Ввод и вывод

Технологии и языки программирования

Юдинцев В. В.

Кафедра теоретической механики
Самарский университет

1 декабря 2017 г.

Содержание

Форматированный вывод

Функция print

- Функция print выводит информацию в стандартное устройство вывода (`sys.stdout`)

```
print(value, ..., sep=' ', end='\n')
```

- Все позиционные аргументы воспринимаются функцией как имена, содержимое которых необходимо напечатать:

```
print(1, 2, 3, 'Hello!')
```

```
>> 1 2 3 Hello!
```

- Дополнительные параметры передаются именованными аргументами `sep`, `end`, `file`, ...

Функция print

sep – символы-разделители выводимых значений (пробел):

```
1 print(1, 2, 3, 'Hello!', sep='::')
```

```
1::2::3::Hello!
```

end – последний символ (перевод строки):

```
1 print(1)
2 print(2)
```

```
1
2
```

```
1 print(1, end=', ')
2 print(2)
```

```
1, 2
```

Метод format

Метод `format` объекта-строки позволяет выполнять подстановку значений в строковые константы при помощи фигурных скобок:

```
1 from math import sqrt
2 a, b = 2.0, 3.0
3 S = sqrt(a**2+b**2)
4
5 mes="Гипотенуза треугольника с катетами {} и {} равна {}"
6
7 print( mes.format(a, b, S) )
```

```
Гипотенуза треугольника с катетами 2.0 и 3.0 равна 3.60555127546
```

Метод format: позиция аргумента

В фигурных скобках может быть указан порядковый номер аргумента функции `format`:

```
1 a, b = 10, 5
2
3 mes="{0}+{1}={2}, но {0}-{1}={3}"
4
5 print( mes.format(a, b, a+b, a-b) )
```

```
10-5=15, но 10-5=5
```

Метод format: имя аргумента

В фигурных скобках может быть указан именованный аргумент функции `format`:

```
a, b = 10, 5
mes="{arg1} + {arg2} = {res}"

print(mes.format(arg1 = a, arg2 = b, res = a + b))
```

```
10 + 5 = 5
```

Именованные или позиционные аргументы могут быть коллекциями, для которых в методе `format` можно указывать индекс элемента:

```
a = [1, 3, 5, 7, 9, 11]
print( 'First: {0[0]}, second: {0[1]}'.format(a) )
```

```
First: 1, second: 3
```

Метод format: имя аргумента

Можно смешивать обе записи, но позиционные аргументы должны быть первыми:

```
1 a, b = 10, 5
2 mes="{arg1} + {} = {res}"
3
4 print(mes.format(b, arg1 = a, res = a + b))
```

Спецификация форматирования

После двоеточия можно указать спецификации форматирования: ширину поля, выравнивание, тип, точность:

- Вывод двух целых чисел в поле шириной 10 символов и 5 символов с выравниванием по правому краю и по левому краю:

```
1 print( '{0:10d}, {1:<5d}'.format(15,25) )
```

```
        15, 25
```

- Превое число выравнивается по центру, второе – по правому:

```
1 print( '{0:^10d}, {1:>10d}'.format(15,25) )
```

```
    15     ,         25
```

Спецификация форматирования

- Печать вещественного числа с заданным числом знаков после запятой:

```
print('{0:10.3f}, {0:<10.2f}'.format(12.11254)
    )
```

```
    12.113, 12.11
```

- Обязательный вывод знака числа

```
print('{0:+10.3f}'.format(11.3))
print('{0:-10.3f}'.format(11.3))
print('{0:-10.3f}'.format(-12.3))
```

```
   +11.300
    11.300
   -12.300
```

Спецификация форматирования

Символ-заполнитель:

```
1 print('{0:_^10.3f}'.format(12.11254))
2 print('{0:_<10.2f}'.format(12.11254))
```

```
1 __12.113__
2 12.11_____
```

Ввод

Функция input

```
user_name = input('Введите Ваше имя: ')
print('Здравствуйте,', user_name)
```

- Функция `input` возвращает введённое значение в виде строки
- Для преобразования в другие типы, например `float`, необходимо использовать функции преобразования:

```
value = float(input('Введение значение аргумента: '))
print(value)
```

Чтение и запись файла

Открытие файла

Для открытия файла используется функция open:

```
1 f = open('data_file.txt', 'r')
```

- Первый аргумент – имя файла
- Второй аргумент (одна или две буквы): режим работы с файлом и его тип
  r – чтение
  w – запись с заменой содержимого существующего файла
  x – создание файла и запись только если файла не существует
  a – добавление данных в файл
  r+ – чтение и запись
  t – текстовый файл (по умолчанию)
  b – двоичный файл

Запись в файл: функция write

Запись в текстовый файл:

```
1 message = 'Live long and prosper!'
2 f = open('message.txt', 'wt')
3 f.write(message)
4 f.close()
```

Функция `write` возвращает количество записанных символов.

Запись в файл: функция print

Для записи можно использовать функцию `print`, передав ссылку на файл в именованном аргументе `file`:

```
1 f = open('message.txt', 'wt')
2 print('a', 'b', 'c', file=f)
3 f.close()
```

Функция `print` по умолчанию добавляет пробел после каждого записанного элемента (`'a b c'`) и заканчивает запись символом перевода строки.

Чтение из файла: функция read

Чтение всех данных из файла:

```
1 f = open('results.txt', 'rt')
2 data = f.read()
3 f.close()
```

Чтение блоками по 100 символов:

```
1 f = open('results.txt', 'rt')
2 data=''
3 block_size = 100
4 while True:
5     block = f.read(block_size)
6     if not block:
7         break
8     data += block
9 f.close()
```

Чтение из файла: функция readline

Для построчного чтения используется метод `readline`:

```
1 f = open('results.txt', 'rt' )
2 data=''
3 while True:
4     line = f.readline()
5     if not line:
6         break
7     data += line
8 f.close()
```

Чтение из файла: файл как итератор

```
1 f = open('results.txt', 'rt' )
2 data=''
3 for line in f
4     data += line
5 f.close()
```

Чтение бинарных данных

```
1 f = open('data.obj', 'rb')
2 bdata = f.read()
3 print(type(bdata))
4 f.close()
```

```
<class 'bytes'>
```

Запись бинарных данных

```
1 data = bytes([0,0,0,1,2,3])
2 f = open('data.obj', 'wb')
3 f.write(bdata)
4 f.close()
```

Бинарные и текстовые файлы

```
1 data=bytes([0,1,2,3])
2 f=open('dat.bin', 'wb')
3 f.write(data)
4 f.close()
```

Содержимое файла `dat.bin` в шестнадцатеричном формате:

```
00 01 02 03
```

Содержимое файла `dat.bin` в текстовом формате:

```
NUL SOH STX ETX
```

```
1 data = '0123'
2 f=open('dat.txt', 'wt')
3 f.write(data)
4 f.close()
```

Содержимое файла `dat.txt` в шестнадцатеричном формате:

```
30 31 32 33
```

Содержимое файла `dat.txt` в текстовом формате:

```
0123
```

Автоматическое закрытие файла

Для автоматического закрытия файла можно использовать следующий шаблон-блок (`with` ... `as` ... `:`):

```
1 data = '0123'
2 with open('dat.txt', 'wt') as f:
3     f.write(data)
```

Перемещение по файлу

- Открытый файл можно представить в виде ленты, с которой считываются данные (символы, байты) или на которую ведётся запись.
- Для отслеживания и изменения текущей позиции внутри файла используется методы `tell` и `seek`:

```
1 f = open('data.txt', 'rt')
2 print( f.tell() )
```

```
0
```

После открытия файла текущая позиция указывает на начало файла: 0 байт от начала файла

tell и seek

- Открытие файла и чтение первых 100 байт:

```
1 f = open('data.bin', 'rb')
2 f.read(100)
3 print( f.tell() )
```

```
100
```

- Текущая указывает на следующий не прочитанный байт файла: сотый байт от начала
- Перемещение указателя в позицию 50 байт от начала:

```
1 f.seek(50)
```

Чтение текстового файла

```
with open('ru.txt','rt', encoding="utf-8") as f:
    x = f.read()
    print(x)
    print(f.tell())
```

Файл ru.txt

```
sputnik
```

Результат

```
sputnik
7
```

Файл ru.txt

```
спутник
```

Результат

```
спутник
14
```

Задание

Форматированный вывод

Ипользуя метод format, напишите программу, выводящую на экран таблицу умножения.

```
2 * 1 = 2       3 * 1 = 3
...             ..

4 * 1 = 4       5 * 1 = 5
...             ..

6 * 1 = 6       7 * 1 = 7
...             ..

8 * 1 = 8       9 * 1 = 9
...             ..
```

Транслитерация

- Загрузить произвольный файл с текстом на русском языке и заменить все русские буквы по правилам транслитерации, сохраняя регистр букв (верхний или нижний).
- Правила преобразования должны загружатсья из другого текстового файла, каждая строка которого содержит русскую букву и её представление латиницей:

```
б:b
в:v
г:g
...
ж:zh
```

- Преобразованный вариант необходимо сохранить в текстовый файл

Пример: загрузка словаря

```
1 def load_dictionary(file_name):
2   with open(file_name,'rt',encoding="utf-8") as f:
3     lines = f.read().splitlines()
4   rep = dict()
5   for line in lines:
6     pair=line.split(':')
7     rep.update({pair[0]:pair[1]})
8   return rep
```