УДК 004

Пример удобного вывода таблиц в консоль на языке Python

Жуков Дмитрий Сергеевич
Приамурский государственный университет им. Шолом-Алейхема
Студент

Аннотация
В данной статье показан пример вывода таблиц в консоль. Красивый и удобный вывод таблиц реализован при помощи библиотеки Prettytable на языке Python. Этот метод полезен при создании консольной программы или скриптов, например для серверов.
Ключевые слова: python, таблица, консоль, программа.

An example of convenient outputting tables to the console in Python

Zhukov Dmitry Sergeevich
Sholom-Aleichem Priamursky State University
Student

Abstract
This article shows an example of outputting tables to the console. Beautiful and convenient output of tables is implemented using the Prettytable library in Python. This method is useful when creating a console program or scripts, for example, for servers.
Keywords: python, table, console, program.

В современном мире существует множество программ и скриптов, не использующих интерфейс. Такие программы выводят всю необходимую пользователю информацию в консоль. Иногда предоставить информацию о разных таблицах пользователю, например таблица базы данных. Для удобства таблица должна быть легко воспринимаемой для пользователя, примеры вывода таких таблиц будут показаны в этой статье.

В статье И.Кутепова рассматривается пример построения вычислителя на основе нечеткой логики и применение языка Python для написания интегрированной среды разработки [2]. И.Е.Бронштейн в своей статье описал вывод типов для программного кода на языке Python. Сначала производится обзор описанных в научной литературе алгоритмов вывода типов для языков с параметрическим полиморфизмом. Затем даётся описание нового алгоритма, являющегося модификацией одного из предыдущих: алгоритма декартова произведения. Показывается, как модуль вывода типов, использующий новый алгоритм, анализирует различные конструкции языка Python. Представляются результаты работы над прототипом [3]. В работе В.В.Найденова протестирована производительность пары аналогичных

приложений, реализующих CRUD логику с помощью прослойки ORM. Сравнивается SQLAlchemy – де-факто стандартный ORM для Python с динамическим ORM для C++ собственной разработки – YB.ORM. Сравнивается производительность при использовании CPython и PyPy. Проверяется влияние отключения логов на производительность [4]. В статье И.Е.Бронштейн рассматриваются виды дефектов, которые обычно встречаются в программном коде на языке Python. Показывается, что возможные дефекты для Python не похожи на те, что часто встречаются в коде на Си/Си++ и, следовательно, необходимо исследование дефектов в крупных проектах с открытым исходным кодом. Даётся классификация найденных дефектов на основе того, нужен ли для нахождения ошибки вывод типов. Показывается, что существует небольшая доля "простых" дефектов, но для обнаружения большинства дефектов вывод типов необходим. Рассматривается вопрос, какие конструкции языка Python должны поддерживаться при выводе типов для нахождения реальных дефектов [5]. В работе Д.А.Кузнецов рассматривается структура интерфейса программы «Фармацевтическая экономическая безопасность» для фармацевтических организаций. Описывается функциональное предназначение и возможности пунктов основного меню прикладной программы [6]. Ю.А.Котов, А.В.Шаповалов в своей статье рассмотрели интерфейс программной реализации экспертной системы для восстановления простой замены букв текста. Описаны базовые элементы интерфейса, включающие выбор функциональных методов и базовых операций (замена, сдвиг, перебор). Не менее значимы иностранные исследования в данной сфере [8-9].

Разработку программы следует начать с загрузки нужной библиотеки. Нам понадобятся библиотека Prettytable для отрисовки таблицы.

Скачать библиотеку можно введя в консоль команду «pip install Prettytable».

Подключим все необходимые функции с этой библиотеки. Создаем объект класса prettytable. Добавляем название наших будущих столбцов в таблице. Добавляем строки таблицы функцией add_row параметром которой является список из значений каждого столбца. Выведем полученную таблицу командой print (Рис.1).

create_by_row.py

```
#!/usr/bin/python3

from prettytable import PrettyTable

x = PrettyTable()

x.field_names = ["City name", "Area", "Population", "Annual Rainfall"]

x.add_row(["Adelaide", 1295, 1158259, 600.5])
x.add_row(["Brisbane", 5905, 1857594, 1146.4])
x.add_row(["Darwin", 112, 120900, 1714.7])
x.add_row(["Hobart", 1357, 205556, 619.5])
x.add_row(["Sydney", 2058, 4336374, 1214.8])
x.add_row(["Melbourne", 1566, 3806092, 646.9])
x.add_row(["Perth", 5386, 1554769, 869.4])

print(x)
```

Рис.1.Создание и вывод таблицы

Результатом вывода является таблица оформленная с помощью символов ascii, которые выровнены между собой, формируя визуальный вид таблицы (рис.2).

```
+-----------+------+------------+-----------------+
| City name | Area | Population | Annual Rainfall |
+-----------+------+------------+-----------------+
|  Adelaide | 1295 |  1158259   |      600.5      |
|  Brisbane | 5905 |  1857594   |      1146.4     |
|   Darwin  | 112  |   120900   |      1714.7     |
|   Hobart  | 1357 |   205556   |      619.5      |
|   Sydney  | 2058 |  4336374   |      1214.8     |
| Melbourne | 1566 |  3806092   |      646.9      |
|   Perth   | 5386 |  1554769   |      869.4      |
+-----------+------+------------+-----------------+
```

Рис.2. Результат вывода таблицы

Ту же самую таблицу можно создать не построчно, а по столбцам. Для этого при заполнении таблицы вместо функции add_row используется add_column, параметрами которого являются все данные столбца. Пример показан на рисунке 3.

create_by_column.py

```
#!/usr/bin/python3

from prettytable import PrettyTable

x = PrettyTable()

column_names = ["City name", "Area", "Population", "Annual Rainfall"]

x.add_column(column_names[0], ["Adelaide", "Brisbane", "Darwin",
    "Hobart", "Sydney", "Melbourne", "Perth"])
x.add_column(column_names[1], [1295, 5905, 112, 1357, 2058, 1566, 5386 ])
x.add_column(column_names[2], [1158259, 1857594, 120900, 205556, 4336374,
    3806092, 1554769])
x.add_column(column_names[3], [600.5, 1146.4, 1714.7, 619.5, 1214.8,
    646.9, 869.4])

print(x)
```

Рис.3. Занесение данных в таблицу столбцами

После занесения данных в таблицу так же можно их удалить (Рис.4).

```
x.del_row(6)
x.del_row(5)
x.del_row(4)
x.del_row(3)
```

Рис.4. Удаление строк

Данная библиотека может формировать таблицу из CSV файлов. Для примера создадим файл data.csv и наполним его данными (Рис.5).

data.csv

```
"City name", "Area", "Population", "Annual Rainfall"
"Adelaide", 1295, 1158259, 600.5
"Brisbane", 5905, 1857594, 1146.4
"Darwin", 112, 120900, 1714.7
"Hobart", 1357, 205556, 619.5
"Sydney", 2058, 4336374, 1214.8
"Melbourne", 1566, 3806092, 646.9
"Perth", 5386, 1554769, 869.4
```

Рис.5. Файл данных CSV

Пример формирования таблицы из файла показан на рисунке 6. Для этого используется функция from_csv параметром которой является считанная строка из файла.

read_from_csv.py

```
#!/usr/bin/python3

from prettytable import from_csv

with open("data.csv", "r") as fp:
    x = from_csv(fp)

print(x)
```

Рис.6. Пример формирования таблицы из файла

Для того что бы отсортировать созданную ранее таблицу нужно указать переменную sortby название столбца, который мы хотим отсортировать (Рис.7).

```
print("Table sorted by population:")
x.sortby = "Population"
print(x)
```

Рис .7. Сортировка по столбцу

Результат данной сортировки показан на рисунке 8. На нем отображена таблица сортированная по столбцу Population.

```
Table sorted by population:
+-----------+------+------------+-----------------+
| City name | Area | Population | Annual Rainfall |
+-----------+------+------------+-----------------+
|   Darwin  | 112  |   120900   |      1714.7     |
|   Hobart  | 1357 |   205556   |      619.5      |
|  Adelaide | 1295 |  1158259   |      600.5      |
|   Perth   | 5386 |  1554769   |      869.4      |
|  Brisbane | 5905 |  1857594   |      1146.4     |
| Melbourne | 1566 |  3806092   |      646.9      |
|   Sydney  | 2058 |  4336374   |      1214.8     |
+-----------+------+------------+-----------------+
```

Рис. 8. Вывод на экран

Таким образом, продемонстрирована работа с библиотекой Prettytable позволяющая формировать, отображать, сортировать таблицы в консоли.

Библиографический список

1. Лутц М. Изучаем python. М., 2009.
2. Кутепов И. Применение языка python при проектировании нечеткого контроллера // Компоненты и технологии. 2013. № 8 (145). С. 148-154.
3. Бронштейн И.Е. Вывод типов для языка python // Труды Института

системного программирования РАН. 2013. Т. 24. С. 161-190.
4. Найденов В.В. Тестирование производительности orm в языках python и c++ // RSDN Magazine. 2014. № 1. С. 05-08.
5. Бронштейн И.Е. Исследование дефектов в коде программ на языке python // Программирование. 2013. Т. 39. № 6. С. 25-32.
6. Кузнецов Д.А. Интерфейс программы "фармацевтическая экономическая безопасность" // Российский медико-биологический вестник им. академика И.П. Павлова. 2009. № 2. С. 162-165.
7. Котов Ю.А., Шаповалов А.В. Интерфейс программы восстановления простой замены букв текста // Современные тенденции развития науки и технологий. 2016. № 4-4. С. 57-59.
8. Smith A. W. et al. Application program interface that enables communication for a network software platform : пат. 7117504 США. – 2006.
9. Parikh V., Moore R., Cheng H. Application program interface for a graphics system : пат. 6456290 США. – 2002.