзамена в устной форме должно составлять не менее 40 минут (по желанию обучающегося ответ может быть досрочным). Время ответа – не более 15 минут.

При подготовке к устному экзамену экзаменуемый, как правило, ведет записи в листе устного ответа, который затем (по окончании экзамена) сдается экзаменатору.

При проведении устного экзамена экзаменационный билет выбирает сам экзаменуемый в случайном порядке.

Экзаменатору предоставляется право задавать обучающимся дополнительные вопросы в рамках программы дисциплины текущего семестра, а также, помимо теоретических вопросов, давать задачи, которые изучались на практических занятиях.

Оценка результатов устного аттестационного испытания объявляется обучающимся в день его проведения. При проведении письменных аттестационных испытаний или компьютерного тестирования – в день их проведения или не позднее следующего рабочего дня после их проведения.

Результаты выполнения аттестационных испытаний, проводимых в письменной форме, форме итоговой контрольной работы или компьютерного тестирования, должны быть объявлены обучающимся и выставлены в зачётные книжки не позднее следующего рабочего дня после их проведения.

Порядок подготовки и проведения промежуточной аттестации в форме зачета/экзамена

действие	сроки	методика	ответственный
выдача вопросов для промежуточной аттестации	1 неделя семестра	на лекционных /практических и др.занятиях, на офиц. сайте суза и др.	ведущий преподаватель
консультации	последняя неделя семестра/период сессии	на групповой консультации	ведущий преподаватель
промежуточная аттестация	в период сессии	устно, письменно, тестирование бланочное или компьютерное, по билетам, с практическими заданиями	ведущий преподаватель, комиссия
формирование оценки	на аттестации		ведущий преподаватель, комиссия