**Информация об уроке**

**Модуль: 1**

**Номер занятия: 11**

**Тип занятия: Практика**

**Название занятия/ Тема занятия:** Циклы; операторы Break, Continue; Массивы

**Цель занятия:** изучить базовые настройки Python.

**Образовательные результаты:** Изучить переменные, их типы данных и базовые операторы.

**Глоссарий:**



**Конспект занятия**

1. **Приветствие**

Приветствуем вас на новом курсе по Python!

1. **Тема урока и целеполагание**

Сегодня мы пройдём:

1. Циклы;
2. операторы Break, Continue;
3. Массивы

Эти темы также входят в базовые во всех языках программирования, не только в Python.

1. **Актуализация**

Циклы позволяют выполнять некоторое действие в зависимости от соблюдения некоторого условия. В языке Python есть следующие типы циклов:

* **for**
* **while**

Цикл while проверяет истинность некоторого условия, и если условие истинно, то выполняет инструкции цикла.

После ключевого слова while указывается условное выражение, и пока это выражение возвращает значение True, будет выполняться блок инструкций, который идет далее.

Все инструкции, которые относятся к циклу while, располагаются на последующих строках и должны иметь отступ от начала ключевого слова while. Для цикла while также можно определить дополнительный блок **else**, инструкции которого выполняются, когда условие равно False:

Другой тип циклов представляет конструкция for. Этот цикл пробегается по набору значений, помещает каждое значение в переменную, и затем в цикле мы можем с этой переменной производить различные действия.

После ключевого слова for идет название переменной, в которую будут помещаться значения. Затем после оператора in указывается набор значений и двоеточие.

При выполнении цикла Python последовательно получает все значения из набора и передает их в переменную. Когда все значения из набора будут перебраны, цикл завершает свою работу.

Цикл for также может иметь дополнительный блок else, который выполняется после завершения цикла.

Одни циклы внутри себя могут содержать другие циклы.

**Выход из цикла. break и continue**

Оператор break осуществляет выход из цикла. А оператор continue выполняет переход к следующей итерации цикла.

**Массив** (Список или list) представляет тип данных, который хранит набор или последовательность элементов.в.

Для создания списка применяются квадратные скобки **[]**, внутри которых через запятую перечисляются элементы списка.

Также для создания списка можно использовать функцию-конструктор **list()**:

Список необязательно должен содержать только однотипные объекты.

**Обращение к элементам списка**

Для обращения к элементам списка надо использовать индексы, которые представляют номер элемента в списка. Индексы начинаются с нуля. То есть первый элемент будет иметь индекс 0, второй элемент - индекс 1 и так далее.

**Перебор элементов**

Для перебора элементов можно использовать как цикл for, так и цикл while.

Для перебора с помощью функции **len()** получаем длину списка. С помощью счетчика i выводит по элементу, пока значение счетчика не станет равно длине списка.

функция **append()** добавляет элемент в конец массива:

**a = []**

**b = int(input())  
a.append(b)**

1. **Основное содержание**

**ЗАДАЧИ:**

**Задача 1:**

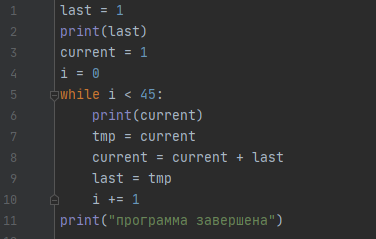
Разработать программу, которая выводит первые 45 чисел Фибоначчи.

Пример чисел Фибоначчи: 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, …

Каждое следующее число это сумма двух предыдущих.

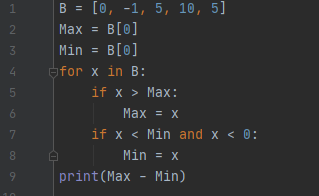
для начала определимся с тем, как мы будем получать эти числа и выводить их. Создавать под каждое число переменную плохое решение, так как мы не знаем сколько именно чисел потребуется вывести и мы потратим больше памяти на переменные, так как они будут дальше храниться в памяти.

Используем цикл.



**Задача 2:**

Найти разницу между самым большим положительным числом и самым маленьким отрицательным числом в целочисленном массиве B[n],



**3)** Ученики 8-го класса написали контрольную работу. Всем, кто получил тройку, придется ее переписать.  
Дан список оценок, заканчивающийся нулем. Выведи, сколько учеников переписывали контрольную.

Пример:

Ввод: 3 Вывод: 2

3

0

Ввод: 5 Вывод: 2

4

4

3

4

3

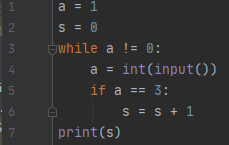
0

**Решение**

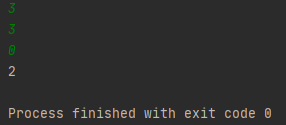
Для ввода используем цикл while. Теперь необходимо сформировать условие, при котором этот цикл завершится: по условию это ввод 0.

То есть если == 0 - False, при котором новая итерация цикла не начинается, то != 0 - True, при котором цикл работает.

Также нам необходимо вводить переменную, данные из которой будем использовать в подсчете количества переписывающих учеников. В цикле мы будем вводить эту самую переменную и если ее значение равно 3, то будем увеличивать переменную счетчик.



Консольный вывод:



**4)** На школьный праздник все принесли воздушные шары, чтобы вместе их запустить.

Дано количество шаров, которое принес каждый ученик, а в конце число 0. Выведи сколько шаров взлетело в небо.

Пример:

Ввод: 1 Вывод: 6

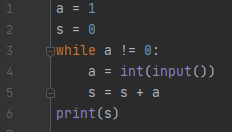
3

2

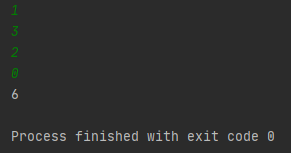
0

**Решение**

Аналогично задаче 3, будем производить ввод переменной через консоль в цикле, концом ввода будет цифра 0. Каждое введенное значение будем прибавлять к общей сумме.



Консольный вывод:



**5)** У Димы есть n маленьких постеров. Он хочет повесить их в ряд один над другим. То место на стене, где будет середина каждого постера, он отмечает символом s.

Помоги Диме разметить стену.

Даны символ s и число постеров n. Выведи символ s в столбик n раз, чтобы увидеть, как постеры будут висеть на стене.

Пример:

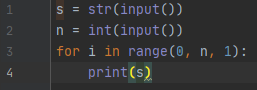
Ввод: + Вывод: +

3 +

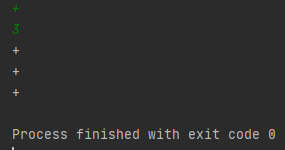
+

**Решение**

Запустим цикл for, в котором будет n итераций, в каждой из которых будем выводить символ +.



Консольный вывод:



**6)** Чтобы заново зашифровать послание, Соне нужно найти минимальное из чисел.

Дано число n и последовательность из n чисел. Выведи результат, который получила Соня.

Пример:

Ввод: 3 Вывод: 1

1

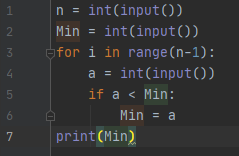
2

3

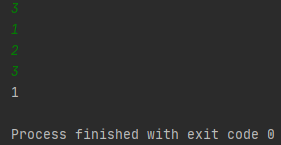
**Решение**

Здесь ситуация противоположна циклу while (задача 3,4). Сначала мы вводим количество чисел, затем в цикле for вводим их. В процессе ввода сразу определяем (в цикле) минимальное число.

Первое число мы вводим Min, поскольку мы не знаем какое может быть минимальное число в задаче, изначально за него мы принимаем первый ввод. Отсюда следует, что цикл будет выполнен на 1 раз меньше.



Консольный вывод:



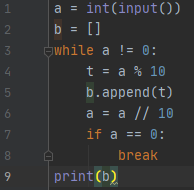
**7)** Разработать программу, которая записывает цифры заданного натурального числа в массив. После чего, пользователь вводит с клавиатуры два любых индекса этого массива. Программа должна вывести на экран сумму соответствующих индексам цифр массива.

**Решение**

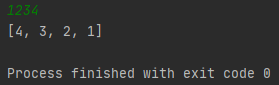
Данную задачу можно разбить на несколько подзадач:

1. Найти цифры заданного натурального числа
2. записать цифры в массив
3. организовать пользовательский вывод массива с индексами
4. организовать корректный ввод индексов (чтобы ввод значений за пределами массива не крашил программу)
5. найти сумму заданных индексов массива.

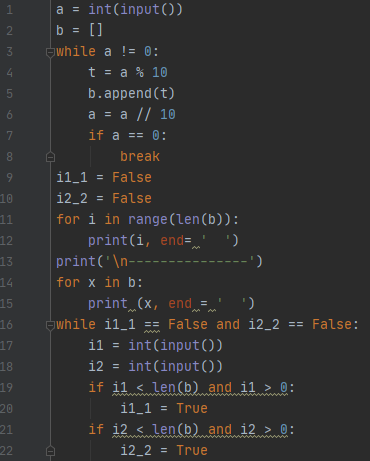
Начнём с первой подзадачи:



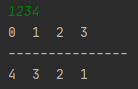
Консольный вывод:



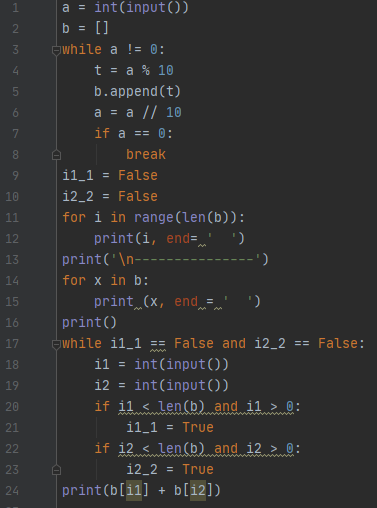
Следующая подзадача - организовать ввод пользователем индексов:



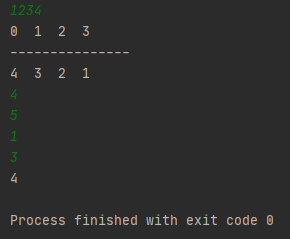
Консольный вывод:



Сумма заданных индексов массива:



Консольный вывод:



1. **Подведение итогов/Рефлексия**

На этом занятии мы узнали что такое циклы, каких видов они бывают. Что делают операторы break, continue. Также мы изучили массивы и методы работы с ними.