1. Реализовать скрипт, в котором должна быть предусмотрена функция расчета заработной платы сотрудника. В расчете необходимо использовать формулу: (выработка в часах\*ставка в час) + премия. Для выполнения расчета для конкретных значений необходимо запускать скрипт с параметрами.

**def** simple\_calc():  
 x = float(input(**"Введите количество отработанных часов : "**))  
 y = float(input(**"Введите ставку оплаты труда за 1 час : "**))  
 c = float(input(**"Укажите сумму премии - "**))  
 pay = x \* y  
 **return** pay + c  
print(**f"Размер заработной платы сотрудника: {**simple\_calc() **}"**)

2. Представлен список чисел. Необходимо вывести элементы исходного списка, значения которых больше предыдущего элемента.

**Подсказка:** элементы, удовлетворяющие условию, оформить в виде списка. Для формирования списка использовать генератор.  
**Пример исходного списка:** [300, 2, 12, 44, 1, 1, 4, 10, 7, 1, 78, 123, 55].  
**Результат:** [12, 44, 4, 10, 78, 123].

list = [300, 2, 12, 44, 1, 1, 4, 10, 7, 1, 78, 123, 55]  
result\_list = [number **for** i, number **in** enumerate(list) **if** i > 0 **and** list[i] > list[i - 1]]  
print(**"Исходный список: "**, list)  
print(**"Список, элементы которого больше предыдущего: "**, result\_list)

3. Для чисел в пределах от 20 до 240 найти числа, кратные 20 или 21. Необходимо решить задание в одну строку.

**Подсказка:** использовать функцию range() и генератор.

list = [i **for** i **in** range(20, 240) **if** i % 20 == 0 **or** i % 21 == 0]  
print(**"Список чисел кратных 20 или 21: "**, list)

4. Представлен список чисел. Определить элементы списка, не имеющие повторений. Сформировать итоговый массив чисел, соответствующих требованию. Элементы вывести в порядке их следования в исходном списке. Для выполнения задания обязательно использовать генератор.

**Пример исходного списка:** [2, 2, 2, 7, 23, 1, 44, 44, 3, 2, 10, 7, 4, 11].  
**Результат:** [23, 1, 3, 10, 4, 11]

my\_list = [2, 2, 2, 7, 23, 1, 44, 44, 3, 2, 10, 7, 4, 11]  
print(**"Исходные элементы списка:\n"**, my\_list)  
new\_list = [i **for** i **in** my\_list **if** my\_list.count(i) == 1]  
print(**"Элементы списка, не имеющие повторений:\n"**, new\_list)

5. Реализовать формирование списка, используя функцию range() и возможности генератора. В список должны войти четные числа от 100 до 1000 (включая границы). Необходимо получить результат вычисления произведения всех элементов списка.

**Подсказка:** использовать функцию reduce().

**from** functools **import** reduce  
  
list = [i **for** i **in** range(100, 1002, 2)]  
print(**"Список чётных чисел в диапазоне [100..1000]:\n"**, list)  
print(**"Произведение всех элементов списка:\n"**, reduce(**lambda** x,y: x\*y, list))

6. Реализовать два небольших скрипта:

а) итератор, генерирующий целые числа, начиная с указанного,  
б) итератор, повторяющий элементы некоторого списка, определенного заранее.

**Подсказка:** использовать функцию count() и cycle() модуля itertools. Обратите внимание, что создаваемый цикл не должен быть бесконечным. Необходимо предусмотреть условие его завершения.

Например, в первом задании выводим целые числа, начиная с 3, а при достижении числа 10 завершаем цикл. Во втором также необходимо предусмотреть условие, при котором повторение элементов списка будет прекращено.

*#итератор, генерирующий целые числа, начиная с указанного***from** itertools **import** count, cycle  
  
print(**"Итератор, генерирующий целые числа, начиная с указанного:"**)  
**for** item **in** count(3):  
 **if** item > 10:  
 **break  
 else**:  
 print(item)  
  
*#итератор, повторяющий элементы некоторого списка, определенного заранее*print(**"\nИтератор, повторяющий элементы некоторого списка, определенного заранее:"**)  
count = 0  
**for** item **in** cycle([1, 2, 3]):  
 **if** count > 8:  
 **break** print(item)  
 count += 1

7. Реализовать генератор с помощью функции с ключевым словом yield, создающим очередное значение. При вызове функции должен создаваться объект-генератор. Функция должна вызываться следующим образом: for el in fact(n). Функция отвечает за получение факториала числа, а в цикле необходимо выводить только первые n чисел, начиная с 1! и до n!.

**Подсказка:** факториал числа n — произведение чисел от 1 до n. Например, факториал четырёх 4! = 1 \* 2 \* 3 \* 4 = 24.

**from** math **import** factorial  
  
**def** fact(n):  
 result = 1  
 **for** i **in** range(1, n + 1):  
 result \*= i  
 **yield** result  
  
**for** el **in** fact(10):  
 print(el)