

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ КЫРГЫЗСКОЙ
РЕСПУБЛИКИ
СОВРЕМЕННЫЙ МЕЖДУНАРОДНЫЙ УНИВЕРСИТЕТ**

КАФЕДРА ИНФОРМАТИКИ

«Согласовано»

Начальник учебной части

Ерке Е.Э.

« 25 » 08 2024 г.



«Утверждаю»

Проректор СМУ

Максат Макамбаев

« ЧАСТ » 20__ г.



РАБОЧАЯ ПРОГРАММА

Алгоритмизация и программирование

для специальности (направления специальности): **710300 «Прикладная информатика»**

Форма обучения: очная/заочная

Виды учебной деятельности:

Программа	Всего часов				СРС	СРС П	Форма отчетности
	всего	аудиторных	лекц ия	Прак.р аб			
2-курс, 4 семестр (о/о)	4 кр 120 час	60	-	60	40 ч	20 ч	экзамен
2-курс, 4 семестр (з/о)	4 кр 120 час	20	-	20	70 ч	30 ч	экзамен

«Согласовано»

Профилирующей кафедрой

зав.каф.

от « 25 » 20__ г.

«Рассмотрено»

на заседании кафедры

от « 25 » 20__ г.

Рабочая программа дисциплины разработана в соответствии с ГОС (1578/1 от 21 сентября 2021 г), ОПОП направления 710300 - Прикладная информатика, Положения о разработке УМК в СМУ.

Разработчик: Э.К. Эркулова, Абилов К.Б.

Жалал-Абад

2024 г.

ЛИСТ ИЗМЕНЕНИЙ

Основание: Протокол № 10 заседания кафедры от «22» мая 2024 г.

№	Раздел рабочей программы	Содержание изменения
1	II. Общая характеристика дисциплины	Объём дисциплины увеличить с 3 кредитов до 4 кредитов
2	V. Расчёт трудоёмкости дисциплины	Скорректировать общую нагрузку: вместо 90 аудиторных часов — 120 часов
3	VI. Содержание занятий	Все ранее запланированные аудиторные лекционные часы перевести в практические
4	VII. Формы контроля и оценивания	Пересмотреть график промежуточных тестирований с учётом увеличения объёма
5	VIII. Рекомендуемая литература и ресурсы	Добавить литературу в твердом переплете из библиотеки СМУ

Зав каф информатики



Белеков

Содержание

.Рабочая программа	1
1.1 реквизиты дисциплины	4
1.2 объем дисциплины и виды учебной работы	4
2. Место дисциплины в структуре ооп	4
3. Пререквизиты и постреквизиты курса	4
4. Необходимость изучения курса	5
5. Цели и задачи курса	5
6. Компетенции обучающегося, формируемые в результате освоения дисциплины формирование компетенций для направления: 710300 «прикладная информатика»	6
7. Образовательные технологии	7
8 критерий оценки знаний студентов на экзамене	7
8.1. Оценка знаний (академической успеваемости) осуществляется по 100 балльной системе (шкале) следующим образом:	9
8.2. Технологическая карта дисциплины	9
9. Тематический план и содержание учебной дисциплины «алгоритмизация и программирование» для студентов очного обучения	10
тематический план и содержание учебной дисциплины «алгоритмизация и программирование» для студентов заочного обучения	19
10. Учебно-методическое, информационное и материально-техническое обеспечение дисциплины	25
11. Вопросы для подготовки к экзамену	27
12. Методические указания для преподавателей дисциплины «алгоритмизация и программирование»	31

1.1 Реквизиты дисциплины

Дисциплина : «Алгоритмизация и программирование »

Курс: 2

Направление: 710300 – Прикладная информатика

Количество кредит часов: 4 кр (120 ч)

Форма обучения: очная

1.2 Объем дисциплины и виды учебной работы

Вид учебной работы	Семестр 3 о/о	Семестр 3 з/о
	Всего часов 120 ч	Всего часов 120 ч
Аудиторные занятия (всего)	60	20
В том числе:		
Лекции (Л)	-	
Практические занятия (ПЗ)	60	20
Семинары (С)		
Лабораторные практикумы (ЛП)		
Клинические практические занятия (КПЗ)		
Самостоятельная работа (всего)	60	100
СРСП	20	30
СРС	40	70
Форма контроля	экзамен	экзамен
Общая трудоемкость (час.)	120 ч	120 ч

2. Место дисциплины в структуре ООП

Учебная дисциплина «Алгоритмизация и программирование» является специальной дисциплиной, формирующей профессиональные знания, необходимые для будущей трудовой деятельности.

Дисциплина относится к базовой части профессионального цикла.

3. Пререквизиты и постреквизиты курса

Пререквизиты:

- математика,
- физика,
- информатика

Постреквизиты:

- разработка и эксплуатация АИС
- программное обеспечение АИС
- Автоматизированные информационные системы

4. Необходимость изучения курса

Курс «Алгоритмизация и программирование» посвящен изучению базовых принципов программирования, получению знаний по методологии языков программирования, а также обзору современных тенденций в программировании. В процессе обучения формируются начальные навыки кодирования и реализации программ путем оптимизации их кода.

5. Цели и задачи курса

Целью освоения дисциплины «Алгоритмизация и программирование» является изучение и освоение базовых понятий и приемов программирования, применяемых на всех основных этапах разработки программ; изучение методов программирования для овладения знаниями в области технологии программирования; подготовка к осознанному использованию как языков программирования, так и методов программирования.

Задачи:

- обучить студентов синтаксису и семантики универсального алгоритмического языка программирования Python;
- ознакомить студентов с технологиями структурного программирования;
- ознакомление с типовыми способами организации данных и построения алгоритмов обработки данных;
- сформировать у студентов навыки и умения использовать инструментальные программные средства для решения прикладных задач, составляющих содержание дисциплины специализации.

6. Компетенции обучающегося, формируемые в результате освоения дисциплины

Формирование компетенций для направления: 710300 «Прикладная информатика»

Б.3.13	Алгоритмизация и программирование	ПК-3, ПК-10, ПК-11	Способен ставить и решать прикладные задачи с использованием основных законов естественно-научных дисциплин и современных ИКТ (ПК-3), Способен применять в профессиональной деятельности современные языки программирования и языки баз данных, операционные системы, электронные библиотеки и пакеты программ, сетевые технологии, Способен принимать участие в процессе создание и управление ИС и сервисы на всех этапах жизненного цикла (ПК-11)
--------	-----------------------------------	--------------------------	---

В результате освоения учебной дисциплины обучающийся должен уметь:

- использовать языки программирования, строить логически правильные и эффективные программы.

В результате освоения учебной дисциплины обучающийся должен знать:

- общие принципы построения алгоритмов, основные алгоритмические конструкции;
- понятие системы программирования;
- основные элементы процедурного языка программирования, структуру программы, операции, управляющие структуры, структуры данных, файлы, кассы памяти;
- подпрограммы, составление библиотек программ;
- объектно-ориентированную модель программирования, понятие классов и объектов, их свойств и методов

7. Образовательные технологии

Изучение дисциплины предполагает использование традиционных способов коллективного обучения – лекций, лабораторных занятий, индивидуальных заданий с последующей отчетностью.

Применяемые информационные технологии: лекции в форме презентаций, обучающие и тестирующие программы, электронные учебники.

Оценочные средства для текущего контроля успеваемости, рубежного контроля по итогам освоения дисциплины и учебно-методическое обеспечение самостоятельной работы бакалавров:

- **формой текущего контроля знаний студентов (аудиторные занятия)** является контроль посещения лекционных и практических занятий, активность студентов на практических занятиях. Каждый из двух текущих контролей оценивается по **30 баллов**.

формой текущего контроля знаний студентов (внеаудиторные занятия) является контроль СРСР, участие в НИРС (выступление на студенческой конференции, публикация статей)

- **формой итогового контроля знаний и умений бакалавров по курсу** является экзамен.

Текущий и рубежный контроль. Студенты после выполнения соответствующих (первому или второму модулю) практических и лабораторных работ допускаются к рубежному контролю. Каждый из двух рубежных контролей (модулей) оценивается по **30** балльной шкале.

Итоговый контроль. Итоговый контроль реализуется в форме защиты собственно созданных программ (в виде компьютерного тестирования) и оценивается по **30** балльной шкале.

Правила оценивания рубежного и итогового контроля.

8 Критерий оценки знаний студентов на экзамене

Выставление оценок на экзаменах осуществляется на основе принципов объективности, справедливости, всестороннего анализа качества знаний студентов, и других положений, способствующих повышению надежности оценки знаний обучающихся, и устранению субъективных факторов.

В соответствии с действующими нормативными актами и рекомендациями Министерства образования и науки КР

устанавливаются следующие критерии выставления оценок на экзаменах:

- **оценка "отлично"** выставляется студенту, который обнаружил на экзамене всестороннее, систематическое и глубокое знание учебно-программного материала, умение свободно выполнять задания, предусмотренные программой, который усвоил основную литературу и ознакомился с дополнительной литературой, рекомендованной программой. Как правило, оценка "отлично" выставляется студентам, усвоившим взаимосвязь основных понятий дисциплины и их значений для приобретаемой профессии, проявившим творческие способности в понимании, изложении и использовании учебно-программного материала;

- **оценка "хорошо"** выставляется студенту, который на экзамене обнаружил полное знание учебно-программного материала, успешно выполнил предусмотренные в программе задания, усвоил основную литературу, рекомендованную в программе. Как правило, оценка "хорошо" выставляется студентам, показавшим систематический характер знаний по дисциплине и способным к их самостоятельному выполнению и обновлению в ходе дальнейшей учебной работы и профессиональной деятельности;

- **оценка "удовлетворительно"** выставляется студенту, обнаружившему знание основного учебного материала в объеме, необходимом для дальнейшей учебы и предстоящей работы по профессии, справляющемуся с выполнением заданий, предусмотренных программой, который ознакомился с основной литературой, рекомендованной программой. Как правило, оценка "удовлетворительно" выставляется студентам, допустившим погрешности в ответе на экзамене и при выполнении экзаменационных заданий, но обладающим необходимыми знаниями для их устранения под руководством преподавателя;

- **оценка "неудовлетворительно"** выставляется студенту, обнаружившему пробелы в знаниях основного учебно-программного материала, допустившему принципиальные ошибки в выполнении предусмотренных программой заданий, не ознакомившемуся с основной литературой, предусмотренной программой, и не овладевшему базовыми знаниями, предусмотренными по данной дисциплине и определенными соответствующей программой курса (перечень основных знаний и умений, которыми должны овладеть студенты, является обязательным элементом рабочей программы курса).

8.1. Оценка знаний (академической успеваемости) осуществляется по 100 балльной системе (шкале) следующим образом:

30 балльная система	100 балльная система	Оценка по буквенной системе	Цифровой эквивалент оценки по GPA	Оценка по традиционной системе
26 - 30	87 – 100	A	4,0	Отлично
24 - 25	80 – 86	B	3,33	Хорошо
22 - 23	74 – 79	C	3,0	
20 - 21	68 - 73	Д	2,33	Удовлетворительно
18 - 19	60 – 67	E	2,0	
9 - 17	31 - 60	FX	0	Неудовлетворительно
0 - 8	0 - 30	F	0	

8.2. Технологическая карта дисциплины

Всего		Ауд. часы	СРС, СРСП	1-модуль (90 ч., 30 б.)			2-модуль (90 ч., 30 б.)			Итоговый контроль (ЖТ) (30 б.)				П			
90	45			Ауд. ч.		СРС, СРСП	Рубежный контроль	Ауд. ч.		СРС, СРСП	Рубежный контроль	Лекция	Лаборатория	СРС	Итоговый контроль		
				Лекция	Практические занятия												
90	45	45	15	30	45		15	30	45								
Баллы			30	30	30	30 б.	30	30	30	30 б.	30	30	30	30 б.	10 б.		

Итоги модулей	$TK = (Лек + Лаб + СРС) / 3$, $M1 = (TK1 + TK2 + PK1) / 3$	$TK = (Лек + Лаб + СРС) / 3$, $M2 = (TK3 + TK4 + PK2) / 3$	$ИК = (Лек + Лаб + СРС) / 3$, $Экз = M1 + M2 + ИК + П$	100
------------------	--	--	--	------------

9. Тематический план и содержание учебной дисциплины «Алгоритмизация и программирование» для студентов очного обучения

МОДУЛЬ 1. Основы алгоритмизации и программирования (Python)

№	Тема	Задание на выполнение	Ожидаемые результаты	час
1	Практическое занятие №1 Введение в алгоритмизацию и программирование	Изучение базовых понятий алгоритмизации, выполнение простых операций в Python	Знание основ алгоритмизации и программирования	2
2	Лабораторная работа №1 Анализ требований и постановка задачи – формулировка задач и проектирование решений	Разбор и формулировка задач перед программированием, проектирование решений	Умение анализировать требования и проектировать алгоритмы	2
3	Практическое занятие №2 Программная реализация разветвляющегося алгоритма	Реализация простых алгоритмов с использованием условных операторов	Умение применять условные конструкции	2
4	Лабораторная работа №2 Работа с условными операторами if-elif-else, вложенные условия	Разработка программы с вложенными условиями	Умение применять сложные условные конструкции	2
5	Практическое занятие №3 Программная реализация циклических алгоритмов	Создание программ с использованием циклов	Знание и применение циклов в программировании	2

6	Лабораторная работа №3 Использование циклов for и while, задачи на вложенные циклы	Решение задач с вложенными циклами	Умение применять циклы для решения задач	2
7	Практическое занятие №4 Процедуры и функции в Python	Разработка программ с использованием функций	Умение работать с функциями, передача аргументов	2
8	Лабораторная работа №4 Создание и использование функций, передача аргументов, возврат значений	Разработка программ с функциями, возврат значений	Понимание принципов использования функций	2
9	Практическое занятие №5 Графическая система Turtle	Построение базовых фигур с использованием библиотеки Turtle	Умение программировать графику	2
10	Лабораторная работа №5 Построение простых фигур (circle, square, triangle) с использованием turtle	Написание программы для рисования фигур	Владение методами построения графических элементов	2
11	Практическое занятие №6 Графическая система Turtle: сложные рисунки	Создание сложных узоров с циклами	Умение программировать графические узоры	2
12	Лабораторная работа №6 Создание анимаций и узоров с циклами в turtle	Разработка анимационных элементов	Применение алгоритмов для создания анимации	2
13	Практическое занятие №7 Обработка событий в Tkinter	Создание программы с обработкой пользовательских событий	Владение навыками взаимодействия с пользователем	2
14	Лабораторная работа №7 Реализация обработки событий, кнопок и текстовых полей в Tkinter	Создание интерактивного GUI-приложения	Умение разрабатывать интерфейсы и обрабатывать события	4
				30

МОДУЛЬ 2. Разработка графических интерфейсов в Python

№	Название практической работы	Задание на выполнение	Ожидаемый результат	час
1	Практическое занятие №8 Введение в Tkinter: базовые элементы интерфейса	Изучение основных компонентов Tkinter (окно, кнопки, метки)	Понимание основ разработки GUI в Tkinter	2
2	Лабораторная работа №8 Создание окна приложения с кнопками и метками (Label, Button, Entry)	Разработка простого интерфейса с кнопками и вводом текста	Умение создавать базовый GUI	2
3	Практическое занятие №9 Обработка событий в Tkinter	Работа с событиями (клики, ввод текста)	Умение обрабатывать пользовательский ввод	2
4	Лабораторная работа №9 Реализация реакций на нажатие кнопок, ввод текста, работу с bind()	Разработка интерактивного интерфейса	Владение механизмами событийного программирования	2
5	Практическое занятие №10 Разработка форм в Tkinter	Создание нескольких окон и форм	Понимание работы с несколькими окнами в Tkinter	2
6	Лабораторная работа №10 Создание многооконного приложения, работа с Frame и Toplevel	Реализация многокомпонентного интерфейса	Владение компоновкой элементов GUI	2
7	Практическое занятие №11 Основы работы с PyQt: компоненты интерфейса	Знакомство с основными виджетами PyQt	Умение работать с базовыми элементами PyQt	2
8	Лабораторная работа №11 Создание базового GUI-приложения с QPushButton, QLabel, QLineEdit	Разработка простого окна с элементами интерфейса	Владение основными инструментами PyQt	2
9	Практическое занятие №12 Работа с базами данных в PyQt	Подключение базы данных к PyQt	Умение связывать GUI с базой данных	2

10	Лабораторная работа №12 Подключение SQLite, отображение данных в QTableView	Разработка приложения с базой данных	Владение методами работы с SQLite в PyQt	2
11	Практическое занятие №13 Создание меню и диалоговых окон в PyQt	Использование меню и диалоговых окон	Умение добавлять в интерфейс меню и сообщения	2
12	Лабораторная работа №13 Реализация QMenuBar, QMessageBox, QFileDialog	Реализация меню, уведомлений и диалогов	Владение инструментами пользовательского взаимодействия	2
13	Практическое занятие №14 Разработка полноценного GUI-приложения на PyQt	Создание сложного интерфейса с динамическими элементами	Владение разработкой комплексных приложений	2
14	Лабораторная работа №14 Создание приложения с сохранением данных, динамическим обновлением интерфейса	Разработка приложения с хранением данных и обновлением UI	Умение сохранять данные и взаимодействовать с пользователем	4
				30

CPC

№	Тема CPC	Описание задания	Приобретаемые компетенции	час
1	Решение линейного уравнения	Разработать псевдокод и программу для решения $ax + b = 0$ с учётом случаев $a=0$, собрать тесты.	Знания: структура ветвления, обработка особых случаев Умение: писать условные конструкции Навык: тестирование и отладка простых алгоритмов	2
2	Циклы и факториал	Реализовать вычисление $n!$ через цикл for и while, сравнить производительность.	Знания: циклы, итерация Умение: выбирать подходящий цикл Навык: измерять время выполнения и оптимизировать код	2

№	Тема СРС	Описание задания	Приобретаемые компетенции	час
3	Поиск максимального и минимального в массиве	Написать функции для поиска max/min в массиве из N чисел, оформить API и протестировать на граничных данных.	Знания: одномерные массивы, индексация Умение: обход массива, возврат результата Навык: работа с граничными условиями	2
4	Обратная строка	Разработать и реализовать алгоритм, который получает строку и возвращает её обратную последовательность символов.	Знания: строковые типы, операции со строками Умение: работать с индексами строк Навык: обработка юникода и проверки ввода	2
5	Рекурсивный подсчет чисел Фибоначчи	Составить рекурсивную функцию fib(n), дополнительно реализовать итеративный вариант и сравнить.	Знания: рекурсия, стек вызовов Умение: писать рекурсивные и итеративные алгоритмы Навык: анализ стековых переполнений	2
6	Сортировки: пузырьковая, выбором и вставками	Реализовать три алгоритма сортировки, исследовать сложность ($O(n^2)$), собрать статистику сравнения.	Знания: принципы сортировки, алгоритмическая сложность Умение: реализовать разные методы Навык: профилировать и сравнивать алгоритмы	2
7	Линейный и бинарный поиск	Написать функции линейного поиска и бинарного поиска в упорядоченном массиве, протестировать на больших данных.	Знания: алгоритмы поиска, требование предсортировки Умение: реализовать оба подхода Навык: логировать количество сравнений	2
8	Односвязный список	Спроектировать односвязный список: операции вставки, удаления, поиска и отладить.	Знания: структуры данных, указатели/ссылки Умение: управлять динамической памятью Навык: отлаживать список с помощью print/debug	2
9	Стек и очередь на массивах	Реализовать стек и очередь через массивы, обеспечить проверки переполнения/опустошения.	Знания: принципы LIFO и FIFO Умение: моделировать структуры на примитивах Навык: обработка исключительных ситуаций	2
10	Деревья: обход в глубину и ширину	Построить простое бинарное дерево, реализовать DFS и BFS, вывести порядок посещения.	Знания: графы и деревья, алгоритмы обхода Умение: использовать рекурсию и очередь Навык: визуализировать структуру и результаты обхода	2

№	Тема СРС	Описание задания	Приобретаемые компетенции	час
11	Представление графа и обход	Смоделировать неориентированный граф матрицей смежности, реализовать DFS и BFS, найти кратчайший путь (по числу рёбер).	Знания: представления графа, теория графов Умение: работать с матрицами Навык: реализовать обход и простой алгоритм поиска пути	2
12	Задача о рюкзаке (жадный алгоритм)	Решить рюкзак с дробным весом жадным методом, провести сравнение жадного решения с оптимальным для примеров.	Знания: жадные алгоритмы, жадная стратегия Умение: сортировать по отношению стоимость/вес Навык: анализ качества жадного решения	2
13	Динамическое программирование: невзвешенный рюкзак	Реализовать DP-решение задачи о рюкзаке 0/1, вывести матрицу промежуточных значений.	Знания: принципы DP, табличные методы Умение: строить таблицу значений Навык: восстанавливать решение из таблицы	2
14	Алгоритм Дейкстры	Реализовать алгоритм поиска кратчайших путей в взвешенном графе от одной вершины.	Знания: жадные алгоритмы, приоритетная очередь Умение: использовать структуру "куча"/priority queue Навык: оптимизировать время поиска	2
15	Разработка калькулятора выражений	Спроектировать парсер и вычислитель арифметических выражений с учётом приоритетов операций.	Знания: теория грамматик, стек-автоматы Умение: реализовать алгоритм сортировочной станции (shunting-yard) Навык: обрабатывать ошибки синтаксиса	2
16	Обработка файлов: чтение и запись данных	Написать программу, читающую набор чисел из файла, выполняющую сортировку и сохраняющую результат.	Знания: файловый I/O, потоки Умение: работать с файловыми путями Навык: обрабатывать ошибки открытия/закрытия файлов	2
17	Рефакторинг и документирование кода	Выбрать из репозитория свой старый проект, провести рефакторинг функций, написать документацию и комментарии.	Знания: стандарты кодирования, документации Умение: структурировать код, писать комментарии Навык: использовать инструменты документации	2
18	Оценка сложности алгоритма	Проанализировать код из предыдущих заданий, определить временную и пространственную сложность, оформить отчёт.	Знания: O-нотация, $O(n)$, $O(n^2)$, $O(\log n)$ Умение: анализировать алгоритмы Навык: оформлять вывод и делать рекомендации по оптимизации	2

№	Тема СРС	Описание задания	Приобретаемые компетенции	час
19	Работа с системами контроля версий (Git)	Создать репозиторий, вести коммиты и ветвление, выполнять слияние веток и разрешать конфликты.	Знания: основы распределённых VCS, Git workflow Умение: выполнять ключевые команды (clone, commit, merge) Навык: разрешать конфликтные слияния	2
20	Итоговый мини-проект: библиотека алгоритмов	Сконструировать библиотеку функций (поиск, сортировка, структуры данных), снабдить примерами и тестами, собрать документацию и запустить CI-проверки.	Знания: организация кода, API-дизайн Умение: модульное программирование Навык: писать юнит-тесты, настраивать CI/CD pipelines	2
				40

СРС II

Уровень 1 (начальный) 1. Написать программу «Hello, World!» 2. Сложение двух чисел, введённых пользователем 3. Подсчитать сумму элементов одномерного массива 4. Определить чётность введённого числа 5. Найти максимум и минимум в массиве из N элементов 6. Вычислить факториал n! циклом for 7. Получить обратную строку 8. Определить, является ли строка палиндромом 9. Подсчитать количество гласных в строке 10. Проверить, делится ли число на 3 и 5 одновременно 11. Вычислить среднее арифметическое массива 12. Отсортировать массив пузырьком 13. Реализовать линейный поиск в массиве 14. Сгенерировать N случайных чисел и вывести их квадрат	Уровень 2 (средний) 1. Реализовать двустороннюю очередь (deque) на массиве 2. Сортировка вставками и подсчёт перестановок 3. Бинарный поиск в отсортированном массиве 4. Вычислить n-е число Фибоначчи рекурсивно 5. Проверить корректность скобочной последовательности 6. Обход бинарного дерева in-order рекурсией 7. Подсчитать частоту встречаемости слов в тексте 8. Парсинг CSV-файла и построение таблицы данных 9. Сериализация/десериализация простого объекта в JSON 10. Подсчитать количество уникальных элементов массива 11. Реализовать очередь на связном списке 12. Жадный алгоритм для задачи о рюкзаке (дробный) 13. Рекурсивный обход графа (DFS) 14. Начиная с A, B, C создать комбинации без повторов 15. Преобразовать инфикс-выражение в постфикс (shunting-yard)
--	---

15. Подсчитать количество слов в предложении 16. Сложить два двумерных массива одинакового размера 17. Обменять местами минимальный и максимальный элементы массива 18. Написать простейший калькулятор (+, -, *, /) 19. Найти сумму диагоналей квадратной матрицы 20. Проверить, является ли год високосным 21. Преобразовать температуру из Цельсия в Фаренгейты 22. Подсчитать количество отрицательных и положительных чисел в массиве 23. Вычислить n-е число Фибоначчи итеративно 24. Создать программу-таймер обратного отсчёта 25. Определить длину введённой строки 26. Найти факториал с помощью while-цикла 27. Сложить элементы чётных позиций массива 28. Сравнить два введённых числа и вывести большее 29. Проверить, входит ли символ 'а' в строку 30. Вывести таблицу умножения для числа k	16. Вычислить значение выражения в обратной польской нотации 17. Применить алгоритм Дейкстры для взвешенного графа 18. Сортировка слиянием и подсчёт сравнений 19. Поиск кратчайшего пути в невзвешенном графе (BFS) 20. Использовать стек для проверки палиндрома 21. Решить задачу «количество путей в матрице» динамическим программированием 22. Организация простого логгера в файл 23. Подключение к REST-API и вывод JSON-ответа 24. Создать функцию-декоратор (или аналог в выбранном языке) 25. Построить простое GUI-окно с кнопкой (любая библиотека) 26. Написать сервер-клиент через TCP-сокеты 27. Обход графа и поиск цикла 28. Разложить число на простые множители 29. Настроить многопоточную задачу producer-consumer 30. Создать модуль для работы с датами и временем
Уровень 3 (продвинутый) 1. Реализовать хеш-таблицу с открытой адресацией 2. Построить и сбалансировать AVL-дерево 3. Алгоритм Кнута–Морриса–Пратта для поиска подстроки 4. Динамическое программирование: задача о рюкзаке 0/1 5. Поиск минимального остовного дерева (Крускал или Прим)	Темы проектов по уровням сложности Начальный 1. Консольный калькулятор с поддержкой +, -, *, / и скобок 2. Менеджер задач (To-Do) с сохранением в текстовый файл 3. Генератор случайных паролей с опциями сложности 4. Простой чат-бот «Эхо» в консоли 5. Анализатор текста: подсчёт слов и предложений в файле

6. Сбор и сортировка больших данных с внешней памятью (external sort) 7. Реализация алгоритма Полларда для факторизации 8. Алгоритм Форда–Фалкерсона для максимального потока 9. Структура данных «фенwickово дерево» (Fenwick tree) для сумм на префиксах 10. Декартово дерево (treap) и операции слияния/разбиения 11. Алгоритм Юникстного рандома (Xorshift) и тестирование RNG 12. Параллельная сортировка слиянием (multithreaded merge sort) 13. MapReduce-эмуляция: распределённая агрегация (локальная симуляция) 14. Реализация простого JVM-подобного интерпретатора байткода 15. Фреймворк плагинов: загрузка динамических модулей 16. Алгоритмическая генерация лабиринтов и поиск пути 17. Кластеризация k-means с визуализацией центроидов 18. Система рекомендаций на основе коллаборативной фильтрации 19. Сжатие данных: алгоритм Хаффмана 20. Реализация криптосистемы RSA с генерацией ключей 21. Алгоритм синтаксического анализа LL(1) для простого языка 22. Самообучающаяся нейросеть (backpropagation) для XOR-задачи 23. Подключение к базе данных SQL, ORM-прослойка 24. Реализация трассировки вызовов (profiling) через дескрипторы 25. Система очередей с гарантированной доставкой (RabbitMQ-эмулятор) 26. Разработка простого HTTP-фреймворка (роутинг, middleware) 27. Реализация event-sourcing и CQRS для CRUD-операций 28. Побудова DAG и топологическая сортировка 29. Парадигма реактивного программирования: базовая реализация Observable	6. Программа-таймер обратного отсчёта в консоли 7. Конвертер температур (Цельсий ↔ Фаренгейт) 8. Игра «Угадай число» с ограниченным числом попыток Средний 1. Веб-приложение «Библиотека»: каталог книг, поиск и выдача (Python+Flask или Node.js) 2. Клиент-серверный чат через TCP-сокеты 3. Система голосования (анонимные опросы) с хранением в SQLite 4. Анализатор логов веб-сервера: графики запросов по часам 5. Интерактивный визуализатор алгоритмов сортировки на JavaScript 6. Приложение для управления студентами: CRUD-операции и фильтры 7. Телеграм-бот-помощник по расписанию занятий 8. Скрипт для парсинга и агрегации данных из публичного API (например, погода) Продвинутый 1. Сервис рекомендаций (content-based или collaborative) на основе реальных данных 2. Реализация алгоритма Дейкстры с визуализацией в веб-интерфейсе 3. Кластеризация k-means и отображение кластеров на графике 4. Генетическая оптимизация параметров задачи (например, задачи рюкзака) 5. Разработка микросервиса с REST-API и аутентификацией JWT 6. Реализация собственного простейшего интерпретатора (REPL) 7. Параллельная обработка данных с использованием потоков и очередей 8. Система управления задачами с очередями на RabbitMQ-эмуляторе
--	--

30. Оптимизация алгоритма через SIMD-инструкции или GPU (если поддерживается)	
--	--

**Тематический план и содержание учебной дисциплины
«Алгоритмизация и программирование» для студентов заочного
обучения**

№	Тема	Задание на выполнение	Ожидаемые результаты	Час
1	Введение в алгоритмизацию и Python	Изучить базовые понятия алгоритма, написать «Hello, World!» и простые арифметические операции	Знание основ алгоритмизации и навыки работы в интерпретаторе Python	2
2	Разветвляющиеся (условные) алгоритмы	Реализовать программу расчёта скидки: если сумма > 1000, скидка 10 %, иначе 0 %	Умение применять операторы if / elif / else	2
3	Циклические алгоритмы	Написать программу подсчёта суммы чётных чисел от 1 до N с использованием for и while	Владение циклами и функцией range()	2
4	Одномерные структуры данных	Создать программу, которая собирает список оценок студента, вычисляет среднее и выводит список «неуд.»	Умение работать со списками (list), методами append(), sum()	2
5	Функции и модули	Написать функцию is_prime(n) для проверки простого числа и использовать её в основной программе	Понимание определения, вызова функций и организации кода в модули	2
6	Работа с файлами	Разработать программу, читающую текстовый файл, подсчитывающую количество строк и слов	Умение открывать, читать, записывать файлы и обрабатывать их содержимое	2
7	Обработка исключений	Написать программу деления: запрашивать	Владение механизмами try /	2

№	Тема	Задание на выполнение	Ожидаемые результаты	Час
		два числа, обрабатывать деление на ноль через try/except	except / finally для обработки ошибок	
8	Рекурсивные алгоритмы	Реализовать рекурсивную функцию подсчёта факториала и тестировать её на нескольких значениях	Понимание работы рекурсии и базового условия остановки	2
9	Словари и множества	Создать телефонную книгу на dict, реализовать поиск по имени и удаление записи	Умение использовать словари (dict) и множества (set)	2
10	Алгоритмы сортировки и поиска	Реализовать простую сортировку выбором и функцию бинарного поиска в отсортированном списке	Знание базовых алгоритмов сортировки и поиска, умение их программировать	2
				20

СРС

№	Тема	Задание на выполнение	Ожидаемые результаты	Час
1	Установка и настройка среды Python	Установить Python и IDE (VS Code/PyCharm), создать виртуальное окружение	Рабочая среда готова, умеете создавать venv	1
2	Основы синтаксиса и переменные	Написать скрипт с разными типами переменных и вывести их на консоль	Понимание базовых конструкций и типов	2
3	Ввод/вывод	Разработать программу, читающую ввод пользователя и записывающую в файл	Умение работать с input(), print(), файловыми операциями	2
4	Строки и методы строк	Обработать текст: убрать пробелы, изменить регистр, подсчитать символы	Владение методами strip(), lower(), count()	2

№	Тема	Задание на выполнение	Ожидаемые результаты	Час
5	Списки и операции с ними	Реализовать операции: добавление, удаление, срезы, сортировка	Умение управлять коллекцией списков	2
6	Кортежи и их применение	Написать программу возврата пары (минимум, максимум) из списка	Понимание неизменяемых последовательностей	1
7	Словари	Создать телефонную книгу, реализовать поиск, добавление и удаление записей	Владение dict и его методами	2
8	Множества	Проанализировать два списка и вывести их пересечение и разность	Умение применять set для операций	1
9	Условные операторы	Написать калькулятор с ветвлением по символу операции	Применение if/elif/else	2
10	Циклы for и while	Подсчитать сумму цифр числа и вывести факториал	Умение организовать циклические вычисления	2
11	Функции	Разработать функцию проверки палиндрома со строкой в качестве аргумента	Понимание определения и вызова функций	2
12	Аргументы функций	Написать функцию со значениями по умолчанию и именованными аргументами	Владение позиционными и именованными параметрами	2
13	Лямбда-функции и map/filter	Отфильтровать список чётных чисел и возвести их в квадрат	Умение применять анонимные функции	2
14	Списковые генераторы	Создать список квадратов чисел от 1 до 20 с помощью генератора	Понимание list comprehension	2
15	Кортежи-генераторы и dict-компрехеншн	Построить словарь {число: факториал} для 1...10	Владение генераторами и dict comprehension	2

№	Тема	Задание на выполнение	Ожидаемые результаты	Час
16	Модули и пакеты	Разбить свой код на модули, импортировать их	Понимание структуры пакета и механизма импорта	2
17	Работа с файлами CSV	Считать CSV-файл студентов и вывести топ-3 по среднему баллу	Умение пользоваться модулем csv	2
18	JSON-сериализация	Сериализовать словарь в JSON и обратно	Владение json.dumps/loads	2
19	Исключения	Написать калькулятор с перехватом деления на ноль и неверного ввода	Понимание try/except/finally	2
20	Классы и объекты	Реализовать класс BankAccount с методами deposit, withdraw	Основы ООП: атрибуты, методы, инкапсуляция	2
21	Наследование	Создать подклассы SavingsAccount и CheckingAccount	Понимание переиспользования кода через наследование	2
22	Магические методы	Реализовать __str__, __add__ для класса Vector	Умение добавлять поведение через dunder-методы	2
23	Итераторы и __iter__ / __next__	Написать свой итератор для генерации бесконечной последовательности	Понимание протокола итераторов	2
24	Генераторы и yield	Создать генератор простых чисел	Владение ленивыми вычислениями через yield	2
25	Декораторы	Написать декоратор логирования вызовов функции	Понимание трансформации функций	2
26	Управление контекстом (with)	Использовать with для безопасного открытия файла	Умение реализовывать контекстные менеджеры	2
27	Регулярные выражения	Найти все e-mail адреса в тексте	Владение модулем re	2

№	Тема	Задание на выполнение	Ожидаемые результаты	Час
28	Работа с датой и временем	Вычислить разницу между двумя датами	Умение пользоваться datetime и timedelta	2
29	Логирование	Настроить logging для записи операций приложения	Понимание уровней логирования и обработчиков	2
30	Тестирование с unittest	Написать тесты для функций из предыдущих заданий	Основы модульного тестирования	2
31	Профилирование	Проанализировать время выполнения функции сортировки списка	Владение cProfile и оптимизацией кода	2
32	Многопоточность	Запустить три потока, выполняющих разные вычисления	Понимание threading и проблем синхронизации	2
33	Асинхронность	Создать асинхронную функцию для загрузки веб-страниц	Владение asyncio и aiohttp	2
34	Веб-скрапинг	Собрать заголовки новостей с сайта с помощью BeautifulSoup	Умение делать HTTP-запросы и парсить HTML	2
35	Работа с базой SQLite	Создать БД задач, реализовать CRUD-операции через sqlite3	Понимание интеграции Python и SQL	2
				70

СРСП

№	Тема	Задание на выполнение	Ожидаемые результаты	Часы
1	Установка и виртуальные окружения	Установить Python, создать виртуальное окружение и активировать его	Рабочая среда с отдельным venv для проектов	2
2	Базовые типы данных и операции	Написать скрипт с примерами чисел, строк, булевых и арифметических/логических операций	Понимание типов, приоритетов операций	2

№	Тема	Задание на выполнение	Ожидаемые результаты	Часы
3	Работа со строками	Разработать программу, выполняющую срезы, замену подстрок и подсчёт символов	Владение методами строк (split, replace, count, f-строки)	2
4	Списки и кортежи	Написать функции для добавления, удаления элементов в списке и попытаться изменить кортеж	Умение работать со списками и понимать неизменяемость кортежей	2
5	Словари и множества	Подсчитать частоту слов в тексте с помощью dict и продемонстрировать операции с set	Навыки использования dict для статистики и set для множественных операций	2
6	Условные операторы	Написать калькулятор со всеми четырьмя арифметическими операциями и проверкой некорректного ввода	Умение строить ветвления с if / elif / else	2
7	Циклы for и while	Решить задачу: найти все простые числа до N двумя способами	Владение циклами и базовыми алгоритмами поиска	2
8	Функции и области видимости	Реализовать функцию расчёта факториала с параметрами по умолчанию и протестировать её	Понимание объявлений, аргументов, возвращаемых значений	2
9	Работа с файлами	Написать скрипт, читающий CSV и выводящий агрегированную информацию	Навыки работы с open, csv.reader/writer	2
10	Обработка исключений	Создать программу ввода чисел с проверкой типов и ловлей ValueError	Знание конструкции try/except/finally	2
11	Классы и объекты	Разработать класс Person с атрибутами name и age, методами __str__ и __eq__	Основы ООП: создание классов, дандер-методы	2
12	Наследование и полиморфизм	Создать базовый класс Animal и подклассы Dog/Cat с разными методами speak()	Понимание наследования, переопределения методов	2
13	Модули и пакеты	Разбить свой код на несколько модулей, импортировать их в главный скрипт	Владение механизмом import, структурой пакетов	2
14	HTTP-запросы и JSON	Написать скрипт, запрашивающий публичное API (например, погода) и	Умение пользоваться библиотекой requests и модулем json	2

№	Тема	Задание на выполнение	Ожидаемые результаты	Часы
		выводящий данные в формате JSON		
15	Модульное тестирование	Создать набор тестов для нескольких функций с использованием unittest	Навыки написания юнит-тестов и использования TestCase	2
				30

10. Учебно-методическое, информационное и материально-техническое обеспечение дисциплины

1	Златопольский Д.М. Основы программирования на языке Python.-М:ДМК Пресс, 2017.-284 с.	https://disk.yandex.com/i/OtGyO5hQruuf-w
2	Джейсон Бриггс (перевод: Станислав Ломакин). Python для детей. Самоучитель по программированию 10.42 Мб, 2017 г., 321с.	https://disk.yandex.com/d/2z27YG17Qw4Now
3	М. И. Абдрахманов. Python. Визуализация данных. Matplotlib. Seaborn. Mayavi 7.8 Мб, 2020 г., 413с.	https://disk.yandex.com/d/2z27YG17Qw4Now
4	Мэттью Уилкс. Профессиональная разработка на Python 10.17 Мб, 2021 г., 503с.	https://disk.yandex.com/d/keWuUGBY8-91Gg
5	Дэвид Бизли. Python. Исчерпывающее руководство 5.87 Мб, 2023 г., 368с.	https://disk.yandex.com/d/VfiiS3LPfKTLzA
6	Билл Любанович. Простой Python. Современный стиль программирования [2-е издание] 7.63 Мб, 2021 г., 592с.	https://disk.yandex.com/d/08ABOMxgjRJJ_Q
7	Шпаргалка по Python 13.32 Мб, 18с.	https://disk.yandex.com/d/GsdgB_ASpb8EYw
8	Станислав Андреевич Чернышев. Основы программирования на Python: учебное пособие для вузов 92.11 Мб, 2022 г., 286с.	https://disk.yandex.com/d/4oWmyixXnxMwmg
9	Дмитрий Мусин. Самоучитель Python. Выпуск 0.2 726 Кб, 2016 г., 143с.	https://disk.yandex.com/d/MQiaHAK5m1WNA
10	Марк Лутц (перевод: И. В. Берштейн). Python. Карманный справочник [5-е издание] 64.01 Мб, 2015 г., 321с.	https://disk.yandex.com/d/Exj22HO5rLtaiw

11	Кэрл Вордерман (перевод: Станислав Ломакин). Программирование на Python. Иллюстрированное руководство для детей 35.6 Мб, 2016 г., 226с.	https://disk.yandex.com/d/ARQMUv6y0PxnZA
12	Рик Гаско. Простой Python просто с нуля 70.39 Мб, 2019 г., 258с	https://disk.yandex.com/d/ikFHJjDxaRe7lw
13	Тони Гэддис. Начинаем программировать на Python [4-е издание] 73.83 Мб, 2019 г., 770с.	https://disk.yandex.com/d/JBsHCHBbPVDLrA
14	Эрик Фримен (перевод: И. В. Василенко, В. Р. Гинзбург). Учимся программировать с примерами на Python [на русском (ru) и английском (en)] 77.97 Мб, 2020 г., 629с.	https://disk.yandex.com/d/ti1KMSh5TK7Q_A
15	Тиаго Антао. Сверхбыстрый Python 11.16 Мб, 372с.	https://disk.yandex.com/d/m7B7W_K1BbP7Fg
16	Антон Леонардович Марченко. Python: большая книга примеров 11.77 Мб, 2023 г., 362с.	https://disk.yandex.com/d/qyO8Cu2neLaB6A
17	Эл Свейгарт (перевод: Михаил Анатольевич Райтман). Учим Python, делая крутые игры 9.93 Мб, 2022 г., 418с.	https://disk.yandex.com/d/nD90qtlKYM5FHQ
18.	Ив Хилпиш (перевод: Сергей Викторович Черников). Python для финансистов 4.71 Мб, 2023 г., 208с.	https://disk.yandex.com/d/iOCZdb_EOessxw
19.	А. И. Широков М. О. Пышняк. Информатика. Разработка программ на языке Питон. Базовые языковые конструкции 17.92 Мб, 2020 г., 141с.	https://disk.yandex.com/d/t4VLu-SjxbaHYA
20	Д. М. Кольцов. Python. Создаем программы и игры [3-е издание] 146.76 Мб, 2022 г., 416с.	https://disk.yandex.com/d/YVeHw410BNIMAA
21	Дэн Бейдер. Чистый Python. Тонкости программирования для профи 6.94 Мб, 2018 г., 2018 г., 288с	https://disk.yandex.com/d/RB_D7BNsPzeQYQ
22	Алексей Николаевич Васильев. Python на примерах. Практический курс по программированию [3-е издание] 17.87 Мб, 2019 г., 434с.	https://disk.yandex.com/d/-t6LdIX8rqCXaA
23	Е. В. Дубовик Д. М. Кольцов. Справочник PYTHON. Кратко, быстро, под рукой 12.9 Мб, 2021 г., 290с.	https://disk.yandex.com/d/NHukfFsKFVF4jA

24	Майкл Доусон. Программируем на Python 2014 г., 5.21 Мб	https://disk.yandex.com/d/g_jQTUHa_SkkRQ
----	--	---

Литература в библиотеке СМУ

1	Алгоритмический тренинг. Решения практических задач на PYTHON и C++	Максим Иванов	2025 г
2	Программирование Python2 часть	Поляков К.Ю.	2023 г
3	Языки и методы программирования	Т.А.Макаровских	2018 г

Ссылки на сетевые ресурсы

1. <http://python.org> Официальный сайт языка Питон.
2. <http://www.python.ru> Русская документация языка Питон.
3. <http://ru.wikipedia.org/wiki/Python> Статья, посвященная Питону в русской Википедии.
4. <http://pyweb.ru> Список ссылок на русскоязычные тематические сайты о языке python
5. <http://www.python.com.ua> Сообщество украинских программистов на языке Питон.
6. <http://forum.ru-board.com/topic.cgi?forum=31&topic=1537> Питон на форумах Ру.Борд.

11. Вопросы для подготовки к экзамену

Экзаменационные вопросы

1. Что такое алгоритм и какие его основные свойства?
2. Каковы этапы разработки алгоритма?
3. Дайте определение псевдокода и приведите пример.
4. В чём разница между линейным и ветвящимся алгоритмом?
5. Как реализовать ветвление в языке программирования (синтаксис)?
6. Что такое цикл и какие виды циклов существуют?
7. В каких случаях предпочтителен цикл while, а в каких — for?
8. Объясните назначение оператора break и continue.
9. Что понимается под массивом и как он хранится в памяти?
10. Как обратиться к элементу массива по индексу?
11. Что такое строка и чем она отличается от массива символов?
12. Как перевернуть строку в коде?

13. Дайте определение рекурсии.
14. Приведите пример простой рекурсивной функции.
15. Что такое базовый и рекурсивный случаи?
16. В чём риск некорректной рекурсии?
17. Опишите алгоритм вычисления n -го числа Фибоначчи рекурсивно и итеративно.
18. Как оценить временную сложность рекурсивного алгоритма?
19. Что такое односвязный список и как ему добавить элемент?
20. Как удалить элемент из двусвязного списка?
21. Что такое стек и какие операции он поддерживает?
22. Объясните принципы работы очереди.
23. Как реализовать очередь на массивах?
24. Как реализовать стек на связном списке?
25. Что такое дерево поиска?
26. Как вставить узел в бинарное дерево поиска?
27. Опишите алгоритмы обхода дерева: pre-, in- и post-order.
28. Что такое граф и как его хранить в памяти?
29. В чём разница между матрицей смежности и списком смежности?
30. Опишите обход графа в глубину (DFS).
31. Опишите обход графа в ширину (BFS).
32. Как найти кратчайший путь в невзвешенном графе?
33. Что такое алгоритм Дейкстры?
34. Для чего нужен приоритетный набор (priority queue) в Дейкстре?
35. Что такое жадный алгоритм?
36. Приведите пример жадного решения задачи о рюкзаке (дробного).
37. Что такое динамическое программирование?
38. Опишите решение задачи о рюкзаке 0/1 через DP.
39. Что понимается под оптимизацией «разделяй и властвуй»?
40. Приведите пример алгоритма «разделяй и властвуй».
41. Как реализовать сортировку слиянием?
42. Как реализовать быструю сортировку (quicksort)?
43. В чём принцип работы сортировки пузырьком?
44. Почему пузырьковая сортировка считается неэффективной?
45. Что такое стабильность сортировки?
46. Какие алгоритмы сортировки являются устойчивыми?
47. Опишите алгоритм бинарного поиска.
48. В каких условиях бинарный поиск неприменим?

49. Как определить сложность алгоритма?
50. Что такое $O(1)$, $O(n)$, $O(n^2)$ в обозначениях «О-нотации»?
51. Как посчитать количество операций в цикле?
52. Как сравнить эффективность двух алгоритмов?
53. Почему важно учитывать «худший» и «средний» случаи?
54. Что такое амортизированная сложность?
55. Как измерить фактическое время выполнения программы?
56. Какие есть инструменты профилирования кода?
57. Как обрабатывать ввод-вывод из текстовых файлов?
58. Что такое бинарный ввод-вывод?
59. Как сериализовать объект в JSON?
60. Как распарсить XML в коде?
61. Что такое REST-API и как к нему подключаться?
62. Как отправить HTTP-GET запрос и обработать ответ?
63. В чём разница между GET и POST?
64. Как работать с сокетами TCP?
65. Как работает протокол UDP?
66. Что такое многопоточность?
67. Как создать поток в выбранном языке?
68. Какие проблемы возникают при параллельном доступе к данным?
69. Как использовать мьютексы и семафоры?
70. Что такое гонка данных (race condition)?
71. Как избежать deadlock (взаимоблокировки)?
72. Что такое пул потоков (thread pool)?
73. В чём преимущества асинхронного программирования?
74. Что такое callback, Promise (или Future)?
75. В каком случае применим async/await?
76. Что такое событийно-ориентированная модель?
77. Как организовать логирование в приложении?
78. Для чего нужен юнит-тест?
79. Как написать простой юнит-тест на функцию?
80. В чём польза тест-драйвен разработки (TDD)?
81. Что такое интеграционные тесты?
82. Как проводить тестирование производительности?
83. Какие есть методологии проектирования ПО?
84. В чём суть паттерна «Стратегия»?
85. В чём суть паттерна «Наблюдатель» (Observer)?

86. Что такое инъекция зависимостей (DI)?
87. Как реализовать фабрику (Factory) в коде?
88. Что такое интерфейс и абстрактный класс?
89. В чём разница между наследованием и композицией?
90. Как реализовать полиморфизм в ООП?
91. Что такое исключение и как его обрабатывать?
92. Почему не рекомендуется ловить «все» исключения?
93. Как использовать блок `finally` (или эквивалент)?
94. Что такое ресурсный утечка и как её избежать?
95. Как работать с базами данных через ORM?
96. В чём преимущества использования ORM?
97. Как написать SQL-запрос `SELECT` с `JOIN`?
98. Что такое транзакция и для чего она нужна?
99. Как реализовать `rollback` транзакции?
100. Что такое ACID-принципы в СУБД?
101. Как применять кеширование в приложении?
102. Когда имеет смысл использовать Redis или Memcached?
103. Что такое очередь сообщений и зачем она нужна?
104. Как настроить брокер RabbitMQ или Kafka (концептуально)?
105. Как использовать MapReduce для обработки больших данных?
106. Что такое Hadoop и Spark?
107. Какие есть подходы к машинному обучению?
108. Как реализовать простую линейную регрессию из scratch?
109. Что такое кластеризация k-means?
110. Как визуализировать данные при помощи библиотек?
111. Что такое RESTful архитектура?
112. Как организовать аутентификацию JWT?
113. Что такое OAuth2 и OpenID Connect?
114. Как обеспечить безопасность веб-приложения?
115. Какие есть типы уязвимостей OWASP Top-10?
116. Как предотвратить SQL-инъекции?
117. Как защититься от XSS-атак?
118. Что такое CSRF и как его предотвращать?
119. Как использовать HTTPS и сертификаты SSL/TLS?
120. Какие лучшие практики декораторов HTTP-заголовков для защиты?

12. Методические указания для преподавателей дисциплины «Алгоритмизация и программирование»

Формулировка целей и результатов обучения

- Чётко определите знания (модели алгоритмов, основные структуры данных, приёмы проектирования), умения (построение и анализ алгоритмов, отладка кода, оценка сложности) и навыки (работа с IDE, системой контроля версий, библиотеками) по итогам каждого модуля.
- Свяжите эти результаты с компетенциями: анализ проблем, выбор эффективных решений и документирование кода.

2. Структурирование содержательной части

- Разбейте курс на тематические блоки: базовые конструкции → структуры данных → сортировки и поиск → рекурсия и динамическое программирование → деревья и графы → параллельное и асинхронное программирование.
- Для каждого блока подготовьте подробный конспект лекции с псевдокодом, иллюстрациями, примерами «из жизни» и разбором ошибок.

3. Организация практических и лабораторных занятий

- Разработайте набор задач разного уровня сложности: от одношаговых упражнений до комплексных проектов. Каждую задачу сопровождайте шаблоном кода, тестовыми сценариями и рубрикой оценивания.
- В лабораториях стимулируйте студентов сначала предлагать псевдокод, затем реализовать прототип, а после — доводить решение до промышленного качества: документировать, писать unit-тесты, оптимизировать.

4. Использование учебных инструментов

- Настройте репозиторий (Git) для всего курса: студенты разрабатывают решения через форки, Pull Request-ы и peer-review.
- Используйте автоматические сборщики тестов (CI-систему) для быстрой обратной связи.
- Ведите в LMS раздел «Обзор типичных ошибок» с примерами и рекомендациями.

5. Оценивание и обратная связь

- Внедрите двухэтапную модель: формирующее (еженедельные онлайн-тесты, мини-задачи) и суммативное (лабораторный отчёт, итоговый проект).
- При оценивании проектов опирайтесь на заранее опубликованные рубрики (код-стиль, эффективность, читаемость, полнота тестирования).
- Проводите регулярные «код-ревью-сессии», где студенты выступают и разбирают решения друг друга.

6. Мотивация и вовлечение

- Организуйте микрозадачи-челленджи (например, «алгоритмическая пятиминутка») в начале семинара для разогрева.
- Проводите «мини-хакатоны» и игровые соревнования с призами за лучшую оптимизацию или креативность.

– Приглашайте выпускников и IT-профессионалов на открытые демонстрации и мастер-классы.

7. Индивидуализация и поддержка

– Выделите «группу риска» по успеваемости и регулярно проводите с ней консультации в формате office hours.

– Разрабатывайте адаптивные задания для сильных и для тех, кто отстаёт, и интегрируйте peer-tutoring: старшие студенты помогают младшим.

8. Преподавательская рефлексия и развитие

– Ведите журнал преподавателя: кратко фиксируйте, какие задания были восприняты лучше, а какие вызвали сложности, и почему.

– После каждого модуля собирайте анонимный фидбэк (что понравилось, что можно улучшить) и корректируйте программу.

– Поддерживайте е-портфолио, где хранятся записи micro-teaching-сессий, отзывы коллег и планы профессионального роста.

9. Взаимодействие с другими дисциплинами

– Интегрируйте задачи смежных курсов (математика, физика) для решения прикладных кейсов: численное моделирование, сбор данных, визуализация.

– Согласуйте требования по оформлению кода и докладов с преподавателями ПО и баз данных.

10. Стратегия непрерывного улучшения

– Применяйте цикл PDCA (Plan–Do–Check–Act) для каждого блока: планируйте изменения, реализуйте их, собирайте метрики успеваемости и вовлечённости, корректируйте методы.

– Делитесь на кафедре лучшими практиками, протоколируйте рекомендации и внедряйте новые инструменты.

Используя эти методические указания, преподаватель обеспечит глубокое освоение алгоритмической грамотности и практического программирования, сформирует у студентов навыки самостоятельного мышления, анализа и гибкой адаптации решений к разным задачам.

График проведенных занятий

№	Дата проведения практических занятий по расписанию	Дата проведения лабораторных занятий по расписанию	Дата проведения СРС по расписанию
1.			
2.			
3.			
4.			
5.			
6.			
7.			
8.			
9.			
10.			
11.			
12.			
13.			
14.			
15.			
16.			
17.			
18.			
19.			
20.			

21.			
22.			
23.			
24.			
25.			
26.			
27.			
28.			
29.			
30.			