рассмотрено на заседании
пед. совета Протокол № 1 от «04»
сентября 2024 г.

УТВЕРЖДАЮ

Директор ООО «ТОП-ТАЙМ»
_____А.И. Даишева
М.П.

Приказ № 1 от «27» августа 2024г.

ОБРАЗОВАТЕЛЬНАЯ ПРОГРАММА
дополнительного образования, дополнительного образования детей и взрослых.
Дополнительная общеразвивающая программа
«курс основы Python-разработчик»

г. Казань
2024

Содержание программы

1. Пояснительная записка

1.1. Дополнительная общеразвивающая программа «Python-разработчик» (далее - Программа) разработана на основе нормативных документов, определяющих правовые позиции и стратегические перспективы развития дополнительного образования в Российской Федерации:

— Федерального закона от 29.12.2012 № 273 -ФЗ «Об образовании в Российской Федерации»;

— Приказа Министерства просвещения РФ от 27 июля 2022 г. N 629 «Об утверждении Порядка организации и осуществления образовательной деятельности по дополнительным общеобразовательным программам»;

— Методических рекомендаций Минобрнауки России по проектированию дополнительных общеразвивающих программ (включая разноуровневые программы) от 18.11.2015 г. № 09-3242;

— Постановление Правительства РФ от 11 октября 2023 г. № 1678 “Об утверждении Правил применения организациями, осуществляющими образовательную деятельность, электронного обучения, дистанционных образовательных технологий при реализации образовательных программ;

— Приказа Министерства труда и социальной защиты РФ от 22 сентября 2021 г. N 652н "Об утверждении профессионального стандарта «Педагог дополнительного образования детей и взрослых»;

— Приказ Министерства здравоохранения и социального развития РФ от 26 августа 2010 г. N 761н «Об утверждении Единого квалификационного справочника должностей руководителей, специалистов и служащих, раздел «Квалификационные характеристики должностей работников образования».

— Дополнительное образование - это вид образования, который направлен на всестороннее удовлетворение образовательных потребностей человека в интеллектуальном, духовно-нравственном, физическом и (или) профессиональном совершенствовании и не сопровождается повышением уровня образования.

1.2. Актуальность программы:

Актуальность программы обусловлена тем, что навыки программирования на языке Python становятся всё более востребованными в современном мире. Python является одним из самых популярных и простых в освоении языков программирования, что делает его отличным

выбором для начинающих. Знание основ программирования и понимание принципов объектно-ориентированного программирования открывает широкие возможности для дальнейшего профессионального развития. Кроме того, с развитием цифровых технологий умение создавать простые приложения, включая чат-боты, становится ценным навыком.

1.3. Категория обучающихся: лица старше 16 лет, желающие освоить программу «Python-разработчик», без предъявления требований к уровню образования обучающихся.

1.4. Цель и задачи программы

Цели программы: освоение основ программирования на Python, включая основные принципы ООП, работу со стандартными библиотеками и создание простых приложений.

Задачи программы:

— освоить базовый синтаксис языка Python;
— научить работать в профессиональных средах разработки
— освоить основы использования системы контроля версий (git)
— изучить основные принципы объектно-ориентированного программирования (ООП);
— научиться работать со стандартными библиотеками Python;
— овладеть навыками создания простых программ и приложений;
— изучить основы создания Telegram-ботов с использованием специализированных библиотек Python;
— развить навыки самостоятельного решения задач и написания кода.

1.5. Объем программы: 107 часов трудоемкости (3 месяца).

1.6. Форма обучения: очная с применением частично электронного обучения дистанционных образовательных технологий.

1.7. Документ, выдаваемый после завершения обучения – сертификат о прохождении обучения установленного образца.

1.8. Виды занятий:

Обучение осуществляется в форме занятий в группах:

— лекции;
— видео уроки;
— практические занятия;
— итоговый проект.

2. Результаты освоения программы

2.1. В результате освоения содержания Программы слушатели должны:

уметь:

- использовать базовый синтаксис Python для написания программ;
- применять основные принципы объектно-ориентированного программирования (ООП);
- работать со стандартными библиотеками Python для решения типовых задач;
- создавать простые программы, включая Telegram-ботов;
- интегрировать программные решения с внешними сервисами через API.

знать:

- основные понятия и конструкции языка Python;
- принципы объектно-ориентированного программирования (ООП);
- функционал стандартных библиотек Python;
- технологии и инструменты для создания Telegram-ботов;

2.2. Дополнительная общеразвивающая программа «Python-разработчик» заканчивается итоговым экзаменом.

Лицам, успешно освоившим соответствующую дополнительную общеразвивающую программу и сдавшим итоговый экзамен, выдается сертификат об успешном прохождении обучения.

На каждом уровне обучения осуществляется контроль знаний:

а) Текущий контроль

Цель – определение степени прогресса слушателей в процессе занятий; выявление трудностей. Итоги тестирования и опроса позволяют скорректировать темп и методику проведения занятий.

б) Итоговый контроль проводится в конце обучения в форме защиты проекта.

3. Учебный план

дополнительной общеразвивающей программы «основы Python-разработчик»

Категория слушателей:
- лица старше 16 лет, желающие освоить программу «основы Python-разработчик», без предъявления требований к уровню образования обучающихся.

Срок обучения: 107 часов трудоемкости (3 месяца).

Форма обучения: очная с применением частично электронного обучения дистанционных образовательных технологий.

№ п/п	Наименование тем	Всего, Час/мин	В том числе		
			Лекции, час/ мин	Домашнее задание/практическое занятие, час/мин	Форма контроля знаний
1	Тема 1. Вводное занятие (онбординг)	1	1	-	-
2	Тема 2. Основы Python	2	1	1	Выполнение задания
3	Тема 3. Операторы и выражения	2	1	1	Выполнение задания
4	Тема 4. Условный оператор if, ветвления	5	2	3	Выполнение задания
5	Тема 5. Цикл while	3	1	2	Выполнение задания
6	Тема 6. Цикл for	7	2	5	Выполнение задания
7	Тема 7. Вложенные циклы	2	1	1	Выполнение задания
8	Тема 8. Числа с плавающей точкой	4	1	3	Выполнение задания

9	Тема 9. Функции	3	1	2	Выполнение задания
10	Тема 10. Система контроля версий git	6	3	3	Выполнение задания
11	Тема 11. Списки и методы работы с ними	3	1	2	Выполнение задания
12	Тема 12. List comprehensions	3	1	2	Выполнение задания
13	Тема 13. Базовые коллекции: строки, словари, множества, кортежи	9	3	6	Выполнение задания
14	Тема 14. Рекурсия	3	1	2	Выполнение задания
15	Тема 15. Работа с файлами	4	1	3	Выполнение задания
16	Тема 16. Исключения и работа с ошибками	2	1	1	Выполнение задания
17	Тема 17. Объектно-ориентированное программирование	10	3	7	Выполнение задания
18	Тема 18. Итераторы и генераторы	3	1	2	Выполнение задания
19	Тема 19. Декораторы	5	2	3	Выполнение задания
20	Тема 20. Библиотеки для работы с данными	4	1	3	Выполнение задания
21	Тема 21. Подготовка к итоговому проекту	8	6	2	Выполнение задания
22	Тема 22. Описание итогового проекта	16	4	12	Выполнение задания
23	Итоговый экзамен	2	-	2	Защита проекта
24	**Итого**	**107**	**39**	**68**	-

4. КАЛЕНДАРНЫЙ УЧЕБНЫЙ ГРАФИК

Режим занятий обучающихся регламентируется календарным учебным графиком, расписанием учебных занятий. Общая продолжительность занятий 3 месяца - 107 часов трудоемкости. Занятия проходят 2-3 раза в неделю, не более 10 часов в неделю.

Учебные темы	Номер недели											
	1	2	3	4	5	6	7	8	9	10	11	12
Вводное занятие (онбординг) для студентов blended-продуктов	1ч											
Основы Python	2ч											
Операторы и выражения	2ч											
Условный оператор if, ветвления	5ч											
Цикл while		3ч										
Цикл for		6ч	1ч									
Вложенные циклы			2ч									
Числа с плавающей точкой			4ч									
Функции				3ч								
Система контроля версий git				6ч								
Списки и методы работы с ними					3ч							
List comprehensions					3ч							
Базовые коллекции: строки, словари, множества, кортежи					4ч	5ч						
Рекурсия						3ч						
Работа с файлами						2ч	2ч					
Исключения и работа с ошибками							2ч					
Объектно-ориентированное программирование							5ч	5ч				
Итераторы и генераторы								3ч				
Декораторы								1ч	4ч			
Библиотеки для работы с данными									4ч			
Подготовка к итоговому проекту									2ч	6ч		
Описание итогового проекта										3ч	8ч	5ч
Итоговый экзамен												2ч

Сокращенные обозначения:
Ч – часы
М- минут

5. Рабочие программы

Тема 1. Вводное занятие (онбординг)

Приветствие и знакомство. Структура программы и активности. Расписание офлайн-активностей: как посещать и записываться. Кто такой куратор и как с ним взаимодействовать? Рекомендации по обучению. Ответы на вопросы.

Тема 2. Основы Python

Основы языка программирования Python. Функция print. Переменные. Функция input. Строки.

Практическая часть:

Задание: Написать простую программу, которая запрашивает у пользователя имя и возраст, а затем выводит сообщение с этими данными.

Тема 3. Операторы и выражения

Целые числа. Арифметические операторы Python. Ввод числа. Сокращенные операторы.

Практическая часть:

Задание 1: Создать программу, которая получает на вход четырёхзначное число, а выводит на экран каждую цифру отдельно (все цифры в одну строчку или каждую цифру с новой строки). При этом само число не должно меняться, то есть нельзя использовать переприсваивание. Но можно работать с любым количеством переменных.

Задание 2: Написать программу, которая меняет значения двух переменных местами. Использовать третью переменную и «синтаксический сахар» (конструкция a, b = b, a) нельзя. В переменные будут вводиться только числа.

Тема 4. Условный опретор if, ветвления

Условный оператор. Полная форма условного оператора. Вложенные условия. Цепочки условий.

Практическая часть:

Задание 1: написать программу, которая выводит на экран максимальное из трёх введенных пользователем чисел (все числа разные). Используйте дополнительные переменные, если нужно.

Задание 2: Почтовое отделение открывается в 08:00 и закрывается в 22:00. С 14:00 до 15:00 все сотрудники уходят на обед, а в 10:00 и 18:00 приезжают машины с посылками, и все сотрудники в течение двух часов заняты их разгрузкой. Во время обеда и разгрузки машин посылки никто не выдаёт. Написать программу, которая получает на вход время в часах — число от 0 до 23 — и пишет, можно ли в этот час получить посылку. Использовать только один условный оператор if-else, без elif и прочих.

Тема 5. Цикл while

Оператор while. Прерывание цикла, break. Логический тип данных. Бесконечный цикл. Оператор continue.

Практическая часть:

Задание: вклад в банке составляет X рублей. Ежегодно он увеличивается на P процентов, после чего дробная часть копеек отбрасывается. Определить, через сколько лет вклад составит не менее Y рублей.

Тема 6. Цикл for

Цикл for. Функция range. Итерирование по строке.

Практическая часть:

Задание 1: разработать программу, которая поможет определить, является ли переданный текст, введённый пользователем, палиндромом.

Задание 2: X мальчиков и Y девочек пошли в кинотеатр и купили билеты на идущие подряд места в одном ряду. Напишите программу, которая выдаст, как нужно сесть мальчикам и девочкам, чтобы рядом с каждым мальчиком сидела хотя бы одна девочка, а рядом с каждой девочкой — хотя бы один мальчик. На вход подаются два числа: количество мальчиков X и количество девочек Y. В ответе выведите какую-нибудь строку, в которой будет ровно X символов B, обозначающих мальчиков, и Y символов G, обозначающих девочек, удовлетворяющую условию задачи. Пробелы между символами выводить не нужно. Если рассадить мальчиков и девочек согласно условию задачи невозможно, выведите строку «Нет решения».

Тема 7. Вложенные циклы

Вложенные циклы. Работа с двумя счетчиками. Блок else для цикла.

Практическая часть:

Задание: написать программу, которая считает количество простых чисел в заданной последовательности и выводит ответ на экран.

Тема 8. Числа с плавающей точкой

Числа с плавающей точкой. Ввод вещественного числа. Функция float. Функция round. Модуль math. Особенности работы с вещественными числами. Алгоритмы с заданной точностью.

Практическая часть:

Задание 1: напишите программу, принимающую на вход размер файла обновления в мегабайтах и скорость интернет-соединения в мегабайтах в секунду. Для каждой секунды программа должна рассчитывать и выводить на экран процент скачанного объёма до тех пор, пока скачивание не завершится. В конце программа должна показать, сколько секунд заняло скачивание обновления. Обеспечьте контроль ввода.

Задание 2: напишите программу, принимающую на вход размер файла обновления в мегабайтах и скорость интернет-соединения в мегабайтах в секунду. Для каждой секунды программа должна рассчитывать и выводить на экран процент скачанного объёма до тех пор, пока скачивание не завершится. В конце программа должна показать, сколько секунд заняло скачивание обновления. Обеспечьте контроль ввода.

Тема 9. Функции

Функции и их вызов. Функции с одним параметром. Функции с несколькими параметрами. Вложенный вызов функций. Оператор return.

Практическая часть:

Задание 1: написать функцию, вычисляющую наибольший общий делитель двух чисел.

Задание 2: Известно, что амплитуда качающегося маятника с каждым разом затухает на 8,4% от амплитуды предыдущего колебания. Если качнуть маятник, то, строго говоря, он не остановится никогда, просто амплитуда будет постоянно уменьшаться до тех пор, пока мы не сочтём такой маятник остановившимся. Напишите программу, определяющую, сколько раз качнётся маятник, прежде чем он, по нашему мнению, остановится.

Программа получает на вход начальную амплитуду колебания в сантиметрах и конечную амплитуду колебаний, которая считается остановкой маятника. Обеспечьте контроль ввода.

Тема 10. Система контроля версий git

Git. Основные команды git. Удаленные и локальные репозитории. Создание локального и удаленного репозиториев. Подключение к удаленному репозиторию. Внесение изменений и отправка. Обновление локального репозитория. Резрешение конфликтов. Работа с ветками. Слияние веток. Просмотр изменений. Удаление незакомиченных изменений. Отмена закомиченных изменений. Отмена слияний.

Практическая часть

Задание:

Нужно выполнить ряд действий в Git, используя как локальный, так и удалённый репозитории. В процессе выполнения задания вам предстоит создать репозиторий, работать с ветками, вносить изменения, решать конфликты, а также отменять изменения и слияния.

1. Создание репозитория:
1.1. Создайте локальный Git-репозиторий.
1.2. Инициализируйте его и создайте первый коммит с файлом README.md, в котором опишите краткую суть задания.
2. Подключение к удаленному репозиторию:
2.1. Создайте удалённый репозиторий на GitHub (или другой платформе).
2.2. Подключите локальный репозиторий к удалённому.
2.3. Отправьте начальный коммит в удалённый репозиторий.
3. Работа с ветками:
3.1. Создайте новую ветку feature/experiment.
3.2. Внесите изменения в файл README.md (например, добавьте новую строку).
3.3. Закоммитьте изменения и отправьте их в удалённый репозиторий.
4. Слияние веток:
4.1. Переключитесь на основную ветку (main или master).
4.2. Внесите другие изменения в README.md (добавьте другую строку).
4.3. Выполните слияние ветки feature/experiment с основной веткой.
4.4. Если возникнут конфликты, разрешите их, выбрав нужные изменения.
5. Отмена изменений:
5.1. Отмените последний коммит в основной ветке, чтобы вернуться к состоянию до слияния (используйте git reset или git revert).
5.2. Убедитесь, что изменения в локальном репозитории соответствуют вашим ожиданиям.
6. Удаление незакоммиченных изменений:
6.1. Внесите изменения в любой файл, но не коммитите их.

6.2. Отмените незакоммиченные изменения.

7. Обновление локального репозитория:

7.1. Убедитесь, что локальный репозиторий не синхронизирован с удалённым.

7.2. Получите последние изменения из удаленного репозитория

Тема 11. Списки и методы работы с ними

Списки. Инициализация списков. Индексы и работа с элементами списков. Базовые возможности при работе со списками. Методы insert, remove, index. Методы count и extend. Вложенные списки.

Практическая часть:

Задание 1: Контейнеры на складе лежат в ряд в порядке невозрастания (меньше либо равно) массы в килограммах. На склад привезли ещё один контейнер, который тоже нужно положить на определённое место. Напишите программу, которая получает на вход невозрастающую последовательность натуральных чисел. Они означают массу каждого контейнера в ряду. После этого вводится число X — масса нового контейнера. Программа выводит номер, под которым будет лежать новый контейнер. Если в ряду есть контейнеры с массой, как у нового, то его нужно положить после них.

Задание 2: оследовательность чисел называется симметричной, если она одинаково читается как слева направо, так и справа налево. Например, следующие последовательности являются симметричными:

1 2 3 4 5 4 3 2 1

1 2 1 2 2 1 2 1

Пользователь вводит последовательность из N чисел. Напишите программу, которая определяет, какое минимальное количество и каких чисел нужно добавить в конец этой последовательности, чтобы она стала симметричной.

Тема 12. List comprehensions

List comprehensions. Условия в list comprehensions. Срезы списков и строк.

Практическая часть:

Задание: Юлий Цезарь использовал свой способ шифрования текста. Каждая буква заменялась на следующую по алфавиту через K позиций по кругу. Если взять русский алфавит и K, равное 3, то в слове, которое мы хотим зашифровать, буква А станет буквой Г, Б станет Д и так далее. Пользователь вводит сообщение и значение сдвига. Напишите программу, которая изменит фразу при помощи шифра Цезаря.

Тема 13. Базовые коллекции: строки, словари, множества, кортежи

Форматирование строк: f-strings, format. Методы split и join. Методы startswith, endswith, upper, lower. Словари. Методы словарей. Вложенные словари. Множества. Функция set. Генерация словарей. Кортежи. Функция enumerate. Перебор ключей и значений в словаре. Составные ключи. Функция zip.

Практическая часть:

Задание 1: Пользователь вводит строку. Напишите программу, которая определяет, действительно ли заданная строка — правильный IP-адрес. Обеспечьте контроль ввода, где предусматривается добавление целых чисел от 0 до 255 и точек между ними.

Задание 2: На вход в программу подаётся N заказов. Каждый заказ представляет собой строку вида «Покупатель — название пиццы — количество заказанных пицц». Реализуйте код, который выводит список покупателей и их заказов по алфавиту. Учитывайте, что один человек может заказать одну и ту же пиццу несколько раз.

Задание 3:

Даны три списка.

array_1 = [1, 5, 10, 20, 40, 80, 100]

array_2 = [6, 7, 20, 80, 100]

array_3 = [3, 4, 15, 20, 30, 70, 80, 120]

Нужно выполнить две задачи:

1. найти элементы, которые есть в каждом списке;

2. найти элементы из первого списка, которых нет во втором и третьем списках.

Каждую задачу нужно выполнить двумя способами:

1. без использования множеств;

2. использованием множеств.

Задание 4: Даны строка и кортеж из чисел. Напишите программу, которая создаёт генератор из пар кортежей «символ — число». Затем выведите на экран сам генератор и кортежи.

Тема 14. Рекурсия

Рекурсия. Передача изменяемых и неизменяемых данных в функцию. Именованные аргументы и значения по умолчанию. *args, *kwargs.

Практическая часть

Задание 1: Реализуйте алгоритм быстрой сортировки (её называют сортировкой Хоара)

Задание 2: Напишите свою функцию sum, которая должна быть более гибкой, чем стандартная. Она должна уметь складывать числа: из списка списков, набора параметров.

Тема 15. Работа с файлами

Модуль os. Генерация путей. Метод listdir. Базовые операции с файлами: open, close, read. Метод write. Режимы записи. Перемещение курсора в файле.

Практическое задание:

Задание 1: Есть файл text.txt, который содержит текст. Напишите программу, которая выполняет частотный анализ, определяя долю каждой буквы английского алфавита в общем количестве английских букв в тексте, и выводит результат в файл analysis.txt. Символы, не являющиеся буквами английского алфавита, учитывать не нужно. В файл analysis.txt выводится доля каждой буквы, встречающейся в тексте, с тремя знаками в дробной части. Буквы должны быть отсортированы по убыванию их доли. Буквы с равной долей должны следовать в алфавитном порядке.

Задание 2: Напишите программу, которая получает на вход путь до каталога (в том числе это может быть просто корень диска) и выводит общее количество файлов и подкаталогов в нём. Также выведите на экран размер каталога в килобайтах (1 килобайт = 1024 байт).

Тема 16. Исключения и работа с ошибками

Обработка исключений: операторы try except. Операторы else, finally. Вызов исключений : оператор raise. Контекстный менеджер: оператор with.

Практическое задание

Задание 1:

У вас есть файл с протоколом регистрации пользователей на сайте — registrations.txt. Каждая строка содержит имя, имейл и возраст, разделённые пробелами. Например: Василий test@test.ru 27.

Напишите программу, которая проверяет данные из файла для каждой строки:

— Присутствуют все три поля.
— Поле «Имя» содержит только буквы.
— Поле «Имейл» содержит @ и точку.
— Поле «Возраст» представляет число от 10 до 99.

В результате проверки сформируйте два файла:

— registrations_good.log для правильных данных; записывать строки как есть;
— registrations_bad.log — для ошибочных; записывать строку и вид ошибки.

Для валидации строки данных напишите функцию, которая может выдавать исключения:

— НЕ присутствуют все три поля: IndexError.
— Поле «Имя» содержит НЕ только буквы: NameError.
— Поле «Имейл» НЕ содержит @ и точку: SyntaxError.
— Поле «Возраст» НЕ представляет число от 10 до 99: ValueError.

Задание 2: Реализуйте программу — чат, в котором могут участвовать сразу несколько человек, то есть программу, которая может работать одновременно для нескольких пользователей. При запуске запрашивается имя пользователя. После этого он выбирает одно из действий: посмотреть текущий текст чата, отправить сообщение (затем вводит сообщение). Действия запрашиваются бесконечно.

Тема 17. Объектное-ориентированное программирование

Классы. Методы класса, аргумент self. Конструктор __init__. Подключение классов из модулей. Инкапсуляция и скрытие данных. Геттеры и сеттеры. Наследование. Полиморфизм. Документация: описание методов и классов. Множественное наследование: примеси и абстрактные классы. Класс как контекст менеджер

Практическая часть:

Задание 1: Создайте класс Matrix для работы с матрицами. Реализуйте методы: сложения, вычитания, умножения, транспонирования матрицы.

Задание 2: Создайте программу, которая реализует игру «Крестики-нолики».

Задание 3: Стек — это абстрактный тип данных, представляющий собой список элементов, организованных по принципу LIFO (англ. last in — first out, «последним пришёл — первым вышел»). Напишите класс, который реализует стек и его возможности (достаточно будет добавления и удаления элемента). После этого напишите ещё один класс — «Менеджер задач». В менеджере задач можно выполнить команду «новая задача», в которую передаётся сама задача (str) и её приоритет (int). Сам менеджер работает на основе стека (не наследование). При выводе менеджера в консоль все задачи должны быть отсортированы по следующему приоритету: чем меньше число, тем выше задача.

Задание 4: Создайте: класс Shape, который будет базовым классом для всех фигур и будет хранить пустой метод area, который наследники должны переопределить;

— класс Circle;
— класс Rectangle;
— класс Triangle.

Классы Circle, Rectangle и Triangle наследуют от класса Shape и реализуют метод для вычисления площади фигуры.

Тема 18. Итераторы и генераторы

Итераторы. Реализация итераторов. Генераторы и их реализация. Аннотации типов.

Практическая часть:

Задание 1: В односвязном списке связь — это ссылка только на следующий элемент, то есть в нём можно передвигаться только в сторону конца списка. Узнать адрес предыдущего элемента, опираясь на содержимое текущего узла, невозможно. Реализуйте такую структуру данных без использования стандартных структур Python (list, dict, tuple и прочие) и дополнительных модулей. Для реализации напишите два класса: Node и LinkedList. В Node должна быть логика работы одного узла (хранение данных и указателя). Для структуры реализуйте следующие методы:

— append — добавление элемента в конец списка;
— get — получение элемента по индексу;
— remove — удаление элемента по индексу.

Дополнительно: сделайте так, чтобы по списку можно было итерироваться с помощью цикла.

Задание 2 : Реализуйте функцию-генератор, которая берёт все питоновские файлы в директории и вычисляет количество строк в каждом файле, игнорируя пустые строки и строки комментариев. По итогу функция-генератор должна с помощью yield каждый раз возвращать количество строк в очередном файле.

Тема 19. Декораторы

Функции высшего порядка. Декораторы. Модуль functools. Декораторы setter и property. Декораторы classmethod. Декторатор context manager. Декораторы с аргументами. Декораторы для классов. Декоратор как класс.

Практическая часть:

Задание 1: Вы разрабатываете программу для оптимизации вычислений чисел Фибоначчи. Числа Фибоначчи вычисляются рекурсивной функцией, каждое число равно сумме двух предыдущих чисел. Однако вы заметили, что при больших значениях чисел Фибоначчи вычисления занимают значительное время, так как многие значения вычисляются повторно. Вам поручено создать декоратор, который кэширует результаты вызова функции и позволяет избежать повторных вычислений для одних и тех же аргументов.

Задание 2: Вы разрабатываете программу для кэширования запросов к внешнему API. Часто повторяющиеся запросы занимают много времени, поэтому вы решаете создать класс LRU Cache (Least Recently Used Cache), который будет хранить ограниченное количество запросов и автоматически удалять самые старые при достижении лимита. Это позволит значительно ускорить повторяющиеся запросы, так как данные будут браться из кэша, а не отправляться повторно.

Задача

— Создайте класс LRU Cache, который хранит ограниченное количество объектов и, при превышении лимита, удаляет самые давние (самые старые) использованные элементы.
— Реализуйте методы добавления и извлечения элементов с использованием декораторов property и setter.

Задание 3: Синглтон — это порождающий паттерн проектирования, который гарантирует, что у класса есть только один экземпляр, и предоставляет к этому экземпляру глобальную точку доступа. Реализуйте декоратор singleton, который превращает класс в одноэлементный. При множественной инициализации объекта этого класса будет сохранён только первый инстанс, а все остальные попытки создания будут возвращать первый экземпляр.

Тема 20. Библиотеки для работы с данными

Регулярные варжения. Моудль re и его методы. Основы парсинга. Модуль requests. Модуль itertools.

Практическая часть:

Задание 1: В одной организации перед записью телефонного номера в базу данных его проверяют на соответствие следующим критериям:

— Длина номера — ровно десять знаков.
— Номер начинается с цифры 8 или 9.
— Остальные знаки — только цифры.

На вход в программу подаётся список номеров (можно взять готовый или запросить у пользователя). Реализуйте код, который проверяет каждый номер из списка на соответствие критериям и выводит на экран соответствующие сообщения.

Задание 2: Выберите любую веб страницу. Изучите код этой страницы и реализуйте программу, которая получает список всех подзаголовков сайта (они заключены в теги <h3>).

Тема 21. Подготовка к итоговому проекту

Как работает бот. Создание телеграм-бота. Библиотека aiogram / pytelegramBotAPI. ORM. Библиотека sqlalchemy / peewee

Практическая часть:

Задание:

Создание простого Telegram-бота

— Инициализация проекта: Создайте минимальную структуру проекта с основными файлами, такими как main.py для запуска бота.

— Реализация базовых команд: Напишите обработчики для простых команд, таких как /start, /help, и интегрируйте их в вашего бота.

— Интеграция с внешним API: Подключите внешний API для демонстрации работы с данными, например, получения текущей погоды или новостей.

— Сохранение данных: Настройте базу данных и реализуйте сохранение истории запросов пользователя с использованием ORM.

Тема 22. Описание итогового проекта

Что такое итоговый проект? Когда можно начать работу над итоговым проектом?

Какие варианты итогового проекта существуют?

1. Итоговый проект по бизнес-задаче от реального заказчика

Это предложенные проекты от реальных заказчиков. Крупные компании предлагают реальные задачи, которые потом могут использовать в работе. В финале — презентация перед комиссией: кураторами Skillbox, разработчиками или заказчиками онлайн (Skype, Google Meet).

2. Собственный проект

Вы можете работать над собственным проектом, если он подходит под критерии итогового проекта. Критерии описаны ниже.

Какой проект от заказчика можно выбрать? Какой собственный проект можно выбрать?

Вы можете взять собственный проект. Что нужно учесть:

1. Это может быть ваш собственный бизнес, но не коммерческий заказ на фрилансе. Можете выбрать проект для благотворительной организации, например, в сфере охраны окружающей среды. Также это может быть выдуманный проект или проект для друзей и знакомых.
2. Вы должны получить бизнес-задачу от заказчика или составить ее самостоятельно, главное, чтобы она была направлена на работу с API стороннего сайта.
3. Разработать для себя техническое задание с подробным описанием функциональных, технических требований и интерфейса бота.
4. Сформированное техническое задание оформите в файле Word и согласуйте с куратором.
5. После согласования вы можете приступать к работе.

Каковы этапы и сроки подготовки итогового проекта?

Основные этапы работы

1. Анализ бизнес задачи клиента, анализ существующих решений, разработка технического задания.
2. Подготовка окружения для разработки, выбор необходимых технологий.
3. Поэтапная реализация проекта, с постоянным тестированием и улучшением кода.
4. Подготовка к презентации.

Общие рекомендации

1. Постарайтесь чётко определить цель проекта. Независимо от того, работаете ли вы с реальным заказчиком или разрабатываете собственный проект, важно ясно понимать,

какую проблему вы решаете и для кого. Цель проекта должна быть сформулирована чётко и конкретно.

2. Разрабатывайте техническое задание с максимальной детализацией. Включите все ключевые аспекты проекта, такие как функциональные требования, технические спецификации, описание интерфейсов. Чем подробнее ТЗ, тем легче будет реализовать проект.

3. Выбирайте технологии, которые соответствуют задачам. При выборе инструментов и библиотек, ориентируйтесь на специфику проекта. Убедитесь, что выбранные технологии поддерживают требуемый функционал и могут быть легко интегрированы между собой.

4. Планируйте работу поэтапно. Разбейте проект на небольшие этапы, с чёткими сроками и задачами на каждом этапе. Это поможет отслеживать прогресс и вносить коррективы на ранних стадиях.

5. Старайтесь делать работу соизмеримо своим силам. Не нужно перегружать проект дополнительным функционалом, особенно если он сложен в реализации и сам того не требует. Расширяется проект по необходимости и старайтесь находить более просты способы достижения поставленных задач, например, используйте готовые решения.

6. Постоянно тестируйте и улучшайте код. Регулярное тестирование и рефакторинг кода помогут избежать накопления ошибок и обеспечат высокое качество конечного продукта.

7. Будьте готовы к изменениям и доработкам. Проект может потребовать изменений или добавления новых функций в процессе разработки. Оставляйте место для манёвра и будьте готовы адаптироваться к новым требованиям.

Этап/ Задача	**Срок**	**Что необходимо сделать**	**Результат работы**	**Критерии оценки**
Анализ бизнес-задачи	Три дня	Изучите требования, чтобы понять с чем предстоит работать Разберитесь, чего не хватает в требованиях. Если нужно, запросите у заказчика или куратора недостающую информацию. Проанализируйте существующие решения. Выявите лучшие практики.	Сформированное представление о задачах и целях проекта. Краткое описание требуемого функционала.	Понимание бизнес-задачи, согласование с заказчиком, наличие исследовательской части в ТЗ. Описанный функционал удовлетворяет поставленной задаче и дополняет ее.
Разработка технического задания	Три дня	Оформите ТЗ с описанием функциональных и технических требований. Согласуйте получившийся вариант с куратором.	Подготовленное техническое задание	Полнота технического задания. Соответствие требованиям и непротиворечивость.
Подготовка окружения для разработ	Два дня	Настройте окружение для разработки Установите необходимые библиотеки и инструменты. Организуйте файловую	Готовое окружение для разработки	Работоспособность окружения, возможность запуска проекта, установка необходимых библиотек. Файловая структура организована

ки		структуру проекта.		понятно и эффективна для дальнейшей работы.
Поэтапная реализация проекта	Две-три недели	Реализуйте основные модули бота Интегрируйте работу с API Реализуйте историю запросов	Рабочий прототип бота с основным функционалом	Соответствие реализации ТЗ, стабильность работы, правильная интеграция с API, работа с историей запросов.
Тестирование и улучшение кода	Три-четыре дня	Проведите тестирование бота. Исправьте найденные ошибки. Оптимизируйте код.	Финальная версия проекта с исправленными ошибками и оптимизированным кодом.	Минимум ошибок, отзывчивость интерфейса, чистота и структурированность кода.
Подготовка к презентации	Три-четыре дня	Подготовьте метериал для презентации. Подготовьте текст выступления. Оформите презентацию.	Текст выступления. Презентация	Чёткость, логичность и информативность презентации.
Презентация проекта	—	Представление вашего проекта на десять минут и обратная связь по вашей работе		Соблюдена понятная структура презентации. Указаны имя пользователя, описание основной идеи, главные этапы работы и финальный результат. Продемонстрирован конечный вид проекта. Вы способны презентовать свою идею и аргументировать принятые функциональные и архитектурные решения решения. Слайды презентации оформлены аккуратно и визуально привлекательно. На слайдах нет лишнего.

6. Перечень литературы, рекомендуемый для изучения

1. Лутц М. Python. Том 1. Основы программирования. - СПб.: Символ-Плюс, 2019. - 800 с.: ил. - (Серия «Программирование»)
2. Бейкер Э. Python Programming for Beginners: An Introduction to Python Language and Programming with Hands-On Projects. - Москва: Эксмо, 2020. - 350 с.
4. Свенсон Д. Современный Python: просто о сложном. - СПб.: Питер, 2020. - 224 с.: ил. - (Серия «Библиотека программиста»)
5. Маттисон М. Python в примерах и задачах: учебное пособие. - М.: БХВ-Петербург, 2021. - 384 с.
6. Ван Россум Г. Python: Руководство по использованию языка и библиотек. - СПб.: Питер, 2021. - 672 с.: ил.

7. Смит Л. Python: Искусство программирования: Учеб. пособие. - М.: Изд. дом «Вильямс», 2019. - 432 с.: ил.
8. Зед Шоу Научитесь программировать на Python: Учеб. пособие. - СПб.: Питер, 2020. - 464 с.: ил. - (Серия «Классика программирования»)

7. Материально-технические условия реализации дополнительной общеразвивающей программы

Условия реализации обеспечивают: достижение планируемых результатов освоения Программы в полном объеме; соответствие применяемых форм, средств и методов обучения.
Материально –технические условия реализации дополнительной общеобразовательной программы:
Учебный кабинет 1 этаж помещение 3:
компьютеры - 9 штук (6 аймак / 3 виндоус);
столы - 11 штук;
стулья - 11 штук;
шкафы для книг - 2 штуки;
тумбы под технику - 2 штуки.

Учебный кабинет 1 этаж помещение 2:
шкаф - 1 штука;
столы - 15 штук;
компьютеры - 15 штук (3 айм / 12 виндоус);
стулья - 15 штук;
лампы у мониторов - 15 штук;
экран / телевизор – 1 штука;
доска для записей с рулоном – 1 штука.

Учебный класс 1 этаж помещение 8:
парты - 12 штук;
телевизор - 1 штука;
стулья – 24 штуки;
тумба для техники - 2 штуки
ноутбук преподавателя – 1 штука.

Учебный класс 2 этаж помещение 5:
стол – 2 штуки;
стулья – 4 штуки;
экран – 1 штука;
шкаф для хранения – 1 штука;
пуфик – 2 штуки.

Лекторий 2 этаж помещение 4:
стулья – 50 штук;
стол – 1 штука;
проектор – 1 штука;
шкаф – 1 штука.

Учебный класс 2 этаж помещение 7:
телевизор – 1 штука;
столы – 14 штук;
стулья – 25 штук;

компьютеры (аймак) – 7 штук;
ноутбук преподавателя – 1 штук.

Учебный класс 2 этаж помещение 2:
мини парты – 14 штук;
стулья – 32 штуки;
стол – 1 штука;
шкаф – 1 штука;
тумба для хранения материалов – 2 штуки.

8.Кадровое обеспечение Программы

Реализацию программы обеспечивают квалифицированные педагоги, соответствующие требованиям Приказа Минздравсоцразвития РФ от 26.08.2010 N 761н Об утверждении Единого квалификационного справочника должностей руководителей, специалистов и служащих, раздел «Квалификационные характеристики должностей работников образования».

Педагог дополнительного образования:

Требования к квалификации. Высшее профессиональное образование или среднее профессиональное образование в области, соответствующей профилю кружка, секции, студии, клубного и иного детского объединения без предъявления требований к стажу работы, либо высшее профессиональное образование или среднее профессиональное образование и дополнительное профессиональное образование по направлению "Образование и педагогика" без предъявления требований к стажу работы.

9. Оценочный материал итогового экзамена

Защита проекта.

1. Итоговый проект по бизнес-задаче от реального заказчика

Это предложенные проекты от реальных заказчиков. Крупные компании предлагают реальные задачи, которые потом могут использовать в работе. В финале — презентация перед комиссией: кураторами Skillbox, разработчиками или заказчиками онлайн (Skype, Google Meet).

2. Собственный проект

Вы можете работать над собственным проектом, если он подходит под критерии итогового проекта. Критерии описаны ниже.

Какой проект от заказчика можно выбрать? Какой собственный проект можно выбрать?

Вы можете взять собственный проект. Что нужно учесть:

1. Это может быть ваш собственный бизнес, но не коммерческий заказ на фрилансе. Можете выбрать проект для благотворительной организации, например, в сфере охраны окружающей среды. Также это может быть выдуманный проект или проект для друзей и знакомых.
2. Вы должны получить бизнес-задачу от заказчика или составить ее самостоятельно, главное, чтобы она была направлена на работу с API стороннего сайта.
3. Разработать для себя техническое задание с подробным описанием функциональных, технических требований и интерфейса бота.
4. Сформированное техническое задание оформите в файле Word и согласуйте с куратором.
5. После согласования вы можете приступать к работе.

Каковы этапы и сроки подготовки итогового проекта?

Основные этапы работы

1. Анализ бизнес задачи клиента, анализ существующих решений, разработка технического задания.
2. Подготовка окружения для разработки, выбор необходимых технологий.
3. Поэтапная реализация проекта, с постоянным тестированием и улучшением кода.
4. Подготовка к презентации.

Общие рекомендации

1. Постарайтесь чётко определить цель проекта. Независимо от того, работаете ли вы с реальным заказчиком или разрабатываете собственный проект, важно ясно понимать, какую проблему вы решаете и для кого. Цель проекта должна быть сформулирована чётко и конкретно.
2. Разрабатывайте техническое задание с максимальной детализацией. Включите все ключевые аспекты проекта, такие как функциональные требования, технические спецификации, описание интерфейсов. Чем подробнее ТЗ, тем легче будет реализовать проект.
3. Выбирайте технологии, которые соответствуют задачам. При выборе инструментов и библиотек, ориентируйтесь на специфику проекта. Убедитесь, что выбранные технологии поддерживают требуемый функционал и могут быть легко интегрированы между собой.
4. Планируйте работу поэтапно. Разбейте проект на небольшие этапы, с чёткими сроками и задачами на каждом этапе. Это поможет отслеживать прогресс и вносить коррективы на ранних стадиях.
5. Старайтесь делать работу соизмеримо своим силам. Не нужно перегружать проект дополнительным функционалом, особенно если он сложен в реализации и сам того не требует. Расширяется проект по необходимости и старайтесь находить более просты способы достижения поставленных задач, например, используйте готовые решения.
6. Постоянно тестируйте и улучшайте код. Регулярное тестирование и рефакторинг кода помогут избежать накопления ошибок и обеспечат высокое качество конечного продукта.
7. Будьте готовы к изменениям и доработкам. Проект может потребовать изменений или добавления новых функций в процессе разработки. Оставляйте место для манёвра и будьте готовы адаптироваться к новым требованиям.

Этап/ Задача	**Срок**	**Что необходимо сделать**	**Результат работы**	**Критерии оценки**
Анализ бизнес-задачи	Три дня	Изучите требования, чтобы понять с чем предстоит работать Разберитесь, чего не хватает в требованиях. Если нужно, запросите у заказчика или куратора недостающую информацию. Проанализируйте существующие решения.	Сформированное представление о задачах и целях проекта. Краткое описание требуемого функционала.	Понимание бизнес-задачи, согласование с заказчиком, наличие исследовательской части в ТЗ. Описанный функционал удовлетворяет поставленной задаче и дополняет ее.

		Выявите лучшие практики.		
Разработка технического задания	Три дня	Оформите ТЗ с описанием функциональных и технических требований. Согласуйте получившийся вариант с куратором.	Подготовленное техническое задание	Полнота технического задания. Соответствие требованиям и непротиворечивость.
Подготовка окружения для разработки	Два дня	Настройте окружение для разработки Установите необходимые библиотеки и инструменты. Организуйте файловую структуру проекта.	Готовое окружение для разработки	Работоспособность окружения, возможность запуска проекта, установка необходимых библиотек. Файловая структура организована понятно и эффективна для дальнейшей работы.
Поэтапная реализация проекта	Две-три недели	Реализуйте основные модули бота Интегрируйте работу с API Реализуйте историю запросов	Рабочий прототип бота с основным функционалом	Соответствие реализации ТЗ, стабильность работы, правильная интеграция с API, работа с историей запросов.
Тестирование и улучшение кода	Три-четыре дня	Проведите тестирование бота. Исправьте найденные ошибки. Оптимизируйте код.	Финальная версия проекта с исправленными ошибками и оптимизированным кодом.	Минимум ошибок, отзывчивость интерфейса, чистота и структурированность кода.
Подготовка к презентации	Три-четыре дня	Подготовьте метериал для презентации. Подготовьте текст выступления. Оформите презентацию.	Текст выступления. Презентация	Чёткость, логичность и информативность презентации.
Презентация проекта	—	Представление вашего проекта на десять минут и обратная связь по вашей работе		Соблюдена понятная структура презентации. Указаны имя пользователя, описание основной идеи, главные этапы работы и финальный результат. Продемонстрирован конечный вид проекта. Вы способны презентовать свою идею и аргументировать принятые функциональные и архитектурные решения решения. Слайды презентации оформлены аккуратно и визуально привлекательно. На слайдах нет лишнего.