

Комплект практических контрольных заданий с ответами

Задания комплекта разработаны нами для того, чтобы ваши ученики могли самостоятельно выполнить практическую задачу написания кода по пройденной теме, а у вас была возможность с их помощью сформировать обратную связь, которая может стать мощным инструментом в улучшении образовательного опыта.

В каждой теме п.1 – условие контрольного задания по практике, п.2 – материал для учителя: возможный вариант решение и рекомендации (опционально).

Т.1. Язык программирования Python, алгоритм и виды алгоритмов

1	<p>Проект «Маршрут».</p> 	<p>Напишите алгоритм для маршрута от пункта В до пункта D. Робот-водитель имеет следующий набор команд: поворот направо, поворот налево, движение вперед, движение назад, стоп-перекресток. Известно, что рекомендованный маршрут содержит два поворота направо.</p>
2	<p>Алгоритм «Маршрут»</p> <ul style="list-style-type: none">· Движение вперед· Стоп-перекресток· Поворот направо· Движение вперед· Стоп-перекресток· Поворот налево· Движение вперед· Стоп-перекресток· Поворот направо· Движение вперед <p><i>Обсудите с учащимися, что помимо рекомендованного маршрута существуют и другие варианты маршрутов. Повторите свойства алгоритма, обратите особое внимание на эффективность.</i></p>	

Т.2. Переменная и оператор присваивания. Типы переменных и операции с ними.

1	<p>Проект «Хохлатые хохотушки». В коде программы задайте исходные текстовые переменные со значениями: «хо», «хлатые», «тушки», «том», «тали». Используя эти переменные с функциями сложения и умножения, составьте фразу «хохлатые хохотушки хохотом хохотали». Не забудьте про пробелы! Фразу запишите в переменную <code>skoro text</code>.</p>
---	---

	После завершения работы над кодом запустите программу и выведите в терминале значение переменной skoro_text .
2	<pre> t1 = "хо" t2 = "хлатые" t3 = "тушки" t4 = "том" t5 = "тали" skoro_text = t1 + t2 + ' ' + t1 * 2 + t3 + ' ' + t1 * 2 + t4 + ' ' + t1 * 2 + t5 </pre>

Т.3. Ввод и вывод информации (print, input).

1	<p>Проект «Генератор объявлений». Запросите у пользователя его имя и адрес (название улицы, номер дома, номер квартиры), а также попросите ввести произвольное целое неотрицательное число.</p> <p>Выведите «объявление» следующего содержания: «Внимание! <i>Имя</i>, проживающий(ая) по адресу: улица <i>название</i>, дом <i>номер</i>, квартира <i>номер</i>! В этом году вам повезет <i>количество</i> раз(а)!»</p>
2	<pre> name = input("Введи имя: ") street = input("Введи название улицы: ") house = input("Введи номер дома: ") apart = input("Введи номер квартиры: ") luck = input("Введи целое неотрицательное число: ") print("Внимание!") print(name + ", проживающий(ая) по адресу: улица " + street + ", дом " + house + ", квартира " + apart + ".") print("В этом году вам повезет " + luck + " раз(а).") </pre>

Т.4. Первый чат-бот.

1	<p>Проект «Чат-бот для бюджетного планирования». Выведите приветственное сообщение. Запросите у пользователя размер его финансовых поступлений за месяц (доход) и суммы обязательных расходов (еда, проезд, интернет, мобильная связь), а также сколько нужно отложить в личный «стабилизационный фонд»; запросите количество дней в текущем месяце. Выведите бюджетный план: остаток после обязательных платежей и дневной бюджет (ежедневный допустимый расход).</p>
2	<pre> print ("Привет! Я чат-бот для бюджетного планирования.") dohod = float(input("Введите ваш ежемесячный доход: ")) eda = float(input("Сколько вы планируете потратить на еду: ")) proezd = float(input("Сколько вы потратите на проезд: ")) internet = float(input("Сколько заплатите за интернет: ")) mobil = float(input("Сколько заплатите за мобильную связь: ")) fond = float(input("Сколько денег отложите: ")) days = int(input("Сколько дней в текущем месяце: ")) ostatok = dohod - eda - proezd - internet - mobil - fond print("Ваш остаток равен:", ostatok) print("Дневной бюджет:", ostatok / days) </pre>

Т.5. Операции с целыми числами: целочисленное деление, остаток от деления. Кратность чисел.

1	Проект « Зеркальное число ». Запросите целое четырехзначное число и выведите «зеркальное» ему число (цифры стоят в обратном порядке), используя целочисленное деление и остаток от деления.
2	<pre> a = int(input("Введи целое четырехзначное число: ")) n1 = a // 1000 n2 = (a // 100) % 10 n3 = (a % 100) // 10 n4 = a % 10 n = n4 * 1000 + n3 * 100 + n2 * 10 + n1 print("Искомое число будет:", n) </pre>

Т.6. Алгоритмы для решения математических задач без деления.

1	Проект « Потребительская корзина ». В потребительской корзине (набор продуктов на год) приведены нормы картофеля и рыбы (соответственно) для трудоспособного населения – 100,4 и 18,5 кг; для пенсионеров – 80 и 16 кг; для детей – 88,1 и 18,6 кг. Запросите цену на эти продукты. Вычислите сколько потратит семья на эти продукты за год, предварительно узнав состав семьи. Выведите результат на экран.
2	<pre> price_1 = float(input("Введи цену за килограмм картошки: ")) price_2 = float(input("Введи цену за килограмм рыбы: ")) n1 = int(input("Введи количество работающих членов семьи: ")) n2 = int(input("Введи количество пенсионеров в семье: ")) n3 = int(input("Введи количество детей в семье: ")) norm1_potatoes = 100.4 norm1_fish = 18.5 norm2_potatoes = 80 norm2_fish = 16 norm3_potatoes = 88.1 norm3_fish = 18.6 summa_potatoes = (norm1_potatoes * n1 + norm2_potatoes * n2 + norm3_potatoes * n3) * price_1 summa_fish = (norm1_fish * n1 + norm2_fish * n2 + norm3_fish * n3) * price_2 print("Общие расходы семьи на эти продукты за год:", summa_potatoes + summa_fish, "руб.") </pre>

Т.7. Алгоритмы для решения математических задач с возведением в степень.

1	<p>Проект «Координаты точек». Дана функция $y = x^n$, где n – целое число. Запросите у пользователя значение n. Затем запросите у пользователя значения абсцисс точек A, B, C, D, E для построения графика этой функции. Вычислите ординаты и выведите координаты этих точек в формате $A(x_a; y_a)$ и т. д.</p>
2	<pre> n = int(input("Введи значение n: ")) x_a = int(input("Введи абсциссу точки A: ")) x_b = int(input("Введи абсциссу точки B: ")) x_c = int(input("Введи абсциссу точки C: ")) x_d = int(input("Введи абсциссу точки D: ")) x_e = int(input("Введи абсциссу точки E: ")) y_a = x_a ** n y_b = x_b ** n y_c = x_c ** n y_d = x_d ** n y_e = x_e ** n print("Точка A: (" , x_a, "," , y_a, ")") print("Точка B: (" , x_b, "," , y_b, ")") print("Точка C: (" , x_c, "," , y_c, ")") print("Точка D: (" , x_d, "," , y_d, ")") print("Точка E: (" , x_e, "," , y_e, ")") </pre>

Т.8. Операторы условия if и else.

1	<p>Проект «Графики линейных функций». Запросите у пользователя коэффициенты a, b, c и d. Исследуйте прямые, заданные линейными функциями $y_1 = ax + c$ и $y_2 = bx + d$. Напишите программу, которая определяет:</p> <ul style="list-style-type: none"> - угол (острый или тупой) между прямой y_1 и положительным направлением оси абсцисс; - проходит ли через начало координат прямая y_2; - пересекаются ли эти две прямые.
2	<pre> a = float(input("Введи коэффициент a: ")) b = float(input("Введи коэффициент b: ")) c = float(input("Введи коэффициент c: ")) d = float(input("Введи коэффициент d: ")) if a > 0: print("между прямой y1 и положительным направлением оси абсцисс угол острый") else: print("между прямой y1 и положительным направлением оси абсцисс угол тупой ") if d != 0: print("прямая y2 не проходит через начало координат") else: print("прямая y2 проходит через начало координат") </pre>

	<pre> if a != b: print("прямые пересекаются") else: print("прямые параллельны или совпадают") </pre>
--	--

Т.9. Сложные условия с помощью **elif**

1	<p>Проект «Скидка постоянного покупателя». В одном известном онлайн-магазине существует индивидуальная система скидок для покупателей, при расчете процента скидки учитывается сумма выкупа за 2 года. Лестница скидок: от 25 тысяч рублей – 5 %, от 50 тысяч рублей – 7 %, свыше 75 тысяч рублей – 10 %.</p> <p>Напишите программу, которая на основе суммы выкупа и цены товара рассчитывает скидку на этот товар.</p>
2	<pre> summa = float(input("Введите сумму выкупа за 2 года: ")) price = float(input("Введите стоимость товара: ")) if summa < 25000: percent = 0 elif summa < 50000: percent = 0.05 elif summa < 75000: percent = 0.07 else: percent = 0.10 discount = price * percent print("Ваша скидка составит:", discount, "руб.") </pre>

Т.10. Вложенное условие. Решение квадратного уравнения

1	<p>Проект «Стоимость посылки». Стоимость доставки посылки зависит от ее веса и расстояния. Условия для отправки посылки:</p> <ul style="list-style-type: none"> ▪ Вес посылки: до 5 кг. Региональная доставка (в пределах одного региона) – 100 рублей, федеральная (доставка по всей стране) – 300 рублей. ▪ Вес посылки: от 5 кг до 10 кг. Региональная доставка – 150 руб., федеральная доставка – 500 руб. ▪ Вес посылки: от 10 кг. Региональная доставка – 200 руб., федеральная доставка – 800 руб. ▪ Напишите программу, которая запрашивает вес посылки и тип доставки («рег» или «фед») и рассчитывает стоимость доставки посылки.
2	<pre> weight = float(input("Введи вес посылки: ")) region = input("Введи тип доставки (рег/фед): ") if region == "per": if weight < 5: cost = 100 elif weight < 10: cost = 150 else: cost = 200 elif region == "фед": </pre>

	<pre> if weight < 5: cost = 300 elif weight < 10: cost = 500 else: cost = 800 print("Стоимость доставки:", cost, "рублей") </pre>
--	--

Т.11. Составные условия, логические операторы and, or, not

1	<p>Проект «Калькулятор дисконтной скидки». Попросите пользователя ввести сумму покупки и количество товаров. Если сумма превышает 1000, а количество товаров больше 5, или сумма превышает 3000, а количество товаров больше 3, то предоставьте дополнительную скидку 10 %.</p>
2	<pre> summa = float(input("Введи сумму покупки: ")) count = int(input("Введи количество товаров: ")) if (summa > 1000 and count > 5) or (summa > 3000 and count > 3): summa = summa * 0.9 print("Сумма к оплате:", summa) </pre>

Т.12. Цикл с условием. Алгоритм Евклида. Разбиение записи натурального числа на цифры

1	<p>Проект «Кофейня». Вы владелец уютной кофейни, где каждый день пекут различные виды выпечки в неограниченном количестве. Вы решили предоставлять скидки в зависимости от времени дня. После 16:00 вы предлагаете скидки в зависимости от выбранного товара: на пирожные – 30 % скидка, на круассаны – 10 % скидка, а булочки остаются без изменений. Режим работы кофейни с 9:00 до 21:00. Напишите программу, которая запрашивает текущее время у пользователя и предоставляет возможность выбора товара с учетом указанных условий. Программа завершает работу, если введено время, когда кофейня закрыта.</p>
2	<pre> user_time = int(input("Введите текущее время (от 0 до 23): ")) while 9 <= user_time and user_time < 21: discount = -1 product = input("Введите название товара (пирожное, круассан, булочка): ").lower() if product == "пирожное": if 16 <= user_time: discount = 30 else: discount = 0 elif product == "круассан": if 16 <= user_time: discount = 20 else: discount = 0 elif product == "булочка": </pre>

	<pre> discount = 0 else: print("Такого товара нет.") if discount != -1: print("Будет применена скидка (%):" , discount) user_time = int(input("Введите текущее время (от 0 до 23): ")) print("Кофейня закрыта.") </pre>
--	--

Т.13. Символьные (строковые) переменные. Встроенные функции для обработки строк

1	<p>Проект «Анализ фразы». Напишите программу, которая позволяет пользователю ввести любую фразу для анализа. Если эта фраза длиннее, чем 20 символов или содержит букву «я», то программа выводит все буквы фразы заглавными; если эта фраза короче, чем 10 символов или содержит букву «ю», то программа выводит все буквы фразы строчными. Приоритет на вывод фразы заглавными буквами. В остальных случаях программа заменяет фразу словом «финиш». Для поиска используйте метод find и осуществляйте проверку буквы в любом регистре.</p>
2	<pre> phrase = input("Введи фразу: ") if len(phrase) > 20 or phrase.find("Я") != -1 or phrase.find("я") != -1: result = phrase.upper() elif len(phrase) < 10 or phrase.find("Ю") != -1 or phrase.find("ю") != -1: result = phrase.lower() else: result = "финиш" print("Результат такой: ", result) </pre>

Т.14. Работа со срезами, получение элемента строки или подстроки

1	<p>Проект «Модификатор фраз». Напишите программу, которая принимает на вход фразу и два целых числа. Если сумма этих чисел меньше длины фразы, то программа обрезает фразу с начала и конца на число символов, равных введенным числам; затем удаляет все пробелы и выводит результат заглавными буквами. Если сумма больше или равна длине фразы, то программа выводит фразу «модификация не получится».</p>
2	<pre> phrase = input("Введи фразу: ") num1 = int(input("Введи первое целое число: ")) num2 = int(input("Введи второе целое число: ")) summa = num1 + num2 if summa < len(phrase): modified = phrase[num1:-num2].replace(" ", "").upper() print("Результат модификации:", modified) else: print("Модификация не получится") </pre>

T.15. Округление чисел: функция round и модуль math

1	Проект « Разное округление ». Составьте программу, которая запрашивает у пользователя число. Если число больше 1, то программа округляет его до целого в большую сторону, если число меньше -1, то округляет его до целого в меньшую сторону. Числа в промежутке от -1 до 1 округляет с точностью до второго знака.
2	<pre> from math import * num = float(input("Введи число: ")) if num > 1: rounded = ceil(num) elif num < -1: rounded = floor(num) else: rounded = round(num, 2) print("Результат округления:", rounded) </pre>

T.16. Циклы с ограниченным количеством повторений. Подсчет частоты появления символа. Посимвольная обработка строк.

1	Проект « Подсчет повторов первого символа и аббревиатура ». Составьте программу, которая запрашивает у пользователя фразу из нескольких слов. Программа составляет аббревиатуру фразы из первых букв каждого слова (заглавными буквами), а также считает количество букв, совпадающих с первой буквой фразы (без учета регистра и не считая первой буквы).
2	<pre> phrase = input("Введите фразу из нескольких слов: ") abbreviation = "" letter = phrase[0].lower() count = 0 for char in phrase: if char.lower() == letter: count = count + 1 for i in range(len(phrase)): if i == 0 or phrase[i - 1] == ' ': abbreviation = abbreviation + phrase[i] print("Аббревиатура:", abbreviation.upper()) print("Количество повторов первой буквы:", count - 1) </pre>

T.17. Проверка делимости и другие алгоритмы для решения математических задач с циклами. Проверка числа на простоту.

1	Проект « Магические числа ». Напишите программу, которая определяет, является ли заданное число «магическим». Магическим считается число, у которого сумма цифр равна произведению цифр.
2	<pre> number = int(input("Введи число: ")) </pre>

	<pre> digit_sum = 0 digit_product = 1 while number > 0: digit = number % 10 number = number // 10 digit_sum = digit_sum + digit digit_product = digit_product * digit if digit_sum == digit_product: print("Это магическое число!") else: print("Это не магическое число.") </pre>
--	---

T.18. Циклы с неограниченным количеством повторений.

1	<p>Проект «Список покупок». Перед походом в магазин полезно составить список покупок. Напишите программу, которая будет запрашивать у пользователя, что ему нужно купить. Чтобы закончить ввод покупок, используйте ключевое слово «стоп». Если количество покупок достигнет 20, остановите программу, сообщив, что место закончилось. В конце программы, если список покупок не пустой, выведи его на экран.</p>
2	<pre> product = "" product_list = "" num = 0 while True: product = input("Что нужно купить? ") if product == "стоп": break num = num + 1 product_list = product_list + ", " + product if num == 20: print("Место закончилось") break if product_list != "": print("Твой список покупок: ", product_list[2:]) </pre>

T.19. Циклы с неограниченным количеством повторений для решения математических задач.

1	<p>Проект «Стоимость автомобиля». Каждый год стоимость автомобиля уменьшается на 15 %. Напишите программу, в которой пользователь вводит стоимость автомобиля на сегодняшний день и до какой суммы должна уменьшиться эта стоимость.</p>
---	---

	<p>Программа определяет, через сколько лет стоимость автомобиля упадет до указанного значения. Учтите, что через 10 лет необходима экспертиза, в результате которой процент уменьшения стоимости может измениться, а значит подсчет станет невозможен. Не забудьте проверить корректность ввода данных.</p>
2	<pre> cur_cost = int(input("Введи стоимость автомобиля на сегодняшний день: ")) min_cost = int(input("Введи, до какой суммы должна уменьшиться стоимость автомобиля: ")) if cur_cost <= 0 or min_cost <= 0 or cur_cost <= min_cost: print("Некорректный ввод данных. Программа завершена.") else: cost = cur_cost years = 0 while True: cost = cost * 0.85 years = years + 1 if years >= 10: print("Потребуется не меньше 10 лет, нужно провести экспертизу") break if cost <= min_cost: print("Через", years, "лет стоимость автомобиля упадет до заданного значения") break </pre>

Т.20. Рисование символами: строка символов, основные операции и функции.

1	<p>Проект «Шапка для страницы дневника». Нужно написать программу для оформления страницы дневника. На верхней строке страницы дневника должны быть выделены места для внесения месяца и года, за каждым из них следует предусмотреть линию (подчеркивание), где ученик сможет вписать соответствующую информацию. На второй строке разместите строку таблицы с четырьмя колонками: «Предмет», «Домашнее задание», «Оценка», «Подпись учителя» (учтите их ширину). Используя команду печати надписи и символы подчеркивания, вертикальной черты, а также пробелов, создайте структуру шапки для страницы дневника.</p>
2	<pre> # Вывод строки для ввода месяца и подчеркивания print("Месяц: " + 15 * " _ " + " Год: " + 11 * " _ ") print("-" * 100) # Вывод строки таблицы с четырьмя колонками print(" Предмет" + 13 * " " + " Домашнее задание" + 20 * " " + " Оценка" + 5 * " " + " Подпись учителя " + 7 * " " + " ") # Вывод разделительной строки таблицы print("-" * 100) </pre>

Т.21. Рисование символами: вывод строк с помощью цикла с ограниченным количеством повторений.

1	<p>Проект «Гирлянда». Напишите программу, которая рисует гирлянду с использованием трех символов. Группа этих символов представляет собой композицию, которая повторяется. «Провод» между группами символов представлен тремя тире. Пользователь вводит символы для гирлянды и длину гирлянды (число, кратное 3). Результат выводится в одну строку.</p> <p>Пример Введите первый символ: * Введите второй символ: o Введите третий символ: # Введите длину гирлянды: 9 ---*o#---*o#---*o#---</p>
2	<pre># Запрашиваем у пользователя символы для гирлянды sym_1 = input("Введи первый символ: ") sym_2 = input("Введи второй символ: ") sym_3 = input("Введи третий символ: ") tire = '-' * 3 # Провод между группами символов # Задаем длину гирлянды length = int(input("Введите длину гирлянды (число, кратное 3): ")) # Вычисляем количество повторений композиции repeats = length // 3 # Рисуем гирлянду for i in range(repeats): print(tire + sym_1 + sym_2 + sym_3, end="") # Добавляем провод справа print(tire)</pre>

Т.22. Рисование символами: вывод строк с помощью цикла с неограниченным количеством повторений. Создание эффекта анимации.

1	<p>Проект «Речная рябь». Напишите программу для создания движущейся оптической иллюзии «Речная рябь». Для создания используйте: символы «~» и пробел, операцию умножения, «восходящий» и «нисходящий» конечные циклы, бесконечный цикл, модуль time и команду sleep.</p>
2	<pre>from time import * while True: for i in range(20): print((" " * i + " ~ ~ ~ ") * 5) sleep(0.1) for i in range(20, 0, -1): print((" " * i + " ~ ~ ~ ") * 5) sleep(0.1)</pre>

Т.23. Рандомизация выбора числа.

1	Проект « Звездное небо ». Напишите программу, которая нарисует «звездное небо», причем звезды должны располагаться случайно. Для этого выведите 10 строк. В каждой строке отобразите случайное количество звезд: от 1 до 5. Расстояние между звездами в одной строке одинаково, но его тоже нужно выбрать случайным образом, причем так, чтобы длина строки не превышала 30 символов.
2	<pre>from random import * for i in range(10): count_stars = randint(1, 5) max_dist = (30 - count_stars) // count_stars dist = randint(1, max_dist) star = " " * dist + "*" print(star * count_stars)</pre>

Т.24. Игра «Угадай число».

1	<p>Проект «Секретная дверь». Напишите программу для увлекательной игры по поиску секретной двери. Пользователю нужно выбрать правильное направление (север, юг, восток, запад) и место (лес, пустыня, горы), чтобы найти дверь. Он должен угадать комбинацию направления и места. Программа позволяет играть до тех пор, пока игрок не отыщет дверь или не решит завершить игру, ответив «нет» на вопрос программы о продолжении.</p> <p>Примечание Для удобства работы с текстовыми значениями направлений и мест, используйте команду choice из модуля random и команду split. Например, random.choice("север юг восток запад".split()) поможет вам случайным образом выбрать одно из направлений.</p>
2	<pre>from random import * print("Добро пожаловать в игру!") directions = "север юг восток запад" places = "лес пустыня горы" while True: direction = choice("север юг восток запад".split()) place = choice("лес пустыня горы".split()) user_direction = input("Выбери направление (север/юг/восток/запад): ").lower() user_place = input("Выбери место (лес/пустыня/горы): ").lower() if user_direction == direction and user_place == place: print("Поздравляю! Ты нашел секретную дверь!") break else: print("Увы, ты выбрал неверное направление или место. Попробуй ещё раз.") answer = input("Продолжить игру? (да/нет): ").lower()</pre>

	<pre> if answer != "да": print("До свидания! Игра завершена.") break </pre>
--	---

Т.25. Создание списка, сортировка, действия над элементами, циклический просмотр списка».

1	<p>Проект «Любимые фильмы». Лиза и Артем дружат очень давно, однако их мнения не всегда полностью совпадают. Напишите программу, которая позволяет каждому из них ввести названия 5 своих любимых фильмов. После ввода программа определяет и выводит на экран фильмы, которые нравятся и Лизе, и Артему. Если таких фильмов нет, выводится соответствующее сообщение.</p> <p><i>Примечание.</i> Для того, чтобы определить, содержится ли элемент в списке, используйте условие: <i>элемент in список</i>. Например,</p> <pre> if el in my_list: print("Элемент есть в списке") </pre> <p>Этот код выведет надпись, если элемент el есть в списке my_list.</p>
2	<pre> # Ввод списка любимых фильмов Лизы print("Лиза, введи названия пяти любимых фильмов:") movies_l = [] for i in range(5): movie = input(str(i+1) + ". ") movies_l.append(movie) # Ввод списка любимых фильмов Артема print("Артем, введи названия пяти любимых фильмов:") movies_a = [] for i in range(5): movie = input(str(i+1) + ". ") movies_a.append(movie) # Поиск общих любимых фильмов common_movies = [] for movie in movies_l: if movie in movies_a: common_movies.append(movie) # Вывод общих любимых фильмов if len(common_movies) > 0: print("Общие любимые фильмы у Лизы и Артема:") for movie in common_movies: print(movie) else: print("У Лизы и Артема нет общих любимых фильмов.") </pre>

Т.26. Работа со списками, выбор случайного элемента. Игра «Предсказание».

1	<p>Проект «Случайное меню». Для популяризации новых блюд, ресторан проводит акцию «Случайное меню». Суть акции заключается в том, что каждый посетитель может получить завтрак из «выбранных случаев» блюд и получить скидку в 10 % от общей стоимости. Напишите программу, которая запрашивает у хозяина ресторана названия блюд для участия в акции: три варианта для горячих блюд, три варианта для напитков и три варианта для десертов. В ходе акции программа предлагает каждому посетителю дать согласие на участие в акции. Если посетитель соглашается, программа случайным образом выбирает горячее блюдо, напиток и десерт из списка акции, а затем выводит на экран «случайный» выбор. В случае отказа от участия, программа выводит пожелание "Всего хорошего". Учтите также случай некорректного ввода ответа на вопрос об участии в акции. Акция завершается в случае, если получен отрицательный ответ на запрос для ее продолжения (цикл «обслуживания посетителей» прерывается).</p>
2	<pre> from random import * # Запрос у хозяина ресторана списка блюд print("Формирование списка блюд для акции:") hot_dish = [] drink = [] dessert = [] for i in range(3): dish_item = input("Введите горячее блюдо №" + str(i+1) + ": ") hot_dish.append(dish_item) drink_item = input("Введите напиток №" + str(i+1) + ": ") drink.append(drink_item) dessert_item = input("Введите десерт №" + str(i+1) + ": ") dessert.append(dessert_item) # Бесконечный цикл для многократного использования программы while True: # Запрос о продолжении акции start_action = input("Запустить акцию 'Случайное меню'? (да/нет): ") if start_action.lower() == "нет": print("Акция остановлена.") break # Участие в акции consent = input("Хотите ли вы участвовать в акции 'Случайное меню'? (да/нет): ") if consent.lower() == "нет": print("Всего хорошего.") elif consent.lower() == "да": print("Горячее блюдо:", choice(hot_dish)) print("Напиток:", choice(drink)) print("Десерт:", choice(dessert)) else: print("Некорректный ввод. Пожалуйста, введите 'да' или 'нет'.") </pre>

1	<p>Проект «Выбор микрофона». Напишите программу для интернет-магазина. Программа позволяет покупателям сортировать товар по критериям: цена, рейтинг и индекс популярности.</p> <p>Задайте заранее информацию о пяти различных микрофонах, представленную в виде сложного списка с их характеристиками. Программа должна предоставлять пользователю критерии сортировки:</p> <ul style="list-style-type: none"> • по популярности – чем выше индекс популярности (от 0 до 100), тем товар более популярен; • по рейтингу – чем выше рейтинг (от 0 до 5), тем качественнее товар; • по возрастанию цены – чем выше цена (от 0 до «бесконечности»), тем дороже товар. <p>Программа должна предоставлять пользователю возможность выбора критерия сортировки и выводить отсортированный список микрофонов в соответствии с нужным критерием.</p>
2	<pre> microphones = [["Микрофон 1", 100, 4.5, 80], ["Микрофон 2", 150, 4.2, 90], ["Микрофон 3", 120, 4.7, 70], ["Микрофон 4", 200, 4.8, 85], ["Микрофон 5", 130, 4.3, 75]] print("Выберите критерий сортировки:") print("1. По возрастанию цены") print("2. По убыванию рейтинга") print("3. По убыванию популярности") option = int(input("Введите номер: ")) if option == 1: microphones.sort(key=lambda x: x[1]) print("Результат сортировки по возрастанию цены:") elif option == 2: microphones.sort(key=lambda x: x[2]) microphones.reverse() print("Результат сортировки по убыванию рейтинга:") elif option == 3: microphones.sort(key=lambda x: x[3]) microphones.reverse() print("Результат сортировки по убыванию популярности:") for mic in microphones: print(mic[0] + ": цена - " + str(mic[1]) + ", рейтинг - " + str(mic[2]) + ",", популярность - " + str(mic[3])) </pre>

Т.28. Работа со списками, перемешивание элементов. Игра «Угадай столицу».

1	<p>Проект «Секретный Санта». Представьте, что вы организуете подарочный обмен среди друзей или коллег. Каждый участник должен дарить подарок другому участнику. Программа «Секретный Санта» поможет автоматизировать этот процесс.</p> <p>Организатор вводит имена участников подарочного обмена. Затем программа формирует случайные пары участников и создает сложный список для хранения</p>
---	---

	<p>этих пар. Наконец, программа выводит на экран для организатора список сформированных пар участников подарочного обмена.</p> <p>Программа должна отслеживать, чтобы в пару не попали одинаковые имена.</p> <p>Примечание Для удаления элемента из списка используйте метод remove.</p> <pre>my_list.remove(el)</pre> <p>Такой код удалит элемент el из списка my_list.</p>
2	<pre>from random import * # Ввод списка участников подарочного обмена members = [] while True: member = input("Введите имя участника подарочного обмена (для завершения введите 'готово'): ") if member.lower() == "готово": break members.append(member) # Создаем копию исходного списка members_copy = members.copy() # Формирование пар участников для подарочного обмена pairs = [] for giver in members: # Случайный выбор получателя подарка (не равного дарителю) receiver = choice(members_copy) while giver == receiver: receiver = choice(members_copy) # Формирование пары и добавление ее в список пар pair = [giver, receiver] pairs.append(pair) # Удаление выбранного получателя из копии списка участников members_copy.remove(receiver) # Вывод пар участников print("Пары участников для подарочного обмена:") for pair in pairs: print(pair[0], "дарит подарок", pair[1])</pre>

Т.29. Множества и методы работы с ними.

1	<p>Проект «Фильтрация сообщений в чате». Напишите программу для фильтраций сообщений в чате. Программа должна использовать множество для хранения запрещенных слов и проверять каждое сообщение на их наличие.</p> <p>После получения сообщения от пользователя для отправки в чат, программа автоматически разбивает его на отдельные слова и осуществляет проверку наличия запрещенных слов.</p> <p>Если запрещенное слово обнаружено, программа сообщает об этом, если не обнаружено – разрешает отправку в чат. Далее программа предлагает ввести новое сообщение или завершить общение.</p>
---	--

	<p>Примечание Разбить сообщение на слова, используя в качестве разделителя пробел, поможет команда split().</p> <pre>text = "сделал дело – гуляй смело" words = text.split() print(words)</pre> <p>Такой код выведет на экран список ['сделал', 'дело', '-', 'гуляй', 'смело'].</p>
2	<pre>banned_words = {"балбес", "балда", "болван", "гад", "недоумок"} while True: message = input("Введите ваше сообщение для отправки в чат: ") words = set(message.lower().split()) if words & banned_words != set(): print("Ваше сообщение содержит запрещенные слова. Пожалуйста, исправьте его.") else: print("Ваше сообщение отправлено в чат.") choice = input("Хотите отправить еще одно сообщение? (да/нет): ").lower() if choice != "да": break</pre>

Т.30. Игра «Быки и коровы». Формирование данных.

1	<p>Проект «Любимые фильмы и книги». У двух подруг, Даши и Маши, есть списки их любимых фильмов и книг. Напишите программу, которая проводит анализ их предпочтений и определяет:</p> <ul style="list-style-type: none"> ■ количество фильмов, которые не являются любимыми сразу у обеих подруг; ■ список книг, которые предпочитает только одна из подруг.
2	<pre>dasha_movies = {"Межсезонье", "Ника", "Русалка", "Светлячок"} masha_movies = {"Королева", "Ника", "Эбигейл", "Выпускной", "Русалка"} dasha_books = {"Дом, в котором..", "Чудо", "Янка", "Винноваты звезды"} masha_books = {"Первокурсница", "Чудо", "Янка", "Пик", "Первая работа"} common_movies = dasha_movies & masha_movies not_common_movies_count = len(dasha_movies) + len(masha_movies) - 2 * len(common_movies) unique_books_dasha = dasha_books - masha_books unique_books_masha = masha_books - dasha_books print("Количество фильмов, не являющихся любимыми у обеих подруг:", not_common_movies_count) print("Список книг, которые предпочитает только Даша:", unique_books_dasha) print("Список книг, которые предпочитает только Маша:", unique_books_masha)</pre>

Т.31. Игра «Быки и коровы». Обработка данных.

1	<p>Проект «Личностные характеристики». Представьте, что вы психолог и хотите исследовать, какие личностные характеристики пересекаются у друзей. У каждого из трех друзей есть свой набор личностных характеристик, представленных в виде множества. Необходимо написать программу, которая определяет общие</p>
---	--

	<p>характеристики между друзьями и выводит их на экран. Найдите общие характеристики между другом 1 и другом 2, а также между другом 1 и другом 3. Пример вывода программы:</p> <p>Общие характеристики для 1 и 2: {'эмпатия'}</p> <p>Общие характеристики для 1 и 3: {'интеллект', 'экстраверсия'}</p>
2	<pre> friend1_characteristics = { "эмпатия", "ответственность", "интеллект", "тактичность", "экстраверсия" } friend2_characteristics = { "амбициозность", "терпимость к риску", "самоконтроль", "творческий потенциал", "эмпатия" } friend3_characteristics = { "склонность к анализу и рефлексии", "интеллект", "экстраверсия" } friend1_friend2_characteristics = friend1_characteristics & friend2_characteristics friend1_friend3_characteristics = friend3_characteristics & friend1_characteristics print("Общие характеристики для 1 и 2:", friend1_friend2_characteristics) print("Общие характеристики для 1 и 3:", friend1_friend3_characteristics) </pre>

Т.32. Создание словаря, работа с его элементами.

1	<p>Проект «Рекламное агентство». Рекламное агентство для организации маркетинговых мероприятий создает уникальную базу контактов своих клиентов. В этой базе каждый контакт должен иметь уникальный номер телефона и адрес электронной почты. Разработайте программу для обработки списка контактов, которая удаляет дубликаты электронных адресов и телефонных номеров в базе контактов агентства, а также выводит на экран информацию о контактах с дубликатами данных и список уникальных контактов.</p> <p>Примечание</p> <p>Для того, чтобы определить, содержится ли элемент в списке, используйте условие: <i>элемент in список</i>. А для того, чтобы определить отсутствие элемента в списке – условие <i>элемент not in список</i>. Например,</p> <pre> if el in my_list: print("Элемент есть в списке") if el not in my_list: print("Элемента нет в списке") </pre> <p>Этот код выведет надпись «Элемент есть в списке», если элемент el присутствует в списке my_list, и надпись «Элемента нет в списке», если отсутствует.</p>
2	<pre> contacts = [{"name": "Иванов", "email": "partner1@example.com", "phone": "123456789"}, {"name": "Петров", "email": "petrov@example.com", "phone": "987654321"}, {"name": "Сидоров", "email": "sidorov@example.com", "phone": "234567891"}, {"name": "Николаев", "email": "nikolaev@example.com", "phone": "111111111"}, {"name": "Кириллов", "email": "partner1@example.com", "phone": "123456789"}, {"name": "Орлов", "email": "petrov@example.com", "phone": "987654322"}, {"name": "Максимов", "email": "maksimov@example.com", "phone": "876543219"}, {"name": "Александров", "email": "partner2@example.com", "phone": "123456789"}, {"name": "Андреев", "email": "andreev@example.com", "phone": "111111111"}] </pre>

	<pre> uniq_emails = [] uniq_phones = [] uniq_contacts = [] print("Контакты с дублем почты или телефона:") for contact in contacts: if contact["email"] in uniq_emails and contact["phone"] in uniq_phones: print("Дублируется и почта, и телефон:", contact) elif contact["email"] in uniq_emails and contact["phone"] not in uniq_phones: print("Дублируется почта, телефон отличается:", contact) elif contact["email"] not in uniq_emails and contact["phone"] in uniq_phones: print("Дублируется телефон, почта отличается:", contact) else: uniq_emails.append(contact["email"]) uniq_phones.append(contact["phone"]) uniq_contacts.append(contact) print() print("Уникальные контакты:") for contact in uniq_contacts: print(contact) </pre>
--	--

T.33. Работа со словарями. Проект «Телефонный справочник».

1	<p>Проект «Каталог калорийности продуктов». Вы решили создать программу для учета калорийности продуктов вашего привычного набора питания. Программа позволит более подробно следить за балансом калорийности в вашем меню.</p> <p><i>Основные возможности программы:</i></p> <ul style="list-style-type: none"> ■ добавление новых продуктов – пользователь вводит его название и калорийность; ■ поиск значений по названию продукта – ввести название продукта и получить его калорийность; ■ удаление продуктов – удалять продукты из каталога. <p><i>Дополнительные требования:</i></p> <ul style="list-style-type: none"> ■ обеспечивать защиту от ошибок ввода данных; ■ использовать уникальный ключ (например, название продукта в нижнем регистре) для обеспечения уникальности записей в каталоге; <p>завершать работу программы по желанию.</p>
2	<pre> products = {} while True: print("1. Добавить новый продукт") print("2. Поиск калорийности по названию продукта") print("3. Удалить продукт") print("4. Выйти из программы") choice = input("Выберите действие: ") if choice == "1": name = input("Введите название продукта: ").lower() calories = float(input("Введите калорийность продукта (в ккал): ")) products[name] = calories print("Продукт успешно добавлен.") </pre>

	<pre> elif choice == "2": name = input("Введите название продукта для поиска: ").lower() if name in products: print("Калорийность продукта " + name + ": " + str(products[name]) + " ккал") else: print("Продукт с таким названием не найден") elif choice == "3": name = input("Введите название продукта для удаления: ").lower() if name in products: del products[name] print("Продукт " + name + " успешно удален") else: print("Продукт с таким названием не найден") elif choice == "4": print("Программа завершена.") break else: print("Некорректный ввод. Пожалуйста, выберите действие из списка.") </pre>
--	---

Т.34. Массивы и способы работы с ними.

1	<p>Проект «Анализ расходов». Разработайте программу для учета финансовых трат в течение месяца, с учетом категорий «продукты» и «не продукты». Программа запрашивает количество дней в текущем месяце и затем позволяет пользователю ввести ежедневные расходы на «продукты» и «не продукты», сохраняя их в соответствующих массивах. Для анализа трат программа запрашивает начальный и конечный день периода в этом месяце и выводит общую сумму расходов на «продукты» и «не продукты» за этот период, а также сравнивает средние траты за этот период со средними тратами за месяц по этим категориям.</p> <p>Примечание Для получения части списка (массива) используйте такой код:</p> <pre>nums_part = nums[start:stop+1]</pre> <p>Этот код скопирует из списка nums ячейки с индексами от start до stop и запишет их в список nums_part.</p>
2	<pre> days_in_month = int(input("Введите количество дней в текущем месяце: ")) expenses_food = [] expenses_non_food = [] for i in range(days_in_month): day_expenses_food = int(input("Введите сумму расходов на «продукты» за день " + str(i + 1) + ": ")) day_expenses_non_food = int(input("Введите сумму расходов на «не продукты» за день " + str(i + 1) + ": ")) expenses_food.append(day_expenses_food) expenses_non_food.append(day_expenses_non_food) print(expenses_food) print(expenses_non_food) start = int(input("Введите начальный день периода: ")) end = int(input("Введите конечный день периода: ")) total_food_period = sum(expenses_food[start - 1:end]) </pre>

	<pre> total_non_food_period = sum(expenses_non_food[start - 1:end]) print("Общая сумма расходов на «продукты» за период:", total_food_period) print("Общая сумма расходов на «не продукты» за период:", total_non_food_period) average_food_month = sum(expenses_food) / days_in_month average_non_food_month = sum(expenses_non_food) / days_in_month print("Средние траты на «продукты» за месяц:", average_food_month) print("Средние траты на «не продукты» за месяц:", average_non_food_month) if total_food_period / (end - start + 1) > average_food_month: print("Средние траты на «продукты» за указанный период больше средних трат за месяц.") elif total_food_period / (end - start + 1) < average_food_month: print("Средние траты на «продукты» за указанный период меньше средних трат за месяц.") else: print("Средние траты на «продукты» за указанный период равны средним тратам за месяц.") if total_non_food_period / (end - start + 1) > average_non_food_month: print("Средние траты на «не продукты» за указанный период больше средних трат за месяц.") elif total_non_food_period / (end - start + 1) < average_non_food_month: print("Средние траты на «не продукты» за указанный период меньше средних трат за месяц.") </pre>
--	---

Т.35. Типовые алгоритмы обработки числовых массивов.

1	<p>Проект «Метеослужба». Представьте, что вы работаете в метеослужбе и ваша задача – вести мониторинг дневных температур для проведения анализа погодных условий прошедшего месяца. Вы решаете создать программу для анализа массива температур, чтобы выявить следующие характеристики:</p> <ul style="list-style-type: none"> ■ максимальная и минимальная температура за месяц, чтобы определить экстремальные погодные условия; ■ среднюю температуру, чтобы понять общий климат за месяц; ■ количество дней, когда температура была близка к средней (изменения не более, чем на 2 градуса), что может указывать на стабильные погодные условия; ■ количество дней, когда были стрессовые скачки температуры (изменения более чем на 5 градусов), которые могут сигнализировать о потенциальных аномалиях. <p><i>Блок ввода данных:</i> запросите количество дней в месяце (или периоде наблюдений), запросите диапазон для температур, организуйте автоматическое заполнение массива температур случайными значениями в указанном диапазоне.</p>
2	<pre> from random import * num_days = int(input("Введите количество дней в месяце: ")) min_temp = int(input("Введите минимальную температуру: ")) max_temp = int(input("Введите максимальную температуру: ")) temperature = [randint(min_temp, max_temp) for i in range(num_days)] print("Массив температур за месяц:", temperature) print("Максимальная температура за месяц:", max(temperature)) print("Минимальная температура за месяц:", min(temperature)) average_temperature = sum(temperature) / num_days print("Средняя температура за месяц:", average_temperature) calm_day = 0 for temp in temperature: if abs(temp - average_temperature) <= 2: calm_day += 1 print("Количество дней с температурой близкой к средней:", calm_day) </pre>

	<pre> stress_day = 0 for i in range(num_days - 1): if abs(temperature[i] - temperature[i + 1]) > 5: stress_day += 1 print("Количество стрессовых скачков температуры за месяц:", stress_day) </pre>
--	--

T.36. Двумерные массивы. Основные алгоритмы обработки двумерных массивов.

1	<p>Проект «Рост волейболистов». На соревнованиях по волейболу прибыло несколько команд, каждая из которых заявила на участие по 14 спортсменов. Разработайте программу для спортивных медиков, которая даст возможность получать следующие сведения:</p> <ul style="list-style-type: none"> • самый высокий и самый низкий рост – определять рост самого высокого и самого низкого игрока из всех участников соревнований; • средний рост для каждой команды – рассчитывать средний рост игроков в команде; • самая «высокая» команда – выводить номер команды, в которой средний рост игроков наибольший. <p><i>Блок ввода данных:</i> запросите количество команд, программа сама задает «разумный диапазон» для роста игроков от 165 см до 224 см* и организует автоматическое заполнение двумерного массива (количество команд × 14) случайными числами в указанном диапазоне.</p> <p>* самый высокий игрок – Вуттичай Суксала, Тайланд, 224 см; самый низкий игрок – Фархад Зариф, Иран, 165 см.</p>
2	<pre> from random import * count_teams = int(input("Введите количество команд: ")) min_height = 165 max_height = 224 # Формирование данных teams = [] for i in range(count_teams): team = [randint(min_height, max_height) for i in range(14)] print("Рост игроков в команде", str(i+1) + ":", team) teams.append(team) # Определяем рост самого высокого и самого низкого игрока из всех участников соревнований max_height = max(teams[0]) min_height = min(teams[0]) for i in range(1, count_teams): if max(teams[i]) > max_height: max_height = max(teams[i]) if min(teams[i]) < min_height: min_height = min(teams[i]) print("Самый высокий игрок:", max_height, "см") print("Самый низкий игрок:", min_height, "см") # Определяем средний рост игроков в каждой команде, находим самую «высокую» команду max_avg_height = 0 max_avg_team = 0 </pre>

	<pre> for i in range(count_teams): avg_height = round(sum(teams[i]) / 14, 2) print("Средний рост в команде", str(i+1) + ":", avg_height, "см") if avg_height > max_avg_height: max_avg_height = avg_height max_avg_team = i + 1 print("Самая 'высокая' команда:", max_avg_team) </pre>
--	---

Т.37. Пользовательские функции и методы работы с ними.

1	<p>Проект «Трекер привычек». Вы решили улучшить свои привычки, чтобы вести более здоровый образ жизни. Вы определили три ключевые области, на которых хотите сосредоточиться: сон, питание и прогулки. Напишите программу, которая поможет отслеживать привычки, сравнивая фактические значения с желаемыми. Программа должна:</p> <ul style="list-style-type: none"> ■ использовать словарь для хранения информации о привычках (сон, питание, прогулки); ■ запрашивать желаемые значения для каждой области; ■ позволять вводить фактические значения для каждой области; ■ генерировать отчет, показывающий фактические и желаемые значения; ■ спрашивать о продолжении отслеживания или выходе из программы. <p>Получение и вывод данных вынесите в пользовательские функции. <i>Пример вывода программы:</i> Отчет о привычках: Сон: факт - 7 часов, цель - 8 часов Питание: факт - 2200 калорий, цель - 2000 калорий Прогулка: факт - 20 минут, цель - 30 минут Продолжить отслеживание (да/нет)?</p>
2	<pre> def get_values(): values = {} for habit_type in habit_types: desired_value = input(habit_type + ": ") values[habit_type] = desired_value return values def print_report(desired, actual): print() print("Отчет о привычках:") for habit_type in habit_types: print(habit_type + ": факт - " + actual[habit_type] + ", цель - " + desired[habit_type]) print() habit_types = ["Сон", "Питание", "Прогулка"] print("Введите желаемые значения") desired_values = get_values() while True: print("Введите фактические значения") actual_values = get_values() </pre>

	<pre> print_report(desired_values, actual_values) choice = input("Продолжить отслеживание (да/нет)? ") if choice == "нет": break </pre>
--	---

Т.38. Разбиение задачи на подзадачи с использованием функций. Проект «Загадки». Рекурсия.

1	<p>Проект «Конструктор рецептов пиццы». Разработайте программу «Конструктор рецептов пиццы». Пользователю предлагается выбрать тип соуса, начинку и тип сыра. После ввода данных программа выводит рецепт пиццы и предлагает выбрать, создать ли еще один рецепт или завершить программу.</p> <p>Для хранения перечня продуктов используйте списки. Выбор пользователя и формирование рецепта вынесите в пользовательские функции.</p> <p><i>Пример работы программы:</i></p> <p>Выберите соус: 1 - томатный 2 - барбекю 3 - песто 4 - карбонара 5 - терияки 6 - бешамель </p> <p>Введите номер или несколько номеров через пробел: 1</p> <p>Выберите продукты для начинки: 1 - ветчина 2 - грибы 3 - бекон 4 - колбаса 5 - оливки 6 - помидоры 7 - маслины 8 - курица 9 - морепродукты </p> <p>Введите номер или несколько номеров через пробел: 1 2 3</p> <p>Выберите сыр: 1 - моцарелла 2 - чеддер 3 - пармезан 4 - фета 5 - рикотта 6 - маскарпоне 7 - грюйер </p> <p>Введите номер или несколько номеров через пробел: 1</p> <p>Рецепт пиццы</p> <p>Разогрейте духовку до 200 градусов.</p> <p>Возьмите полуфабрикат заготовки пиццы, положите её на противень.</p> <p>Смажьте соусом томатный</p> <p>Разложите ровным слоем начинку: ветчина, грибы, бекон</p> <p>Посыпьте сверху сыром моцарелла</p> <p>Отправьте в духовку на 25 минут.</p> <p>Хотите создать еще один рецепт? (да/нет)</p>
2	<pre> def get_user_choice(name, products_list): print("Выберите", name + ": ", end="") for i in range(len(products_list)): print(i+1, "-", products_list[i], end=" ") print() print("Введите номер или несколько номеров через пробел: ", end="") nums = [int(i) for i in input().split()] choice = "" for num in nums: if num > 0 and num < len(products_list) + 1: choice += products_list[num-1] + ", " else: print("Продукта с номером", num, "нет") return choice[:-2] def print_recipe(user_sauce, user_toppings, user_cheese): print() print("Рецепт пиццы") </pre>

	<pre> print("Разогрейте духовку до 200 градусов.") print("Возьмите полуфабрикат заготовки пиццы, положите её на противень.") print("Смажьте соусом", user_sauce) print("Разложите ровным слоем начинку:", user_toppings) print("Посыпьте сверху сыром", user_cheese) print("Отправьте в духовку на 25 минут.") print() sauces = ['томатный', 'барбекю', 'песто', 'карбонара', 'терияки', 'бешамель'] toppings = ['ветчина', 'грибы', 'бекон', 'колбаса', 'оливки', 'помидоры', 'маслины', 'курица', 'морепродукты'] cheeses = ['моцарелла', 'чеддер', 'пармезан', 'фета', 'рикотта', 'маскарпоне', 'грюйер'] while True: user_sauce = get_user_choice("соус", sauces) user_toppings = get_user_choice("продукты для начинки", toppings) user_cheese = get_user_choice("сыр", cheeses) print_recipe(user_sauce, user_toppings, user_cheese) choice = input("Хотите создать еще один рецепт? (да/нет): ") if choice.lower() != 'да': break </pre>
--	---

Т.39. Алгоритм «Мои результаты». Проведение итоговой викторины.

1

Проект «Электронный методист». Разработайте программу «Электронный методист», которая поможет пользователям изучать английский язык самостоятельно. Программа должна включать в себя следующее:

- функция выбора уровня языка – пользовательская функция для выбора уровня языка из шкалы CEFR (A1, A2, B1, B2, C1, C2);
- вывод рекомендуемых нормативов по видам учебной деятельности (слушание, говорение, лексика, грамматика) для выбранного уровня языка;
- функция мониторинга времени и выдачи рекомендаций – пользователь вводит фактически потраченное время на каждый вид учебной деятельности и получает рекомендацию по коррекции распределения в случае необходимости;
- возможность повторения программы.

Рекомендации:


уровень	Слушание (%)	Говорение (%)	Лексика (%)	Грамматика (%)
A1 (новичок)	50	30	10	10
A2 (элементарный)	40	35	15	10
B1 (средний)	30	35	20	15
B2 (выше среднего)	25	35	25	15
C1 (продвинутый)	20	35	30	15
C2 (свободное владение)	15	35	35	15

2

```
def select_language_level(levels):  
    text = ""  
    for lvl in levels:  
        text += lvl + ", "  
    text = "Выберите уровень языка (" + text[:-2] + "): "
```

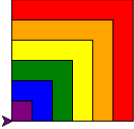
	<pre> while True: level = input(text).upper() if level in levels: break else: print("Некорректный ввод. Попробуйте снова.") return level def print_norms(level, norms, activities): print() print("Рекомендуемые нормы для уровня", level) for i in range(len(activities)): print(activities[i], "-", norms[i], " ", end="") print() def monitor_time(norms, activities): print() time_spent = [] for i in range(len(activities)): tm = int(input("Сколько времени вы потратили на " + activities[i] + " (в минутах)?: ")) time_spent.append(tm) total_spent = sum(time_spent) for i in range(len(activities)): percent = round(time_spent[i] / total_spent * 100) if percent < norms[i]: print("Рекомендуется добавить еще", norms[i] - percent, "% времени на", activities[i]) print() </pre>
--	---

Т.40. Модуль turtle: модель RGB, понятие объект, основные команды управления.


1	<p>Проект «Меандр». Меандр – распространенный тип ортогонального орнамента. В Древнем Риме им украшали подол одежды, а в древнерусской архитектуре применяли в рельефах и фризах. Напишите программу для рисования меандра, ориентируясь на рисунок справа. Используйте цикл для создания ленточного орнамента, самостоятельно выбрав количество повторов (не выходите за размеры экрана).</p> 
2	<pre> from turtle import * setup(1000, 600) bgcolor("grey") pen = Turtle() for i in range(3): pen.fd(20) pen.lt(90) pen.fd(80) pen.rt(90) pen.fd(80) pen.rt(90) </pre>

	<pre> pen.fd(60) pen.rt(90) pen.fd(40) pen.rt(90) pen.fd(20) pen.rt(90) pen.fd(20) pen.lt(90) pen.fd(20) pen.lt(90) pen.fd(40) pen.lt(90) pen.fd(60) pen.lt(90) pen.fd(60) </pre>
--	---

Т.41. Модуль turtle: рисование замкнутых многоугольников, заливка фигур.


1	<p>Проект «Вложенные квадраты». Напишите программу, которая рисует шесть вложенных квадратов разных цветов. При каждом проходе цикла размер фигуры уменьшается на 20 единиц, что создает эффект вложенности. Для заливки квадратов используются цвета: "red", "orange", "yellow", "green", "blue", "purple".</p> 
2	<pre> from turtle import * pen = Turtle() colors = ["red", "orange", "yellow", "green", "blue", "purple"] size = 120 for i in range(len(colors)): pen.color("black", colors[i]) pen.begin_fill() for j in range(4): pen.forward(size) pen.left(90) pen.end_fill() size -= 20 </pre>

Т.42. Модуль turtle: рисование полных и неполных окружностей.

1	<p>Проект «Символ Wi-Fi». Напишите программу для рисования символа Wi-Fi. Подберите толщину и цвет линии на свое усмотрение. Дуги «сигнала» равны четверти окружности.</p> 
---	--

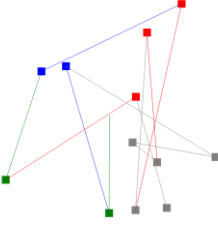
2	<pre> from turtle import * setup(400, 400) pen = Turtle() pen.color("blue") pen.width(3) pen.begin_fill() pen.circle(3) pen.end_fill() pen.up() pen.seth(45) pen.fd(20) pen.seth(135) pen.down() pen.circle(20, 90) pen.up() pen.seth(135) pen.fd(20) pen.seth(45) pen.down() pen.circle(-40, 90) pen.up() pen.seth(45) pen.fd(20) pen.seth(135) pen.down() pen.circle(60, 90) </pre>
---	---

Т.43. Модуль turtle: рисование полных и неполных окружностей.

1	<p>Проект «Разноцветные квадраты». Напишите программу, которая рисует «круг» из разноцветных квадратов. Создайте функцию, которая случайным образом формирует цвет заливки квадрата. Рисуйте разноцветные квадраты, поворачивая на небольшой угол каждый следующий квадрат. Повторяйте рисование до завершения круга.</p> 
2	<pre> from turtle import * from random import * setup(800, 600) pen = Turtle() def rand_colors(): r = random() g = random() b = random() return r, g, b num_squares = int(input("Введите количество квадратов: ")) angle = 360 / num_squares for i in range(num_squares): </pre>

	<pre> rand_colors() pen.color("black", rand_colors()) pen.begin_fill() for j in range(4): pen.forward(40) pen.right(90) pen.end_fill() pen.right(angle) </pre>
--	--

Т.44. Модуль turtle: перемещение объекта по координатам, работа с костюмами.

1	<p>Проект «Броуновское движение в цвете». Напишите программу, которая в окне для рисования пером в форме квадрата имитирует модель броуновского движения (то есть движение по случайным координатам). Выберите форму пера – круг или квадрат, изменяйте цвет пера в зависимости от положения – в каждой координатной четверти свой цвет. Программа запрашивает у пользователя количество этапов в этом движении. Для наглядности оставляйте след костюма в каждой точке этапа перемещения.</p> 
2	<pre> from turtle import * from random import * def change_color(x, y): if x > 0 and y > 0: pen.color("red") elif x < 0 and y > 0: pen.color("blue") elif x < 0 and y < 0: pen.color("green") else: pen.color("grey") setup(800, 800) pen = Turtle() pen.shape("square") num_steps = int(input("Введите количество этапов: ")) for i in range(num_steps): x = randint(-300, 300) y = randint(-300, 300) change_color(x, y) pen.goto(x, y) pen.stamp() </pre>

Т.45. Модуль turtle: обработка событий «мыши» для управления анимацией.

1	Проект «Эффект огня». Напишите программу, которая создает эффект «огня» на нажатие кнопки мыши. Задайте черный цвет фона. Когда пользователь нажимает на кнопку мыши, на месте курсора появляется эффект «огня», представленный круглым объектом, меняющим цвет и размер: цвет случайным образом меняется на оранжевый, желтый или красный, а размер изменяется случайным образом в диапазоне от 1 до 3. Объект мигает, изменяя цвет и размер несколько раз, с установленной краткой задержкой во времени.
2	<pre> from turtle import * from random import * from time import * def pen_effect(x, y): pen.goto(x, y) for i in range(10): pen.color(choice(colors)) pen.shapesize(randint(1, 3)) sleep(0.1) setup(500, 500) bgcolor("black") pen = Turtle() pen.shape("circle") pen.color("yellow") pen.up() colors = ["orange", "yellow", "red"] onscreenclick(pen_effect, 1) </pre>

Т.46. Модуль turtle: обработка событий клавиатуры для управления анимацией.

1	Проект «Веселые черепашки». Напишите программу, которая позволяет управлять движением объекта «черепашка» на экране с помощью клавиш клавиатуры. При нажатии клавиш вверх, вниз, влево или вправо объект перемещается в соответствующую сторону на определенное расстояние, при этом меняя свой размер и цвет случайным образом. При каждом перемещении объект оставляет за собой след. Чем больше будет вариантов для случайного выбора, тем «веселее» будут черепашки.
2	<pre> from turtle import * from random import * def random_turtle(): pen.color(choice(["red", "green", "blue", "orange", "purple"])) pen.shapesize(randint(1, 3)) def up(): pen.seth(90) random_turtle() pen.forward(50) def down(): pen.setheading(270) random_turtle() pen.forward(50) def left(): pen.setheading(180) random_turtle() pen.forward(50) def right(): </pre>

	<pre>pen.setheading(0) random_turtle() pen.forward(50) setup(800, 600) pen = Turtle() pen.shape("turtle") pen.shapesize(2) onkeypress(up, "Up") onkeypress(down, "Down") onkeypress(left, "Left") onkeypress(right, "Right") listen()</pre>
--	---