**Вопросы к экзамену по дисциплине «Инженерное программирование»**

1. Опишите роль переменных в языке программирования Python и их значение в процессе программирования.
2. Объясните основные характеристики типов данных в Python и классифицируйте их.
3. Разберите виды операторов в Python и их задачи в программах.
4. Осветите значение ветвящихся операторов в программировании и их принципы работы.
5. Теоретически объясните процесс обработки с помощью операторов повторения.
6. Опишите особенности строкового типа данных и его преимущества в программировании.
7. Проанализируйте основные возможности строковых функций в Python и области их применения.
8. Объясните ключевые особенности списка (list) как типа данных и его использование в программах.
9. Расскажите о типе данных «множества» (set) и его отличиях от других типов данных.
10. Объясните значение одномерных массивов и их роль в процессе программирования.
11. Опишите особенности двумерных массивов и их применение в программах.
12. Проанализируйте процесс создания функций и их преимущества в программировании.
13. Раскройте принципы создания рекурсивных функций и их использование для решения инженерных задач.
14. Опишите технику работы с файлами в Python и их роль на практике.
15. Проанализируйте возможности стандартных библиотек Python и их преимущества в программировании.
16. Объясните, как решать математические и временные задачи с помощью модулей math и datetime.
17. Теоретически объясните методы генерации случайных данных с помощью модуля random.
18. Расскажите о принципах объектно-ориентированного программирования (ООП) и их преимуществах для создания программных систем.
19. Объясните принципы наследования и их особенности применения в программировании.
20. Раскройте понятие множественного наследования и его роль в объектно-ориентированном программировании.
21. Опишите возможности создания интерфейсов с помощью графических модулей Python.
22. Теоретически объясните процесс создания графических объектов с помощью модуля Turtle.
23. Объясните основные принципы создания графических интерфейсов с помощью модуля Tkinter.
24. Раскройте возможности работы со сложными виджетами в Tkinter.
25. Опишите этапы настройки среды программирования Python и её значение в процессе разработки.
26. Объясните технологии анализа данных с использованием библиотек NumPy, Pandas и Matplotlib.
27. Опишите процесс создания классов и объектов и их роль в программировании.
28. Раскройте преимущества создания программных систем с использованием принципов наследования и полиморфизма.
29. Объясните подключение к базе данных и значение операций CRUD в программных системах.
30. Расскажите о понятии MicroPython и его применении в IoT-устройствах.
31. Объясните принципы создания структуры модель–вид–шаблон (Model-View-Template) в Django.
32. Осветите преимущества работы с шаблонами в Django и их значение в процессе программирования.
33. Объясните основные принципы работы с базой данных в Django и их возможности.
34. Раскройте теоретические основы статистического анализа методов с помощью Python.
35. Опишите принципы обработки данных в IoT-устройствах с использованием Python.
36. Объясните преимущества обмена данными с помощью асинхронных запросов.
37. Раскройте теоретические аспекты криптографической защиты данных и их применение с библиотеками Python.
38. Теоретически обоснуйте различия между библиотеками TensorFlow и PyTorch.
39. Опишите принципы создания простого чат-бота с использованием Python.
40. Объясните теоретические основы анализа алгоритмов машинного обучения с использованием Python.
41. Осветите принципы оптимизации взаимодействия между страницами в проектах Django.
42. Расскажите о возможностях управления системами Django с помощью Docker Compose.
43. Объясните принципы создания графических интерфейсов с использованием модуля Turtle.
44. Теоретически опишите процесс создания графических приложений с помощью Tkinter.
45. Объясните теоретические основы создания асинхронных алгоритмов с использованием Python.
46. Осветите принципы создания библиотек и пакетов Python с теоретической точки зрения.
47. Объясните теоретические основы создания моделей баз данных и их использования в Django.
48. Раскройте основы статистического анализа и визуализации данных с использованием Python.
49. Осветите принципы организации управления в IoT-системах с использованием MicroPython.
50. Объясните теоретические основы организации автоматизированных процессов с использованием Python.
51. Каковы преимущества использования языка Python в научных и инженерных расчётах?
52. Объясните принципы инкапсуляции и наследования в ООП и приведите практический пример.
53. Как структурировать проект на Python, используя модули и пакеты, чтобы код был удобен для поддержки?
54. Какие способы импортирования модулей в Python вы знаете и в каких ситуациях каждый из них применим?
55. Опишите основные операции с текстовыми файлами в Python. Как безопасно читать и записывать данные?
56. В чём отличие реляционных баз данных от нереляционных и как выбрать подходящую для проекта?
57. Какие классические структуры данных (список, стек, очередь) вы знаете и в чём их основные отличия?
58. Как сформировать тест-кейсы, которые позволят выявить как можно больше ошибок в программе?
59. В чём заключается суть принципа полиморфизма и как он помогает при проектировании программного кода?
60. Как организовать ветвление (branching) в Git при работе над проектом в команде?
61. Что такое жизненный цикл разработки программного обеспечения и какие основные фазы в него входят?
62. Каково назначение модульных тестов и чем они отличаются от интеграционных?
63. Опишите базовые подходы к оптимизации кода (например, профилирование и рефакторинг).
64. Какими методами можно повысить производительность Python-приложения, помимо оптимизации алгоритмов?
65. Какие принципы лежат в основе объектно-ориентированного подхода и почему они полезны в крупных проектах?
66. Как реализовать простейшее графическое приложение на Python с помощью Tkinter?
67. В чём ключевое отличие между параллельным и асинхронным выполнением задач в Python?
68. Как синхронизировать потоки в Python для безопасного доступа к общим ресурсам?
69. Опишите основные методы машинного обучения и укажите, в каких случаях их целесообразно применять.
70. Как в Python можно взаимодействовать с API? Приведите пример отправки запроса и обработки ответа.
71. Что такое Agile и как эта методология влияет на процесс планирования задач?
72. Как DevOps-практики помогают обеспечить непрерывную интеграцию и развертывание (CI/CD)?
73. Объясните разницу между TCP и UDP при работе с сетевыми сокетами.
74. Как в Python можно организовать простой TCP-сервер и какой код потребуется на стороне клиента?
75. В чём преимущество модульной архитектуры проектов с точки зрения командной разработки?
76. Как правильно документировать код и на каких этапах это особенно важно?
77. Что такое REST и какие методы HTTP-запросов чаще всего применяются в веб-приложениях?
78. В чём отличие между локальным и глобальным пространствами имён в Python и как правильно с ними работать?
79. Какие факторы следует учитывать при выборе алгоритма оптимизации для конкретной научной задачи?
80. Объясните, как NumPy повышает эффективность работы с многомерными массивами по сравнению со списками.
81. Как с помощью библиотеки Matplotlib построить простой график функции и сохранить его в файл?
82. Какие методы тестирования вы используете при проверке корректности работы функций, связанных с базами данных?
83. Как создать и использовать виртуальное окружение в Python и почему это важно?
84. В чём заключается цель линейного программирования и как можно решить такую задачу в Python?
85. В чём состоят основные этапы отладки программы и какие средства для этого предоставляет Python?
86. Какой подход более эффективен при реализации многозадачности в Python: threading, multiprocessing или asyncio?
87. Что следует учесть при проектировании структуры базы данных, если объём данных может значительно вырасти?
88. Как определить, что в коде имеются «узкие места» и какие инструменты Python используются для профилирования?
89. Что такое code review и какие основные моменты нужно проверять при анализе чужого кода?
90. Какие преимущества даёт наследование в ООП при масштабировании программного проекта?
91. Как организовать резервное копирование и восстановление базы данных, чтобы минимизировать риски?
92. Зачем нужен менеджер пакетов pip и как с его помощью устанавливать зависимости проекта?
93. Какими методами можно обеспечить безопасность при работе с сетевыми приложениями и веб-сервисами?
94. В чём заключаются ключевые отличия Scrum от Kanban, и как они влияют на распределение задач?
95. Как реализовать простейшую асинхронную функцию в Python с помощью ключевых слов async и await?
96. Какие типы тестов (юнит, интеграционные, системные, приёмочные) вы знаете и чем они различаются?
97. Как использовать GitFlow или похожую стратегию ветвления, чтобы повысить эффективность командной разработки?
98. В чём состоят современные тенденции развития инженерного программирования и как они влияют на выбор инструментов?
99. Расскажите об основных принципах написания «чистого кода» (Clean Code) в Python и о том, как соблюдение этих принципов влияет на качество и сопровождаемость проектов.
100. Как эффективно тестировать и отлаживать асинхронные приложения на Python, чтобы обеспечить высокую производительность и надёжность?