Программирование, лекция 16. Регулярные выражения, модуль re. Модули sys, functools, itercools, collections и др.

Кафедра ИУ7 МГТУ им. Н. Э. Баумана,
2022 год

Декомпозиция программных задач

Декомпозиция - разделение задачи на множество частных задач, не превосходящих по совокупной сложности исходную.

Один из способов декомпозиции - использование подпрограмм.

Подпрограммы необходимо использовать для:

- уменьшения дублирования кода
- возможности повторного использования кода в других программах
- упрощения отладки

Регулярные выражения

Регулярные выражения - формальный язык поиска и модификации подстрок в тексте, основанный на использовании метасимволов.

Основные специальные символы:

- . - любой символ (кроме перевода строки)
- ^ - начало строки
- $ - конец строки
- * - любое количество вхождений, включая 0
- + - любое количество вхождений > 0
- ? - 0 или 1 вхождений

Специальные символы

- {m} - m вхождений
- {m, n} - от m до n вхождений
- \ - экранирование символа (escape-последовательность)
- [] - набор символов
  - [aeiouy]
  - “-” - диапазон символов [A-F]
  - ^ - инверсия набора [^.!?] (только при включении первым символом)
- | - выбор из двух выражений
- () - группировка подвыражений

Специальные символы

- \d - цифра
- \D - не цифра
- \s - пробельный символ (\t, \n, \r, …)
- \S - непробельный символ
- \w - буквенно-цифровой символ
- \W - не буквенно-цифровой символ

“Ленивый” и “жадный” поиск

“Жадный” (greedy) режим поиска найдёт первый самый длинный фрагмент текста, удовлетворяющий выражению

“Ленивый” (non-greedy) режим поиска найдёт первый самый короткий фрагмент

Например, для текста “abc defg” и выражения “\w+”:

- жадный - abc
- ленивый - a

Для включения “ленивого” режима требуется добавлять “?”: *?, +?, ??, {m,n}?

Разновидности регулярных выражений

- базовые регулярные выражения POSIX (Portable Operating System Interface - стандарт совместимости UNIX-подобных ОС)
  - требуется экранировать {, }, (,)
  - отсутствуют +, ?, |
- расширенные регулярные выражения POSIX
- регулярные выражения, совместимые с Perl (PCRE, Perl Compatible Regular Expressions)

Модуль re

Функции:

- compile(pattern, flags=0) - “скомпилировать” регулярное выражение в объект для ускорения последующей работы
- search(pattern, string, flags=0) - найти первое соответствие
- match(pattern, string, flags=0) - проверить начало строки на соответствие
- fullmatch(pattern, string, flags=0) - проверить всю строку на соответствие
- split(pattern, string, maxsplit=0, flags=0) - возвращает список строк по разделителю
- findall(pattern, string, flags=0) - возвращает список неперекрывающихся совпадений
- finditer(pattern, string, flags=0) - возвращает итератор
- sub(pattern, repl, string, count=0, flags=0) - заменяет совпадения
- subn(pattern, repl, string, count=0, flags=0) - заменяет вхождения и возвращает количество замен
- escape(pattern) - экранировать специальные символы
- purge() - очистить кэш

Методы и свойства класса Match

expand(template)

group(number)

groups()

groupdict()

start(group)

end(group)

span(group)

pos

endpos

lastindex

pastgroup

re

string

Модуль functools

Модуль для работы с функциями высших порядков
Функции:

- cache() - декоратор для ускорения повторных вызовов
- lru_cache(max_size) - декоратор с ограничением числа сохранённых вызовов
- total_ordering() - декоратор для классов, дополняющий методы сравнения
- partial() - создаёт новую функцию на основе существующей, “замораживая” часть параметров
- reduce() - накапливает результат для функции двух аргументов
- singledispatch() - позволяет определять поведение функции в зависимости от типа аргумента

Модуль itertools

Набор инструментов для построения итераторов
Функции:
count(start[, step]) - бесконечный счётчик, начиная с параметра
cycle(p) - обход элементов в бесконечном цикле
repeat(elem[, n]) - повтор элемента n раз
accumulate(p) - накопление суммы
chain(p, q, …) - обход всех последовательностей по очереди
Комбинаторика: combinations(), combinations_with_replacement(), permutations(), product()
compress(data, selectors) - итерирование только по “выбранным” элементам
groupby(iterable[, key]) - группировка по ключу

Модуль collections

Специализированные типы данных для коллекций, дополняющие встроенные dict, list, set, tuple:

- ChainMap - связывание нескольких словарей
- Counter - счётчик различных значений в словаре
- deque - двухсторонняя очередь
- defaultdict
- namedtuple()

Модуль operator

Содержит реализацию всех операторов в виде функций.

Дополнительно:

attrgetter()

itemgetter()

methodcaller()

Модуль sys

- argv - список параметров командной строки
- base_exec_prefix, base_prefix, prefix
- byteorder - порядок байт в системе
- executable - путь к интерпретатору
- flags - флаги интерпретатора
- float_info - свойства типа float
- getsizeof(object)
- modules - словарь загруженных модулей
- stdin, stdout, stderr - файловые объекты потоков ввода-вывода
- version - версия интерпретатора

Завершение программы

Функции quit(), exit(), sys.exit(), os._exit()

quit() и exit() использовать **не рекомендуется**, поскольку они добавляются в пространство имён модулем site, который подключается по умолчанию, но может отсутствовать.

Модуль argparse

Предназначен для разбора аргументов командной строки

Модуль timeit

- timeit()
- repeat()
- Timer

Модули marshal, pickle, shelve

Модули предназначены для сериализации объектов Python в бинарные структуры с целью сохранения на диск либо передачи на другой компьютер.