Функции. Словари. Интерпретатор. Файлы. Модули.

Функции.

Разбиение на логические части.

Сокрытие деталей реализации.

Переиспользование кода

Создание новых примитивов

Объявление функции:

Def имя функции (def min2(a,b))

Аргументы (параметры)

Тело функции(if a,=b:

return a

else:

return b)

Return

Вызов функции:

Функция должна быть объявлена ранее первого вызова

Различные функции:

Без возвращаемого значения

Без параметров

Произвольное число параметров (через звёздочку)

Параметры со значениями по умолчанию (через равно и некое значение по умолчанию)

Локальные переменные:

Переменные, которые задействованы только внутри функции

Глобальные переменные:

Переменные, объявленные вне всяких функций, их можно использовать во всей программе.

Множества.

Позволяют хранить наборы данных и быстро отвечают на запросы.

Set - множество

Операции с множествами:

.add() – добавление элемента

.remove() - удаление

.discard() – удаление без ошибок при отсутствии

.clear() – удаляет всё

Len() – число элементов

Словари.

Ассоциативный массив

Позволяет хранить ключ и значение.

Dict – словарь

Изменяемы

Элементы не имеют порядка

Все ключи различны

Ключи неизменяемы

Файловый ввод и вывод:

Чтение из файла:

Inf = open() – чтение

Readline – чтение одной строки

Close – закрытие файла

With open () as inf:

/// тело работы с файлом

Дальше файл закроется. Этот способ рекомендуется.

Полезные функции:

.strip() – убирает все служебные символы при чтении

Os.path.join – возвращает полный путь к файлу

Нужно подключить модуль os

Запись в файл:

Обязательно указать режим открытие ‘w’

.write(текст) – добавление текста в файл

Str – преобразование в строку

Также можно использовать конструкцию with

Модули:

Модуль содержит функции и данные в отдельном файле

Объекты из модуля можно импортировать в другие модули

Импорт модуля.

Рассмотрим файл my\_module.py

Пусть в нём описана функция foo()

Import my\_module

My module.foo()

Импорт from

From my\_module import foo

Foo()