Рубежный контроль № 3: конспект по скриптовому языку

30 декабря 2023 г.

Бойко Роман, ИУ9-12Б

Язык программирования Python

1. Типизация и система типов языка

Классификации систем типов:

- Наличие системы типов: есть.
- Типизация динамическая.
- Типизация неявная.
- Типизация сильная.
- Основные типы данных в Python:
  - Простые:
    - Числовые типы: int (целые числа), float (числа с плавающей точкой), complex (комплексные числа)
    - Булевой тип: bool (True или False)
    - NoneType: нейтральное пустое значение, аналогичное null в других языках программирования
  - Составные:
    - Строковый тип: str
    - Списки: list (изменяемый упорядоченный набор элементов)
    - Кортежи: tuple (неизменяемый упорядоченный набор элементов)
    - Множества: set (неупорядоченная коллекция уникальных элементов)
    - Словари: dict (неупорядоченная коллекция пар ключ-значение)
    - Байтовые типы – bytes (байты), bytearray(изменяемая байтовая строка)

2. Основные управляющие конструкции

Прим.: Конструкция else выдет себя по-разному в зависимости от того, где она вызвана - Условные операторы: - if: выполняет определенные действия в зависимости от условия - elif: дополнительное условие для if - else: выполняется, если ни одно из условий не является истинным - Циклы: - for: выполняет набор операций для каждого элемента в последовательности - while: выполняет набор операций, пока условие остается истинным - break: прерывает выполнение цикла - continue: пропускает текущую итерацию цикла и переходит к следующей - else: блок инструкций внутри else выполнится только в том случае, если выход из цикла произошел без помощи break. - Функции: - def: начало объявления функции - lambda: начало объявления анонимной функции - Исключения: - try: в блоке try мы выполняем инструкцию, которая может породить исключение - except: в блоке except мы перехватываем исключение - finally: выполняет блок инструкций в любом случае, было ли исключение, или нет - else: выполняется в том случае, если исключения не было - Контекстный менеджер: - with…as: используется для оборачивания выполнения блока инструкций менеджером контекста. - Другие управляющие конструкции: - pass: блок, который ничего не делает - assert: это утверждение, которое проверяет, является ли условие истинным. Если условие истинно, выполнение программы продолжается. Если условие ложно, оператор assert вызывает исключение AssertionError, что приводит к остановке выполнения программы.

3. Подмножество языка для функционального программирования

- Иммутабельность данных:
  - Кортежи и строки в Python являются неизменяемыми объектами, что обеспечивает иммутабельность данных.
- Функции как объекты 1-го класса:
  - Функции могут быть присвоены переменным, переданы в качестве аргументов или возвращены из других функций.
- Функции высших порядков:
  - Функции могут принимать другие функции в качестве аргументов или возвращать функции.
- Встроенные функции высших порядков для работы с последовательностями:
  - map: применяет функцию к каждому элементу последовательности
  - filter: фильтрует последовательность на основе условия, заданного функцией

4. Важнейшие функции для работы с потоками ввода/вывода, строками, регулярными выражениями

- Функции для работы с потоками ввода/вывода:
  - input(): считывает строку с клавиатуры
  - print(): выводит данные на экран
  - open(): открывает файл для чтения или записи

- Функции для работы со строками:
  - len(): возвращает длину строки
- Методы строк:
  - split(): разделяет строку на подстроки по заданному разделителю
  - join(): объединяет подстроки в одну строку с заданным разделителем
  - upper(): преобразует все символы строки в верхний регистр
  - lower(): преобразует все символы строки в нижний регистр
- Регулярные выражения(для работы с регулярными выражениями в Python есть модуль re):
  - re.match(pattern, string): ищет соответствие регулярному выражению в начале строки
  - re.fullmatch(pattern, string): проверить, подходит ли строка string под шаблон pattern
  - re.search(pattern, string): найти в строке string первую строчку, подходящую под шаблон pattern
  - re.findall(pattern, string): найти в строке string все непересекающиеся шаблоны pattern
  - re.split(pattern, string, maxsplit=0): аналог str.split(), только разделение происходит по подстрокам, подходящим под шаблон pattern
  - re.finditer(pattern, string): итератор по всем непересекающимся шаблонам pattern в строке string (выдаются match-объекты)
  - re.sub(pattern, repl, string, count=0): заменить в строке string все непересекающиеся шаблоны pattern на repl