2 Типы, переменные и основы ввода-вывода. Программа «Бесполезные факты»[1]

Теперь вы более или менее представляете себе, как сохранять и запускать программу. Настал час заглянуть поглубже и научиться чуть более сложным вещам. В этой главе я расскажу, какие есть способы категоризации и хранения данных в компьютере, а также (что, безусловно, важнее) как пользоваться данными в программах. Вы научитесь получать информацию от пользователя и таким образом создавать интерактивные приложения.

Итак, предстоит узнать:

- о применении строк в тройных кавычках (triple-quoted) и escape-последовательностей, которые дают больше функциональности в работе с текстом;
- о математических операциях в Python;
- о хранении данных в памяти компьютера;
- о переменных, с помощью которых можно получать доступ к данным и манипулировать ими;
- о пользовательском вводе и создании интерактивных программ.

Знакомство с программой «Бесполезные факты»

Освоив при изучении этой главы все нужные навыки, вы сумеете самостоятельно написать программу «Бесполезные факты», рабочее окно которой показано на рис. 2.1.

---

[1] В этой главе и далее мы переводим в образцах кода все английские высказывания (комментарии и текст, отображаемый на экране) на русский язык. В отличие от Python 2.x, Python 3.1 не требует в таких случаях включать русскую локаль, но в начале кода может понадобиться директива # coding: cp1251 или # coding: utf-8. *(Примеч. перев.)*

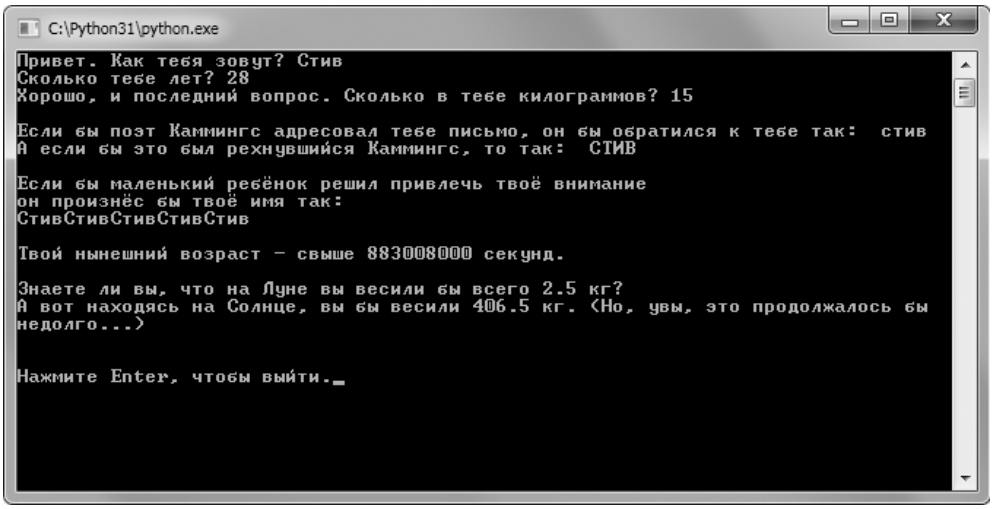

Рис. 2.1. Прежде чем посетить Солнце, 28-летнему Стиву не помешало бы сбросить лишний вес!

Из пользовательского ввода программа получает информацию об имени, возрасте пользователя и массе его тела. На основе этих нехитрых данных программа способна сгенерировать несколько забавных, но совершенно бесполезных фактов о пользователе: например, сколько тот будет весить на Луне.

На вид, как, впрочем, и внутренне, программа «Бесполезные факты» очень проста. Занимательность ей придает пользовательский ввод. Вы с нетерпением ждете момента, когда на экране появятся результаты, потому что они относятся лично к вам и ни к кому более. Большинство программ: от игр до бизнес-приложений — работают точно так же.

Строки и кавычки

В предыдущей главе приводился пример строки: `"Game Over"`. Но строковые данные могут быть намного сложнее и объемнее. Вообразим, например, что надо довести до сведения пользователя инструкцию из нескольких абзацев или показать на экране особым образом отформатированный текст. Все эти задачи способны решать строковые литералы в кавычках.

Знакомство с программой Game Over 2.0

Game Over 2.0 — улучшенная версия программы-предшественницы Game Over: теперь уведомление о том, что игра подошла к концу, отображается на экране более внушительно. Как работает эта программа, можно увидеть на рис. 2.2. Game Over 2.0 показывает, что с использованием кавычек текст можно оформлять разными способами. Код программы доступен на сайте-помощнике (www.courseptr.com/downloads) в папке Chapter 2; файл называется `game_over2.py`.

Рис. 2.2. Так бы сразу и сказали…

```
# Game Over - версия 2
# Показывает разные приемы работы со строковыми литералами
print("Программа 'Game Over' 2.0")
print("То же", "самое", "сообщение")
print("Только",
      "чуть-чуть",
      "побольше")
print("Вот", end=" ")
print("оно…")
print(
      """

       _____          ___          ___  ___  _____
      /  ___|      /   |      /   |/   | |  ___|
      | |         / /| |     / /|   /| | | |__
      | |  _     / ___ |    / / |__/ | | |  __|
      | |_| |  / /   | |   / /        | | | |___
      \_____/ /_/    |_| /_/         |_| |_____|

       _____   _     _   _____  _____
      /  _  \ | |   / / |  ___||  _  \
      | | | | | |  / /  | |__  | |_| |
      | | | | | | / /   |  __| |  _  /
      | |_| | | |/ /    | |___ | | \ \
      \_____/ |___/     |_____||_|  \_\
      """

    )
input("\n\nНажмите Enter, чтобы выйти.")
```

Кавычки внутри строк

Вы уже умеете создавать простейшие строковые литералы: для этого достаточно окружить текст парой кавычек, причем можно воспользоваться или апострофами

(''), или «лапками» (“”) — компьютеру все равно. Поэтому 'Game Over' — та же самая строка, что и “Game Over”. Но обратите внимание на код, в котором эта строка обнаруживается в нашей программе:

```
print(“Программа 'Game Over' 2.0”)
```

Здесь использованы кавычки обоих типов. А теперь вернитесь к рис. 2.2. На нем видны только апострофы, потому что они такая же часть строкового значения, как, например, буква G. Двойные кавычки не входят в строку. Они в каком-то роде аналог книжной обложки: компьютер по ним узнает, где строковый литерал начинается и где заканчивается. Итак, если в роли ограничителей у вас выступает пара кавычек-«лапок», то внутри ограниченной ими строки будет использовано сколько угодно апострофов. И наоборот: если строка ограничена апострофами, то в ее состав может входить сколько угодно «лапок».

Применив кавычки одного типа в роли ограничителей, вы уже не сможете пользоваться ими внутри строкового литерала. Это целесообразно, ведь второе по порядку вхождение открывающей кавычки компьютер считает концом строки. Так, например, “Словами 'Хьюстон, у нас проблемы' Джим Ловелл вошел в историю американской астронавтики” — это корректная строка. Напротив, “Словами “Хьюстон, у нас проблемы” Джим Ловелл вошел в историю американской астронавтики” — некорректная строка. Как только компьютер видит вторую по счету кавычку-«лапку», он считает, что строка закончилась, и распознает в качестве таковой лишь “Словами “. Вслед за тем упоминается Хьюстон — не служебное слово и не имя переменной. Компьютер даже не подозревает, что это за Хьюстон, и, конечно, выдаст ошибку.

Вывод на экран нескольких значений

Печатать на экране несколько значений можно одним вызовом функции print(). Для этого достаточно задать аргументы списком, перечислив их через запятую. Именно так сделано в коде нашей программы:

```
print(“То же”, “самое”, “сообщение”)
```

Здесь функции передаются три аргумента: “То же”, “самое” и “сообщение”. Этот код выводит на экран слова То же самое сообщение. Заметьте, что в роли разделителя между значениями на экране выступает пробел. Функция print() по умолчанию ведет себя именно так.

Аргументы, заданные списком, вы можете разбить на несколько строк кода так, чтобы эти строки (кроме последней) заканчивались разделителем-запятой. Следующий код образует единое выражение, которое выводит на экран одну текстовую строку Только чуть-чуть побольше. Я писал каждый следующий аргумент после запятой с новой строки:

```
print(“Только”,
      “чуть-чуть”,
      “побольше”)
```

Иногда полезно так делать, чтобы код было легче читать.

Вывод заключительных символов строки

По умолчанию функция `print()` завершает выводимый на экран текст переходом на новую строку. Это значит, что вызов `print()` в следующий раз заставит дальнейший текст отобразиться на экране ниже. Этого, в общем-то, мы и хотим, но не помешала бы также возможность вручную назначать заключительный символ или символы. Можно было бы, например, вместо новой строки определить пробел как концевой символ при вызове функции `print()`. Это означало бы, что при вызове `print()` в следующий раз печатаемые значения отобразятся сразу после пробела. Данная функциональность используется в коде нашей программы:

```
print("Вот", end=" ")
print("оно…")
```

Текст `Вот оно…` печатается, таким образом, в одну строку. Это потому, что в первой команде `print()` в качестве заключительного символа выбран пробел. Компьютер печатает на экране `Вот` (с пробелом после «т») и не переходит на новую строку. Следующая команда `print()` начинает выводить текст `оно…` сразу после пробела, следующего за «т». Легко понять, что такого эффекта я достиг, сделав пробел значением параметра `end` в функции `print(): end=" "`. Создавая собственные выражения с `print()`, вы можете тем же способом назначать любые строковые литералы в качестве заключительных печатаемых символов: запятая, имя параметра `end`, знак равенства, собственно литерал. Теперь вы можете гораздо более гибко форматировать выводимый текст.

ПОДСКАЗКА

Если вам пока неясно, что такое параметр, не беспокойтесь. О параметрах и присвоении им значений вы сможете подробно прочесть в главе 6, в разделе «Параметры и возвращаемые значения».

Создание строк в тройных кавычках

Интереснее всего в новой программе, конечно, то, что она печатает `Game Over` огромными псевдографическими буквами. За это ответственна в коде строка:

```
    """
     _____      ___       ___   ___  _____
    /  ___|     /   |     /   |/   | |  ___|
    | |        / /| |    / /|   /| | | |__
    | |  _    / ___ |   / / |__/ | | |  __|
    | |_| |  / /  | |  / /       | | | |___
    \_____/ /_/   |_| /_/        |_| |_____|

     _____   _     _  _____  _____
    /  _  \ | |   / / |  ___||  _  \
    | | | | | |  / /  | |__  | |_| |
    | | | | | | / /   |  __| |  _  /
    | |_| | | |/ /    | |___ | | \ \
    \_____/ |___/     |_____||_|  \_\
    """
```

Вот что называется *строкой в тройных кавычках*. Это строковый литерал, ограниченный парой строенных кавычек. Как и ранее, неважно, пользуетесь вы «лапками»

или апострофами; надо следить лишь за тем, чтобы открывающие и закрывающие кавычки были одинаковыми.

Как видите, строковый литерал в тройных кавычках может занимать в коде несколько строк и выводится на экране точно в таком же виде, как и вводится.

НА САМОМ ДЕЛЕ

Если вам по душе составленные из отдельных символов огромные буквы, как в Game Over 2.0, то вам понравится и вся так называемая ASCII-графика (ASCII Art). Это, собственно говоря, картинки, составляемые из символов, которые доступны на клавиатуре. ASCII расшифровывается как American Standard Code for Information Interchange (Американский стандартный код для информационного обмена). В этом коде доступно представление 128 стандартных символов. Между прочим, графика на основе букв — не новое изобретение. Она появилась задолго до компьютера: первый доказанный факт выполнения графики на пишущей машинке датируется 1898 годом.

Escape-последовательности в строках

Escape-последовательности (или экранированные последовательности) позволяют вставлять в строки специальные символы. С помощью этого инструмента вы достигнете большей мощи и гибкости в отображении текста на экране. Типичную escape-последовательность образуют два символа: обратный слеш и еще какой-либо символ. Все это пока лишь таинственные слова, но достаточно увидеть своими глазами несколько escape-последовательностей, и вы сразу поймете, как же просто ими пользоваться.

Знакомство с программой «Воображаемые благодарности»

Некоторые программы, известив пользователя о том, что игра окончена, выводят «благодарности» — список всех лиц, принимавших участие в работе над проектом. «Воображаемые благодарности» — это программа, в которой с помощью escape-последовательностей можно получить некоторые эффекты, недостижимые иным образом. Результат запуска программы показан на рис. 2.3.

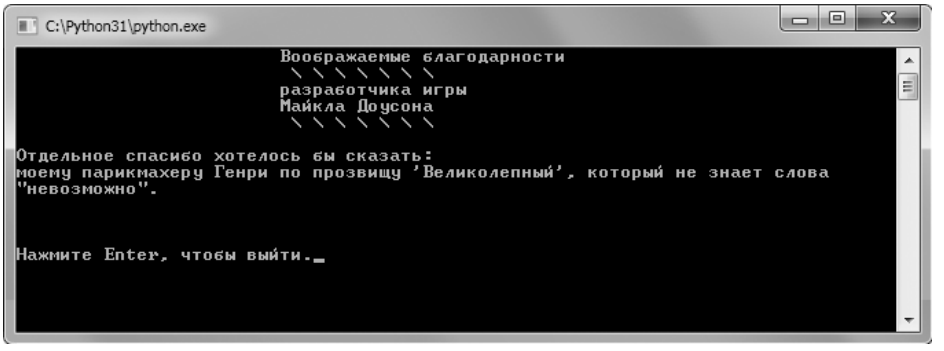

Рис. 2.3. Не надо оваций…

На первый взгляд код этой программы довольно загадочен, но вы его скоро поймете. Код можно найти на сайте-помощнике (courseptr.com/downloads) в папке Chapter 2; файл называется `fancy_credits.py`.

```
# Воображаемые благодарности
# Образец применения escape-последовательностей
print("\t\t\tВоображаемые благодарности")
print("\t\t\t \\ \\ \\ \\ \\ \\ \\")
print("\t\t\tразработчика игры")
print("\t\t\tМайкла Доусона")
print("\t\t\t \\ \\ \\ \\ \\ \\ \\")
print("\nОтдельное спасибо хотелось бы сказать:")
print("моему парикмахеру Генри по прозвищу \'Великолепный\', который не знает слова \"невозможно\".")
# Звучит системный динамик
print("\a")
input("\n\nНажмите Enter, чтобы выйти.")
```

Вставка табуляционного отступа

Иногда надо «отодвинуть» текст от левой кромки окна, по которой он обычно выравнивается. В текстовом редакторе можно воспользоваться для этого клавишей Tab, а в данных строкового типа применяется соответствующая escape-последовательность `\t`. Именно таково значение символов `\t` в строке кода:

```
print("\t\t\tВоображаемые благодарности")
```

Escape-последовательность табуляционного отступа встречается здесь три раза подряд, так что, выводя текст на экран, программа сначала делает три отступа, а затем печатает слова `Воображаемые благодарности`. Поэтому в консольном окне заголовок `Воображаемые благодарности` находится приблизительно в центре.

Табуляция удобна не только для создания отступов, как в этой программе, но и для оформления текста в виде нескольких столбцов.

Вывод обратного слеша

Если вы достаточно предусмотрительны, то, возможно, уже задумались о том, как же напечатать обратный слеш, если интерпретатор всегда распознает его как начальный символ escape-последовательности. Ответ прост: надо использовать два обратных слеша подряд. Каждая из следующих строк кода выводит на экран три табуляционных отступа и вслед за тем семь обратных слешей (столько же, сколько пар `\\`), разделенных пробелами:

```
print("\t\t\t \\ \\ \\ \\ \\ \\ \\")
print("\t\t\t \\ \\ \\ \\ \\ \\ \\")
```

Вставка пустой строки

Едва ли не самая полезная из escape-последовательностей отвечает за переход на новую строку и выглядит как `\n`. Эта последовательность позволяет вставлять

в текст, где необходимо, символ пустой строки. Символ `\n` в начале строкового литерала может, например, выразительно отделить текст от напечатанного на экране выше. Этот эффект и достигнут в выражении:

```
print("\nОтдельное спасибо хотелось бы сказать:")
```

Компьютер видит последовательность `\n`, печатает пустую строку и лишь потом выводит текст `Отдельное спасибо хотелось бы сказать:`.

Вставка кавычек

Оказывается, вставить в строковый литерал цитату (даже с такими же ограничителями, как ограничители самой строки) очень легко. За вывод буквального апострофа отвечает escape-последовательность `\'`, а двойные кавычки представляются в виде `\"`. Эти команды компьютер не может перепутать с концом строки. Вот почему не вызывает ошибки следующая строка кода, в которой использованы кавычки обоих типов:

```
print("моему парикмахеру Генри по прозвищу \'Великолепный\', который не знает слова \"невозможно\".")
```

В качестве ограничителей строки здесь выступают двойные кавычки. Рассмотрим строку по частям, чтобы стало яснее.

1. `\'Великолепный\'` отображается на экране как `'Великолепный'`.
2. Каждая из двух последовательностей `\'` отображается как апостроф.
3. `\"невозможно\"` на экране принимает вид `"невозможно"`.
4. Обе последовательности `\"` печатаются как двойные кавычки.

Звук системного динамика

При запуске этой программы вы без труда заметите ее отличительную черту: она звучит! Команда `print("\a")` заставляет звучать системный динамик вашего компьютера. Это непосредственно делает escape-последовательность `\a` — «символ системного динамика». Каждый раз при применении к нему функции `print()` динамик звучит. Можно «напечатать» только эту последовательность, как я и сделал, а можно поместить ее внутрь строкового литерала. Ничто не мешает также инициировать звук системного динамика несколько раз — «напечатать» несколько символов `\a`.

Некоторые escape-последовательности работают предусмотренным образом лишь тогда, когда программа запускается из операционной системы, а не в IDLE. Таков, в частности, символ системного динамика. Пусть, например, я написал программу из одного выражения `print("\a")`. Если запустить ее в IDLE, на экране появится квадратик, а компьютер сохранит молчание. Но если запустить ту же самую программу из Windows, то есть дважды щелкнуть на значке файла, то системный динамик прозвучит, как и должен.

Итак, в действии escape-последовательности кажутся не так уж и плохи: они обеспечивают большой прирост функциональности. В табл. 2.1 приведены некоторые полезные escape-последовательности.

Таблица 2.1. Escape-последовательности

Последовательность	Описание
\\	Обратный слеш. Выводит один знак обратного слеша
\'	Апостроф, или одиночная кавычка. Выводит один апостроф
\"	Двойные кавычки. Выводит одну такую кавычку
\a	Звук системного динамика. Заставляет звучать динамик компьютера
\n	Пустая строка. Перемещает курсор в начало следующей строки
\t	Горизонтальный отступ — символ табуляции. Перемещает курсор вправо на один табуляционный отступ

Сцепление и повторение строк

Итак, вы увидели, как можно вставлять в строки специальные символы. Но есть возможность манипулировать и целыми строками, например соединять две в одну или повторять какую-либо строку произвольное количество раз.

Знакомство с программой «Забавные строки»

Программа «Забавные строки» выводит на экран несколько строк. Результаты ее работы отражены на рис. 2.4. Хотя с печатью строк на экране вы уже знакомы, предлагаемый здесь способ вывода текста на экран будет для вас в новинку. Код этой программы вы можете найти на сайте-помощнике (www.courseptr.com/downloads) в папке Chapter 2. Файл называется `silly_strings.py`.

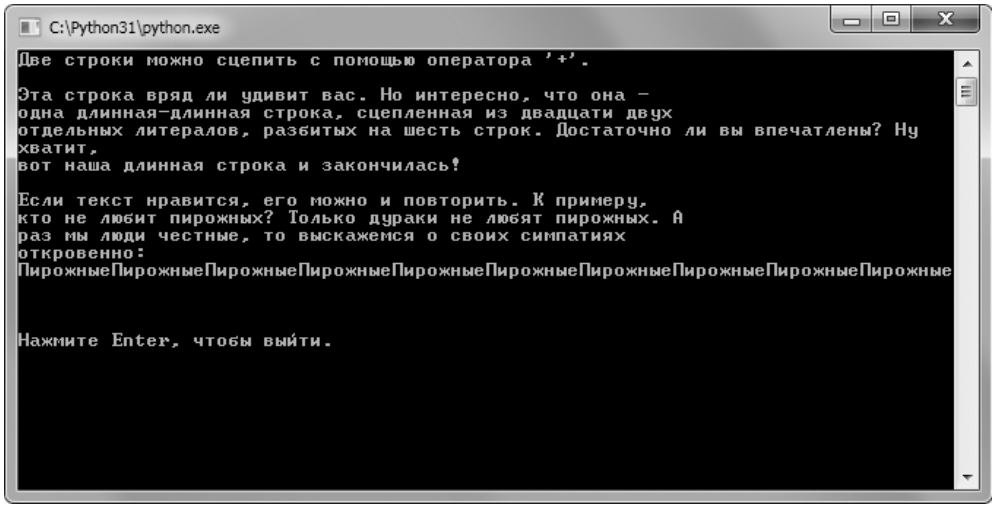

Рис. 2.4. Строки на экране выглядят иначе, чем в исходном коде программы