Коллекции в Python

Дербышева Т.Н. МФТИ 2018

Типы коллекций

Коллекция	Тип	Пример	Пустая	
Список	list	[1, 8, -2]	[ ]	a[i]
Кортеж (неизм)	tuple	(1, 8, -2)	( )	x, y = y, x
Множество	set	{1, 2, 2, 1, 1}		уникальные
Словарь	dict	{'abc':2, 7:'xy'}	{ }	d['abc'] = 'q'
Строка	str	"abc"	""	неизм.

Последовательности

- Тип данных, где есть:
  - **in** - проверка есть элемент в последовательности
  - **[ ]** - срезы
  - **for** - перебор элементов в цикле
- s = "Hello"
  if "el" **in** s:
  print("el is substring of Hello")
- print (s **[1 : 3]**) # el
- **for** c **in** "Hello":
  print(c)

Типы последовательностей

- Изменяемые (list)
  - set, dict - хэш-таблицы - нет операции [ ]
- Неизменяемые (str, tuple - кортеж)

- a = [7, "Hello", 3.14, [44, 55]] # список
- t = (7, "Hello", 3.14, [44, 55]) # кортеж
- s = "Python" # строка

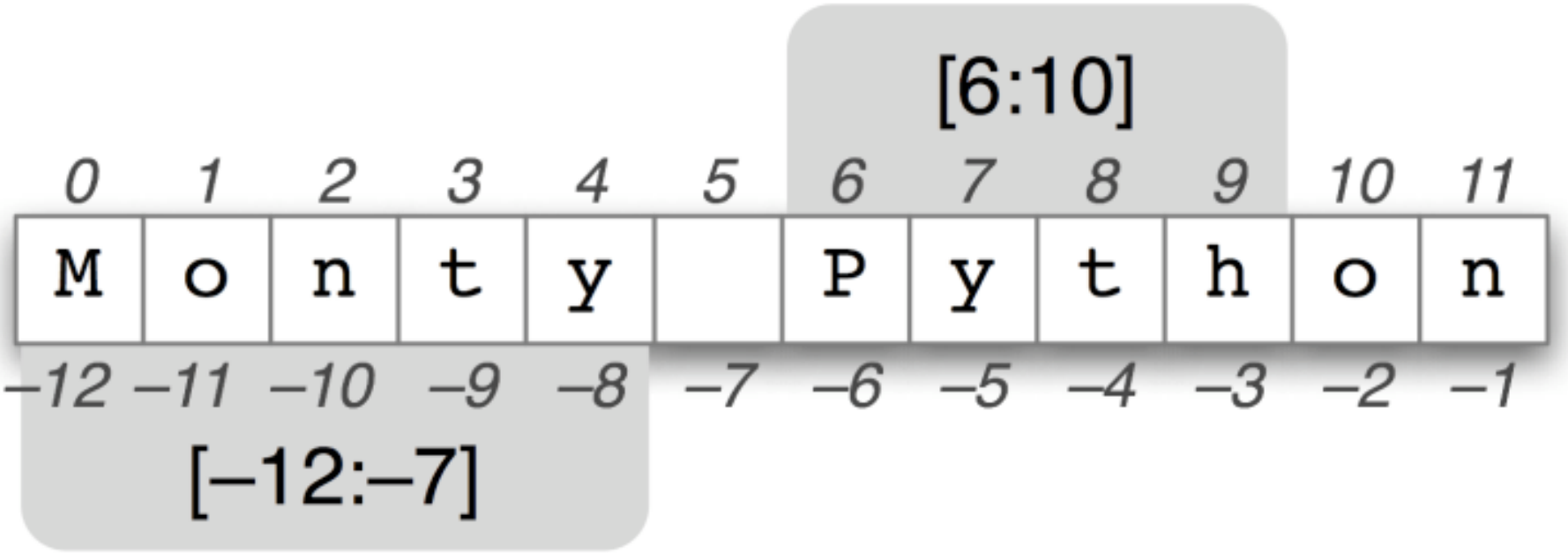
Операции над неизменяемыми 1

- s = "Python"
  a = [5, 3, -2, 10]
- "th" **in** s
  "th" **not in** s
- 10 **in** a
  66 **not in** a
- "abc" + 'xy' # "abcxy"
- [1, 2, 3] + [4, 5] # [1, 2, 3, 4, 5]

Операции над неизменяемыми 2

- "hi" * 3 # "hihihi"
  3 * "hi"
- [1, 2] * 3 # [1, 2, 1, 2, 1, 2]
  3 * [1, 2]
- a = [5, 3, -2, 10]
- min(a) # -2
- max(a) # 10
- sum(a) # 13

Индексы и срезы (slice) [от : до : шаг]

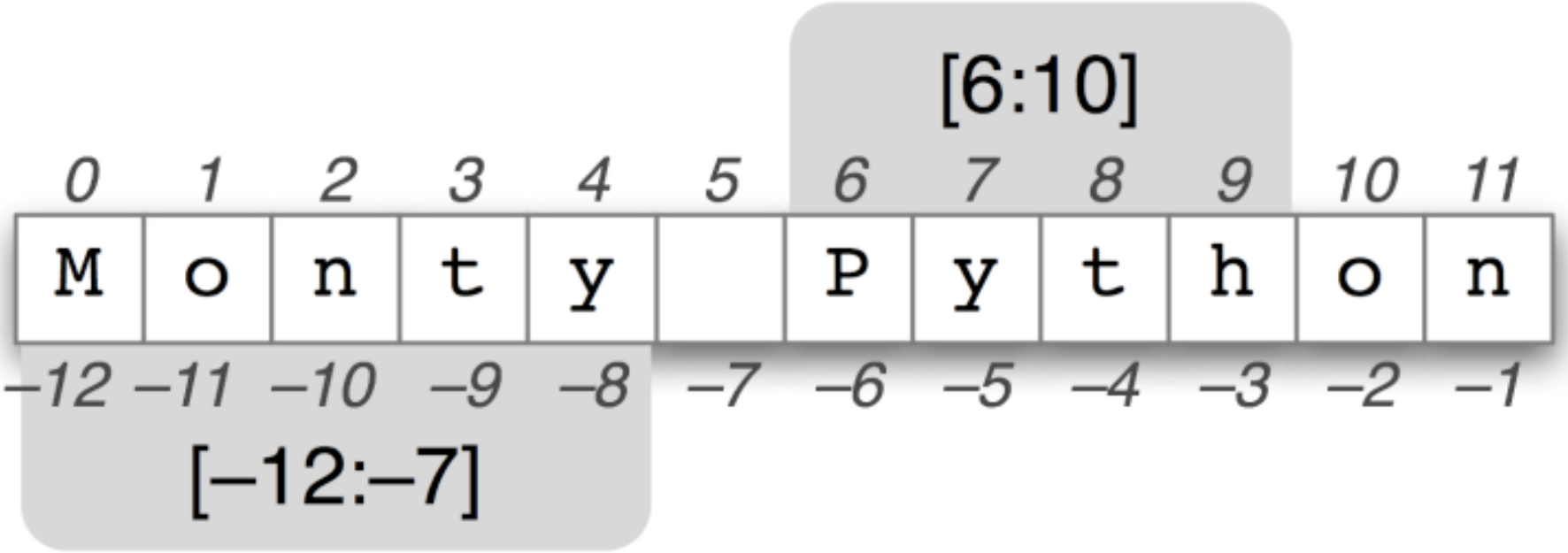


- s [2]
- s [-2]
- s [6 : 10]
- s [-12 : -7]

- s [: 2]
- s [6 :]
- s [:]

- s [2 : 10 : 2]
- s [2 : : 3]
- s [: : 2]
- s [: : -1]

Поиск



- **len** ("abc")
- s. **count** ("y")
- s.**index** ("y")
- s.index ("y", 5)
- s.index ("y", 5, 7)

Изменяемые последовательности - 1

- a = [5, 7, -2, 10]
- a [1] = 9 # [5, 9, -2, 10]
- a [2 : 3] = [1, 2, 3] # [5, 9, 1, 2, 3, 10]
- **del** a [2 : 4] # [5, 9, 3, 10]
- a [1 : 3] = [ ] # [5, 10]
- a. **append** (8) # [5, 10, 8]
- a.**insert** (1, 7) # [5, 7, 10, 8]
- a.**extend** ([4, 6]) # [5, 7, 10, 8, 4, 6]

append vs extend

- b0 = [1, 2] + [3, 4] # новый список
- b1 = [1, 2]
  - b1 += [3, 4] # тот же список
  - b1 . extend([3, 4])
- b1 *= 3
- [1, 2].append ([3, 4]) # [1, 2, [3, 4]]
- [1, 2].extend ([3, 4]) # [1, 2, 3, 4

Изменяемые последовательности - 3

- a.**clear**( ) # del a [:]
- a.**remove** (x) # ValueError
- x = a.**pop**( )
- x = a.**pop**(i)
-
- a.**reverse**( ) # a = a [: : -1]

Ссылки и копии

- a = [1, 2]
- b = a
- b [0] = 7
- print (a) # [7, 2]
- b1 = **list** (a)
- b2 = a [:]
- b3 = a.**copy**( )

shall & deep copy

- a = [[1, 2], [3, 4] , 7]
  b = a.copy( )
  b [2] = -1

  b [0][0] = 100

```
[[1, 2], [3, 4], 7]     # a
[[1, 2], [3, 4], -1]    # b

[[100, 2], [3, 4], 7]     # a
[[100, 2], [3, 4], -1]    # b
```

- import copy
  b1 = copy.copy (a)
  b2 = copy.deepcopy (a)

Перебираем элементы

- **for** c **in** "xyz" **:**
  print(c)
- **for** i, c **in enumerate** ("xyz") **:**
  print(i, c)
- 0 x
  1 y
  2 z

Некоторые методы строк

- str. isalnum ( )
- str. isalpha ( )
- str. isdecimal ( )
- str. isdigit ( )
- str. isidentifier ( )
- ...
- str. islower ( )
- str. isnumeric ( )
- str. isprintable ( )
- str. isspace ( )
- str. istitle ( )
- str. isupper ( )

strip

- str.**lstrip** ( )
- str.**rstrip** ( )
- str.**strip** ( )
- ' spacious '.strip ( ) # 'spacious'
- 'www.example.com'.strip('cmowz.') # 'example'
- comment_string = '#..... Sec. 3.2.1 Issue #32 ....'
- comment_string.strip('.#! ')
  получим 'Sec. 3.2.1 Issue #32'

подстрока в строке

- "el" **in** "hello"
- s.**startswith** ("c:")
- s.**endswith** (".txt")
- индекс подстроки:
- s.**index** ("el") # ValueError
- s.**find** ("el") # -1
- s.**replace** (old, new, *count*)

Сборка и разборка

- s = '5 7 -2'
- a = s.**split**( ) # a = ['5', '7', '-2']
- s1 = ' '.**join** (a) # s1 = '5 7 -2'
- s = '5;7;-2'
- a = s.split (';') # a = ['5', '7', '-2']
- a = list(**map**(int, a)) # a = [5, 7, -2]
- s1 = ';'.join (map(str, a)) # 5;7;-2

Оно пустое?

- Неправильно:
- if len(s) == 0:
  print('Empty string')
- Правильно:
- if s:
  print('НЕ пустая строка, список')
- if not s:
  print('пустая строка, список')

List comprehensions - делаем список

- a = [5, 7, -2, 10]
- b1 = **[** x*x **for** x **in** range(5) **]**
- b2 = [x*x for x in a]
  # [25, 49, 4, 100]
- b3 = **[** x*x **for** x **in** a **if** x > 0 **]**
  # [25, 49, 100]
- q = [ ] q = list ( ) - пустой список

Кортежи - неизменяемые списки

- t = tuple( )
- t = tuple([1, 3, 7])
- t = 1, 3, 7
- x, y = y, x
- Зачем?
  - функция возвращает "несколько значений"
  - быстрее
  - ключи в словарях

Множество - уникальные элементы

- b = {1 , 2, 3, 1, 1, 2, 5} # {1, 2, 3, 5}
- b = set([1, 2, 3, 1, 2, 5])
- Пустое множество:
  - b = set ( )
  - b = { } - нельзя, это НЕ множество
- произвольный порядок перебора
- нет операций [ ]

Множество set

- b = [ ]
  for x in a:
      if x not in b: # O(len(b))
          b.append(x)

- b = set (a)
  if x in b # O(1)
- Операции над множествами

Словарь dict

- Ключ - значение
- ключ - хэшируемый (как в set)
  - неизменяемые - хэшируемые
- значение - любое
- d = {'Russia' : 'Moscow',
  'Ukraine' : 'Kiev',
  'China' : 'Beijing'}
- d = { } или d = dict( ) - пустой

Оценки на зачете

- d = {"Mike":9, "Igor":7, "Alex":2}
- print(d["Mike"]) # 9
- d["Alex"] = 5 # исправил
- d["Olga"] = 8 # новая пара
- {"Mike" : 9,
  "Igor":7,
  "Alex":5
  "Olga":8 }

Ключа нет в словаре

- d['Kolobok'] = 6
- print(d['Sauron']) # KeyError Sauron
- if 'Sauron' in d:
    d['Sauron'] += 1
- d.get('Sauron', 2) # оценка за зачет

Перебираем словарь

- d = {'a' : 1, 'b' : 2, 'c' : 3}
- for k in d: # ['a', 'b', 'c']
- for k in d.**keys**( ): # ['a', 'b', 'c']
- for v in d.**values**( ): # [1, 2, 3]
- for k, v in d.**items**( ):
  # [('a', 1), ('b', 2), ('c', 3)]
- Порядок - ?, может меняться

Прикладное применение

- id клиентов при покупке в магазине
  a = [1, 3, 10, 3, 1, 2, 5, 7, 3, 2]
- set(a) - все клиенты без повторений
- d = dict( )
  for x in a:
  d[x] = d.get(x, 0) + 1

  # d - словарь:
  номер клиента - сколько раз покупал

Куда дальше?

- Генераторы ("отложенные вычисления")
- Сортировка (+ lambda функции)
- itertools
- и т.д

Вопросы?

Спасибо за внимание.