Оператор вывода инфы – **print**

* print(2 + 3)
* print("Hello")

Некоторые конструкции языка являются многострочными и к ним пременяеться отсту с помощью **таба**

* if 1 < 2:
* print("Hello")

**python - регистро-зависимый язык**

text = "Message:\n\"Hello World\"" print(text)

* userName = "Tom"
* userAge = 37
* user = f"name: {userName} age: {userAge}"
* print(user) # name: Tom age: 37

консольный ввод данных **- input**

* name = input("Введите свое имя: ")
* print(f"Ваше имя: {name}")

арифметические опперации с присвоением

* +=
* Присвоение результата сложения
* -=
* Присвоение результата вычитания
* \*=
* Присвоение результата умножения
* /=
* Присвоение результата от деления
* //=
* Присвоение результата целочисленного деления
* \*\*=
* Присвоение степени числа
* %=
* Присвоение остатка от деления

условный опператор

**if логическое\_выражение:**

**инструкции**

**[elif логическое выражение:**

**инструкции]**

**[else:**

**инструкции]**

* language = "english"
* if language == "english":
* print("Hello")
* print("End")

25.09.2024

Цикл For – данный цикл пробегает по набору значений, спомощью переменной цикла

* ------------------------
* for переменная in набор\_значений: инструкции
* ------------------------

Конструкции вложенных циклов:

for c1 in "ab":

for c2 in "ba":

print(f"{c1}{c2}")

**08.10.2024**

Функция - часть кода выполняющая определённую задачу, которую можно использовать в различных частях программы.

Функции делятся на "Внутрение"(которые уже есть в python, например print) и пользовательсякие функции(те которые мы определяем сами).

def имя\_функции ([параметры]):

инструкции

одни функции могут определяться внутри других функций(внутренние функции - локальные, они могут использоваться только внутри той функции которой они определены)

в программе может быть определено n кол-во функций для их упорядочивания, например используеться специальная функция main()

**параметры функции**

используються для передачи в функцию данных .

параметров может быть несколько .

Значения могут передоваться следующим образом:

def print\_person(name, age):

print(f"Name: {name} Age: {age}")

print\_person(age = 22, name = "Tom")

с помощью '\*' можно определить параметр в который можно передовать n кол-во значений

**оператор return**

функция может возвращать результат

функция может быть использована как параметр другой функции

lambda выражение - небольшие анонимные функции, котороые определяються с помощью оператора lambda

премущество - сокращение определения функции, но с точки зрения недостатков - они могут выполнять только 1-но выражение

**Списки**

Список(List) - тип данных который хранит набор элементов (анагол массивов в др. ЯВУ)

для создания списка используються []

C точки зрения вывода элемента массивов используеться print()

Для обращения к элементам массивов используються индексы

Для перебора элементов массива можно испольлзовать следующие функции:

Списки можно сравнивать, причём 2 списка равны, если они содержат одинаковый набор элементов

Мы можем взять часть списка (срез)

методы для работы со списком:

* append(item): добавляет элемент item в конец списка
* insert(index, item): добавляет элемент item в список по индексу index
* extend(items): добавляет набор элементов items в конец списка
* remove(item): удаляет элемент item. Удаляется только первое вхождение элемента. Если элемент не найден, генерирует исключение ValueError
* clear(): удаление всех элементов из списка
* index(item): возвращает индекс элемента item. Если элемент не найден, генерирует исключение ValueError
* pop([index]): удаляет и возвращает элемент по индексу index. Если индекс не передан, то просто удаляет последний элемент.
* count(item): возвращает количество вхождений элемента item в список
* sort([key]): сортирует элементы. По умолчанию сортирует по возрастанию. Но с помощью параметра key мы можем передать функцию сортировки.
* reverse(): расставляет все элементы в списке в обратном порядке
* copy(): копирует список

Функции:

* len(list): возвращает длину списка
* sorted(list, [key]): возвращает отсортированный список
* min(list): возвращает наименьший элемент списка
* max(list): возвращает наибольший элемент списка