Автономная некоммерческая организация высшего образования
«Санкт-Петербургская юридическая академия»

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ

Основы программирования и алгоритмизации

Направление подготовки 38.03.05 Бизнес-информатика

Квалификация выпускника Бакалавр

Направленность (профиль) Информационно-аналитическое сопровождение управленческой деятельности

Санкт-Петербург
2026

1. Планируемые результаты обучения по дисциплине, соотнесенные с индикаторами достижения компетенций

В процессе изучения дисциплины формируются результаты обучения, соотнесённые с закреплёнными за дисциплиной индикаторами достижения компетенций.

ОПК-3. Способен управлять процессами создания и использования продуктов и услуг в сфере информационно-коммуникационных технологий, в том числе разрабатывать алгоритмы и программы для их практической реализации.

Индикаторы достижения компетенций	Планируемые результаты обучения
ИД.1 (ОПК-3) Разрабатывает алгоритмы и программы для практической реализации задач в сфере информационно-коммуникационных технологий.	**Знает:** **РО-1ИД-1(ОПК-3)** основы алгоритмизации, программирования и технологии разработки программных решений.
	Умеет: **РО-2ИД-1(ОПК-3)** разрабатывать алгоритмы и программные компоненты для решения прикладных задач.
	Владеет: **РО-3ИД-1(ОПК-3)** навыками алгоритмизации и программной реализации прикладных задач.

2. Содержание оценочных средств для текущего контроля успеваемости

2.1. Примерный перечень тем докладов, рефератов, презентаций

1. Алгоритмы обработки табличных бизнес-данных.
2. Сравнение линейного и бинарного поиска.
3. Работа с файлами CSV средствами Python.
4. Обработка исключений в прикладных программах.
5. Автоматизированная проверка функций модульными тестами.
6. Оценка сложности алгоритмов сортировки.
7. Алгоритмы и программирование.
8. Типы данных и выражения.
9. Управляющие конструкции.
10. Функции и модульность.
11. Структуры данных.
12. Файлы и данные.
13. Основы объектного подхода и тестирования.
14. Прикладная задача бизнес-информатики.
15. Понятие алгоритма и его свойства.
16. Способы представления алгоритмов.
17. Переменные и типы данных.
18. Операторы и выражения.
19. Условные конструкции.
20. Циклы for и while.
21. Списки и кортежи.
22. Словари и множества.
23. Функции и параметры.
24. Область видимости переменных.
25. Модули и пакеты.
26. Работа с текстовыми файлами.
27. Чтение и запись CSV.
28. Обработка исключений.
29. Отладка программы.

30. Модульное тестирование.

2.2. Типовые задания

1. Кейс: аналитик проверяет ежедневные CSV-выгрузки продаж. Разработайте алгоритм обработки набора управленческих показателей и представьте его блок-схемой.

2. Кейс: интернет-магазин рассчитывает скидки и бонусы. Напишите программу на Python для чтения CSV-файла, проверки данных и расчета агрегированных показателей.

3. Кейс: отдел кадров объединяет данные сотрудников. Реализуйте функции для расчета KPI с проверкой входных параметров.

4. Кейс: финансовый отдел автоматизирует ежемесячный KPI-отчёт. Используйте списки, словари и множества для обработки данных о клиентах.

5. Кейс: аналитик проверяет ежедневные CSV-выгрузки продаж. Реализуйте сортировку и поиск, сравните вычислительную сложность вариантов.

6. Кейс: интернет-магазин рассчитывает скидки и бонусы. Добавьте обработку исключений и журналирование ошибок в программу обработки данных.

7. Кейс: отдел кадров объединяет данные сотрудников. Разработайте набор модульных тестов для созданных функций.

8. Кейс: финансовый отдел автоматизирует ежемесячный KPI-отчёт. Создайте небольшой консольный аналитический модуль, формирующий отчет по исходным данным.

9. Разбор практической ситуации. Программа считает средний чек делением выручки на число заказов; при нуле заказов падает. Исправьте функцию и напишите тесты для крайних случаев.

10. Разбор практической ситуации. Дан CSV с разным разделителем, пропущенными суммами и строковыми датами. Реализуйте валидатор, журнал ошибок и отчёт об отклонённых строках.

11. Разбор практической ситуации. В коде поиска клиентов используется вложенный цикл $O(n^2)$. Предложите вариант со словарём или множеством и оцените выигрыш сложности.

12. Разбор практической ситуации. Приложение рассчитывает скидку, но иногда даёт отрицательную стоимость. Найдите логическую ошибку по тест-кейсам и исправьте условия.

Требования к выполнению: решение должно отражать исходные данные, выбранный метод, промежуточные результаты, итоговый артефакт и пояснение принятых решений. В программных заданиях прилагаются воспроизводимый код и результаты проверок.

2.3 Вопросы для самоконтроля

1. Что такое алгоритм и какими свойствами он должен обладать.
2. Как переменные, типы данных и операции влияют на корректность программы.
3. Чем ветвление отличается от циклического алгоритма.
4. Когда целесообразно использовать список, словарь, множество и кортеж.
5. Что такое функция, параметры, возвращаемое значение и область видимости.
6. Как оценивать временную и пространственную сложность простых алгоритмов.
7. Какие ошибки возникают при работе с файлами и пользовательским вводом.
8. Что такое исключения и зачем нужна их обработка.
9. Как декомпозиция и модульность повышают сопровождаемость программы.
10. Как программно обрабатывать табличные данные для бизнес-задач.
11. Какие приемы тестирования применимы к небольшим программам.
12. Как документировать код и результаты вычислений для воспроизводимости анализа.
13. Понятие алгоритма и его свойства.
14. Способы представления алгоритмов.
15. Переменные и типы данных.
16. Операторы и выражения.
17. Условные конструкции.
18. Циклы for и while.
19. Списки и кортежи.

20. Словари и множества.
21. Функции и параметры.
22. Область видимости переменных.
23. Модули и пакеты.
24. Работа с текстовыми файлами.
25. Чтение и запись CSV.
26. Обработка исключений.
27. Отладка программы.
28. Модульное тестирование.
29. Линейный и бинарный поиск.
30. Алгоритмы сортировки.

3. Задания для проведения промежуточной аттестации по дисциплине

3.1. Перечень вопросов к зачету с оценкой

1. Понятие алгоритма и его свойства.
2. Способы представления алгоритмов.
3. Переменные и типы данных.
4. Операторы и выражения.
5. Условные конструкции.
6. Циклы for и while.
7. Списки и кортежи.
8. Словари и множества.
9. Функции и параметры.
10. Область видимости переменных.
11. Модули и пакеты.
12. Работа с текстовыми файлами.
13. Чтение и запись CSV.
14. Обработка исключений.
15. Отладка программы.
16. Модульное тестирование.
17. Линейный и бинарный поиск.
18. Алгоритмы сортировки.
19. Оценка вычислительной сложности.
20. Принципы структурного программирования.
21. Алгоритмы и программирование.
22. Типы данных и выражения.
23. Управляющие конструкции.
24. Функции и модульность.
25. Структуры данных.
26. Файлы и данные.
27. Основы объектного подхода и тестирования.
28. Прикладная задача бизнес-информатики.
29. Свойства корректного программного алгоритма.
30. Подбор структуры данных для задачи.
31. Проверка граничных условий алгоритма.
32. Декомпозиция программы на функции.
33. Документирование результатов выполнения программы.
34. Сравнение сложности способов поиска.
35. Различие значения и ссылки на объект.
36. Проверка истинности логических выражений.
37. Итерации по элементам списка.
38. Словарь для подсчёта частот.
39. Сравнение изменяемых и неизменяемых коллекций.

40. Использование return в функциях.

3.2. Перечень вопросов к экзамену

1. Текстовые файлы как источник данных программы.
2. Открытие и закрытие файла.
3. Работа с кодировками текстовых данных.
4. Чтение и запись записей CSV.
5. Проверка корректности входных файлов.
6. Обработка ошибок чтения данных.
7. Исключения при выполнении программы.
8. Конструкция try–except.
9. Использование блока finally.
10. Контекстный менеджер работы с файлами.
11. Разбиение программы на модули.
12. Импорт функций из модуля.
13. Назначение пакетов программных модулей.
14. Аргументы и возвращаемые значения функций.
15. Локальные и глобальные переменные.
16. Передача коллекций в функции.
17. Принцип единственной ответственности функции.
18. Классы и объекты в программировании.
19. Атрибуты и методы объекта.
20. Принцип инкапсуляции.
21. Отладка прикладной программы.
22. Использование точек останова.
23. Модульное тестирование функций.
24. Проверка граничных случаев.
25. Понятие тестового набора.
26. Обработка некорректного пользовательского ввода.
27. Линейный поиск в коллекции.
28. Бинарный поиск в отсортированном массиве.
29. Условия корректности алгоритма сортировки.
30. Сравнение алгоритмов сортировки.
31. Временная сложность алгоритма.
32. Пространственная сложность алгоритма.
33. Оценка порядка роста Big O.
34. Декомпозиция задачи обработки заказов.
35. Подсчёт агрегатов по табличным данным.
36. Фильтрация и группировка записей в программе.
37. Формирование отчёта из входного CSV.
38. Проверка воспроизводимости вычислений.
39. Документирование прикладного кода.
40. Защита программы от типовых ошибок данных.

4. Шкала оценивания результатов промежуточной аттестации и критерии выставления оценок

Для оценивания результатов промежуточной аттестации применяется шкала оценивания, включающая следующие оценки: «отлично», «хорошо», «удовлетворительно», «неудовлетворительно».

Экзамен (зачет с оценкой). Критерии выставления оценок

Знания обучающихся оцениваются путем выставления по результатам ответа обучающегося итоговой оценки «отлично», либо «хорошо», либо «удовлетворительно», либо «неудовлетворительно».

Оценка «отлично» выставляется в случае:

- полного, правильного и уверенного изложения обучающимся учебного материала по каждому из вопросов билета;
- уверенного владения обучающимся понятийно-категориальным аппаратом учебной дисциплины;
- логически последовательного, взаимосвязанного и правильно структурированного изложения обучающимся учебного материала, умения устанавливать и прослеживать причинно-следственные связи между событиями, процессами и явлениями, о которых идет речь в вопросах билета;
- приведения обучающимся надлежащей аргументации, наличия у обучающегося логически и нормативно обоснованной точки зрения при освещении проблемных, дискуссионных аспектов учебного материала по вопросам билета;
- лаконичного и правильного ответа обучающегося на дополнительные вопросы преподавателя.

Оценка «хорошо» выставляется в случае:

- недостаточной полноты изложения обучающимся учебного материала по отдельным (одному или двум) вопросам билета при условии полного, правильного и уверенного изложения учебного материала по, как минимум, одному вопросу билета;
- допущения обучающимся незначительных ошибок и неточностей при изложении учебного материала по отдельным (одному или двум) вопросам билета;
- допущения обучающимся незначительных ошибок и неточностей при использовании в ходе ответа отдельных понятий и категорий дисциплины;
- нарушения обучающимся логической последовательности, взаимосвязи и структуры изложения учебного материала по отдельным вопросам билета, недостаточного умения обучающегося устанавливать и прослеживать причинно-следственные связи между событиями, процессами и явлениями, о которых идет речь в вопросах билета;
- приведения обучающимся слабой аргументации, наличия у обучающегося недостаточно логически и нормативно обоснованной точки зрения при освещении проблемных, дискуссионных аспектов учебного материала по вопросам билета;
- допущения обучающимся незначительных ошибок и неточностей при ответе на дополнительные вопросы преподавателя.

Любой из указанных недостатков или их определенная совокупность могут служить основанием для выставления обучающемуся оценки «хорошо».

Оценка «удовлетворительно» выставляется в случае:

- невозможности изложения обучающимся учебного материала по любому из вопросов билета при условии полного, правильного и уверенного изложения учебного материала по как минимум одному из вопросов билета;
- допущения обучающимся существенных ошибок при изложении учебного материала по отдельным (одному или двум) вопросам билета;
- допущении обучающимся ошибок при использовании в ходе ответа основных понятий и категорий учебной дисциплины;
- существенного нарушения обучающимся или отсутствия у обучающегося логической последовательности, взаимосвязи и структуры изложения учебного материала, неумения обучающегося устанавливать и прослеживать причинно-следственные связи между событиями, процессами и явлениями, о которых идет речь в вопросах билета;
- отсутствия у обучающегося аргументации, логически и нормативно обоснованной точки зрения при освещении проблемных, дискуссионных аспектов учебного материала по вопросам билета;
- невозможности обучающегося дать ответы на дополнительные вопросы преподавателя.

Любой из указанных недостатков или их определенная совокупность могут служить основанием для выставления обучающемуся оценки «удовлетворительно».

Оценка «неудовлетворительно» выставляется в случае:

- отказа обучающегося от ответа по билету с указанием, либо без указания причин;
- невозможности изложения обучающимся учебного материала по двум или всем вопросам билета;
- допущения обучающимся существенных ошибок при изложении учебного материала по двум или всем вопросам билета;
- скрытное или явное использование обучающимся при подготовке к ответу нормативных источников, основной и дополнительной литературы, конспектов лекций и иного вспомогательного материала, кроме случаев специального указания или разрешения преподавателя;
- не владения обучающимся понятиями и категориями данной дисциплины;
- невозможность обучающегося дать ответы на дополнительные вопросы преподавателя.

Любой из указанных недостатков или их совокупность могут служить основанием для выставления обучающемуся оценки «неудовлетворительно».

Обучающийся имеет право отказаться от ответа по выбранному билету с указанием, либо без указания причин и взять другой билет. При этом с учетом приведенных выше критериев оценка обучающемуся должна быть выставлена на один балл ниже заслуживаемой им.

Дополнительные вопросы могут быть заданы обучающемуся в случае:

- необходимости конкретизации и изложенной обучающимся информации по вопросам билета с целью проверки глубины знаний отвечающего по связанным между собой темам и проблемам;
- необходимости проверки знаний обучающегося по основным темам и проблемам курса при недостаточной полноте его ответа по вопросам билета.

При проведении промежуточной аттестации в форме тестирования с использованием шкалы, включающей оценки «отлично», «хорошо», «удовлетворительно», «неудовлетворительно», оценивание результата проводится следующим образом:

«**Отлично**» - получают обучающиеся в том случае, если верные ответы составляют от 80% до 100% от общего количества;

«**Хорошо**» - получают обучающиеся в том случае, если верные ответы составляют от 65 до 79% от общего количества;

«**Удовлетворительно**» - получают обучающиеся в том случае, если верные ответы составляют 50 –64 % правильных ответов;

«**Неудовлетворительно**» - работа, содержащая менее 50% правильных ответов.

5. Задания для диагностической работы (количество 20)

1. Вопрос: Что такое алгоритм и какими свойствами он обладает?

Примерный ответ: Алгоритм – конечная последовательность однозначно определённых действий для решения класса задач. К его свойствам относят определённость, дискретность, конечность, результативность и применимость к заданным входным данным.

2. Вопрос: Какими способами можно представить алгоритм?

Примерный ответ: Алгоритмы представляют словесным описанием, псевдокодом, блок-схемой или программным кодом. Способ выбирают в зависимости от цели: объяснения идеи, формализации логики либо непосредственного исполнения компьютером.

3. Вопрос: Для чего используются переменные и типы данных?

Примерный ответ: Переменная связывает имя с хранимым значением, а тип данных определяет вид значения и допустимые операции. Примерами типов являются целые и вещественные числа, строки и логические значения.

4. Вопрос: Чем оператор отличается от выражения?

Примерный ответ: Выражение вычисляется и даёт значение на основе констант, переменных и операций. Оператор выполняет действие программы, например присваивание, ветвление или вызов процедуры.

5. Вопрос: Как работают условные конструкции в программе?

Примерный ответ: Условные конструкции выбирают ветвь выполнения на основании логического выражения. Конструкция if выполняет блок при истинном условии, а ветви elif и else позволяют обработать альтернативные случаи.

6. Вопрос: Чем различаются циклы for и while?

Примерный ответ: Цикл for обычно применяют для последовательного перебора элементов или диапазона значений. Цикл while повторяет блок, пока условие истинно; при его использовании важно обеспечить возможность завершения.

7. Вопрос: Чем список отличается от кортежа?

Примерный ответ: В Python список представляет изменяемую последовательность элементов, а кортеж – неизменяемую последовательность. Оба типа допускают индексацию и перебор, но выбор зависит от необходимости изменять состав данных.

8. Вопрос: Для чего используют словари и множества?

Примерный ответ: Словарь хранит соответствия между уникальными ключами и значениями, обеспечивая удобный поиск по ключу. Множество содержит уникальные элементы и поддерживает операции объединения, пересечения и разности.

9. Вопрос: Что такое функция и как передаются параметры?

Примерный ответ: Функция – именованный блок кода, который выполняет определённую задачу и может возвращать результат. Параметры задают входные данные функции и позволяют использовать один алгоритм с различными аргументами.

10. Вопрос: Что понимают под областью видимости переменных?

Примерный ответ: Область видимости определяет, в каких частях программы доступно имя переменной. В Python различают, в частности, локальные и глобальные имена; корректное разграничение уменьшает нежелательные зависимости функций.

11. Вопрос: Чем модуль отличается от пакета?

Примерный ответ: В Python модуль обычно представляет файл с определениями функций, классов и переменных. Пакет объединяет связанные модули в структуру, упрощая организацию и повторное использование кода.

12. Вопрос: Как организуют чтение и запись текстовых файлов?

Примерный ответ: Для работы с текстом открывают файл в нужном режиме и кодировке, затем читают или записывают данные. Контекстный менеджер with обеспечивает своевременное закрытие файла даже при возникновении ошибки.

13. Вопрос: Какие особенности имеет формат CSV?

Примерный ответ: CSV хранит табличные данные в строках с разделителями полей и не сохраняет типы и оформление ячеек. При чтении и записи в Python используют модуль csv с учётом кодировки, разделителя и правил экранирования.

14. Вопрос: Для чего нужна обработка исключений?

Примерный ответ: Исключения сигнализируют об ошибочных или нестандартных ситуациях во время выполнения программы. Конструкция try/except позволяет перехватить ожидаемую ошибку и выполнить предусмотренную обработку вместо аварийного завершения.

15. Вопрос: Какие действия включает отладка программы?

Примерный ответ: Отладка направлена на поиск причин неправильного поведения кода. Используют воспроизведение ошибки, проверку значений переменных, точки останова, журналы и пошаговое выполнение, после чего подтверждают исправление тестом.

16. Вопрос: Что такое модульное тестирование?

Примерный ответ: Модульное тестирование проверяет отдельные функции или небольшие компоненты программы в контролируемых условиях. Тесты сравнивают фактический результат с ожидаемым и помогают обнаруживать регрессии после изменений кода.

17. Вопрос: Чем линейный поиск отличается от бинарного?

Примерный ответ: Линейный поиск последовательно просматривает элементы и работает без предварительной сортировки. Бинарный поиск делит отсортированный диапазон пополам, уменьшая число сравнений при увеличении размера данных.

18. Вопрос: Для чего применяют алгоритмы сортировки?

Примерный ответ: Алгоритмы сортировки располагают элементы в заданном порядке по определённому ключу. Существуют разные методы, например сортировка вставками и слиянием, которые отличаются скоростью, затратами памяти и устойчивостью.

19. Вопрос: Что показывает вычислительная сложность алгоритма?

Примерный ответ: Вычислительная сложность характеризует зависимость потребления времени или памяти от размера входных данных. Асимптотическая нотация O позволяет сравнивать рост затрат, например $O(n)$ и $O(n^2)$, не привязываясь к конкретному компьютеру.

20. Вопрос: На каких принципах основано структурное программирование?

Примерный ответ: Структурное программирование использует последовательное выполнение, ветвление и циклы, а также декомпозицию задачи на понятные части. Такой подход улучшает читаемость, тестируемость и сопровождение программ.