小小图灵社“爱上编程的孩子”公益活动

Python 零基础入门讲义

编撰：詹有丘

小小图灵社 Little Turings

1. 计算机基础

1. 文件管理系统
    1. 目录

        当前目录是运行命令的环境。
    2. 文件

        文件的内容是一串字符。

        文件名中最后一个“.”后的内容是它的后缀名。

2. 浏览器
    1. 使用浏览器访问网页
    2. 使用浏览器下载文件

3. 命令行的使用（Windows）
    1. 命令的运行

        Windows 上运行命令行的终端程序为 `cmd`。

        快捷地运行命令：开始-运行。快捷键：Win+R。

        运行命令 `help` 可以看到 Windows 上的常用命令。
    2. 用命令行启动应用程序

        运行命令时，Windows 将从当前目录和 PATH 环境变量中搜索与所要运行的命令的名称相符的可执行文件，然后执行命令。
    3. 控制命令的运行

        按 Ctrl+C 结束命令。

        按 Ctrl+D 结束输入。

小图灵社 Little Turings

2. 环境搭建

1. Windows 上安装 Python
    1. 下载 Python

       访问 https://www.python.org/downloads/windows/，下载 executable installer。

       根据系统是 32 位系统还是 64 位系统下载 x86 或 x86-64 的 installer。
    2. 安装 Python

       运行 installer 开始安装。建议勾选 Add Python to PATH。

2. 测试 Python

   运行 `python -V` 后，计算机上所安装的 Python 的版本号将被输出。确保此命令能正常运行。

3. 运行 Python
    1. 交互式解释器

       使用命令 `python` 来运行交互式解释器。
    2. 运行文件中的 Python 脚本

       将文件名作为命令 `python` 的参数，可以运行文件中的 Python 脚本。
    3. 使用 IDLE

       IDLE 是安装 Python 时附带的软件，集成了上述功能和编辑器的功能。
    4. Hello, World!

       写一个文件，内容是

       ```
       print('Hello, world!')
       ```

       然后用 `python` 命令运行它。

4. 编辑器
    1. 纯文本编辑器

       纯文本编辑器用于以纯文本的方式编辑文件。

小图灵社 Little Turings

常用的纯文本编辑器有 Windows 自带的记事本（`notepad` 命令），以及 Sublime Text、Atom、Notepad++、Edit Plus、Ultra Edit……

2. 集成开发环境（IDE）

   IDE 是辅助程序开发人员开发软件的应用软件。

   常用的支持 Python 的 IDE 有 Python 自带的 IDLE、Thonny，以及 Visual Studio、PyCharm、Spyder……

小小图灵社 Little Turings

3. 基础语法

1. 输出

   使用 `print` 函数来输出并换行。

2. 赋值语句

   1. 变量
   2. 标识符

      标识符由字母、数字、下划线组成。第一个字符不能是数字。

      标识符不能是保留字。运行下面的脚本查看 Python 中的保留字：

      ```
      import keyword
      print(keyword.kwlist)
      ```

   3. 赋值符号

      用“=”进行赋值操作。右值送左值。

   4. `del` 语句

      用于删除变量。

3. 注释

   Python 解释器会忽略注释。

   单行注释以“#”开头。

   多行注释用“'''”或“"""”括起来。

4. 一行语句分成多行

   如果语句很长，可以用“\”分成多行。

5. 多行语句并成一行

   用“;”把多个语句连接起来。

小小图灵社 Little Turings

4. 基本数据类型

1. 七个标准数据类型:

   Number（数），String（字符串），List（列表），Tuple（元组），Set（集合），Dictionary（字典），Range（范围）。

2. 获取和判断数据类型

   用 `type` 函数来获取数据类型。

   用 `isinstance` 函数来判断是否属于某一数据类型。

3. Number

   1. 分类

      Number 分为 `int`，`float`，`bool`，`complex`。

   2. 运算

      加、减、乘、除、整除、取余、乘方。

4. 数据类型的转换

   1. `int` 函数和 `float` 函数

      转换成 `int` 型数据或 `float` 型数据。

   2. `str` 函数和 `repr` 函数

      转换成 `String`。

   3. `eval` 函数

      运行字符串中的脚本。

小小图灵社 Little Turings

5. 运算符

1. 算术运算符

   数学运算符：加+，减-，乘*，除以/，商数//（对商向下取整），余数%，幂**。

   位运算符：按位与&，按位或|，按位非~，按位异或^，左移<<，右移>>。

2. 比较运算符

   等于==，不等于!=，大于>，小于<，大于等于>=，小于等于<=。

   身份运算符：`is`，`is not`。

3. 赋值运算符

   自运算符：`+=`，`-=`，`*=`……

   海象运算符（Walrus operator）“`:=`”用于在表达式内部为变量赋值。

4. 逻辑运算符

   与`and`，或`or`，非`not`。

5. 成员运算符

   `in`，`not in`用于判断是否是序列的成员。序列可以是String，List，Tuple，Range等。

小小图灵社 Little Turings

6. 数学

1. 实数

   可以认为 Python 中的 `float` 就是实数。

2. 实数的大小比较

   Python 中用比较运算符进行比较。

   在 Python 2 中还可以用 `cmp` 函数。

   Python 中用 `min` 函数和 `max` 函数分别求序列中的最小值和最大值。

3. 绝对值

   绝对值指的是数在数轴上的对应点到原点的距离的长度。

   x的绝对值记作$|x|$，定义为$|x| = \begin{cases} x, & x > 0 \\ 0, & x = 0 \\ -x, & x < 0 \end{cases}$。

   Python 中可以用 `abs` 函数或 `math` 库中的 `fabs` 函数求绝对值。

4. 四则运算

   Python 中用算术运算符进行四则运算。

5. 乘方

   乘方定义为$a^n = \underbrace{a \times a \times a \times \cdots \times a}_{n}$，其中$a$被称为底数，$n$被称为指数，运算的结果称为幂。读作$a$的$n$次方或$a$的$n$次幂。

   Python 中用`**`运算符或 `math` 库中的 `pow` 函数求幂。

6. 开方

   若$a^n = b$，则a是b的n次方根。

   根号：对于$a > 0$，将a的正的平方根记作$\sqrt{a}$，读作根号a或a的算术平方根。

小小图灵社区 Little Turings

Python 中用 `math` 库中的 `sqrt` 函数求算术平方根。

7. 取整

   向下取整：$\lfloor x \rfloor$表示小于等于x的最大整数。Python 中的 `floor` 函数。$\lfloor x \rfloor$又叫做x的整数部分。

   Python 中 `modf` 函数可以求整数部分和小数部分，注意它与数学上的整数部分和小数部分的定义不符。

   向上取整：$\lceil x \rceil$表示大于等于x的最小整数。Python 中的 `ceil` 函数。

   将x四舍五入到个位的方法是求$\lfloor x + 0.5 \rfloor$。Python 中的 `round` 函数与四舍五入的算法类似，但不完全一样。

   上面提到的函数中除了 `round` 以外都在 `math` 库中。

8. 随机数

   `choice` 函数从序列中随机挑选一个元素。

   `randrange` 函数从指定范围内随机获取一个整数。

   `random` 函数随机挑选一个$[0, 1)$内的实数。

   `shuffle` 函数打乱序列中的元素。

   `uniform` 函数随机挑选一个闭区间内的实数。

   以上函数都在 Python 的 `random` 库中。

小小图灵 Little Turings

7.列表

1. 访问列表中的值

```
list1 = [11, 12, 31, 41]
print(f"{list1[2]=}") # => list1[2]=31
print(f"{list1[0]=}") # => list1[0]=11
print(f"{list1[4]=}") # => IndexError
print(f"{list1[-2]=}") # => list1[-2]=31
print(f"{list1[1:3]=}") # => list1[1:3]=[12, 31]
print(f"{list1[:3]=}") # => list1[:3]=[11, 12, 31]
print(f"{list1[1:]=}") # => list1[1:]=[12, 31, 41]
print(f"{list1[:]=}") # => list1[:]=[11, 12, 31, 41]
```

2. 写入列表

```
list2 = [11, 12, 31, 41]
print(f"{list2[2]=}") # => list2[2]=31
list2[2] = 23
print(f"{list2[2]=}") # => list2[2]=23
list2.append(3)
print(f"{list2=}") # => list2=[11, 12, 31, 41, 3]
del list2[1]
print(f"{list2=}") # => list2=[11, 31, 41, 3]
```

3. 涉及列表的运算符
    1. 拼接：+

    ```
    print([1, 2, 3] + [4, 5, 6]) # => [1, 2, 3, 4, 5, 6]
    ```

    2. 重复：*

    ```
    print([1, 2] * 3) # => [1, 2, 1, 2, 1, 2]
    ```

    3. 判断成员：in

    ```
    list3 = [520, 1314]
    print(520 in list3) # => True
    ```

小小图灵 Little Turings

```
print(1108 in list3) # => False
```

4. 涉及列表的函数
    1. `len(list)`返回 `list` 的元素个数
    2. `max(list)`和 `min(list)`分别返回 `list` 中的最大和最小元素
    3. `list(seq)`返回从 `seq` 转换成的列表

小小图灵社 Little Turings

8. 字符串

1. 创建字符串

   使用单引号“'”或双引号“"”来创建字符串。

   使用三个连续引号使字符串能跨越多行。

2. 读写字符串

   字符串中的字符可以像列表中的元素一样读写。

3. 转义字符

   反斜杠\\，单引号\'，双引号\"，空\000，换行符\n，制表符\t。

4. 涉及字符串的运算符

   连接+，重复*，判断成员 in，格式化%。

5. 原始字符串

   在字符串被创建时在前面加上“r”标记，表示不处理转义字符。

6. 格式化字符串

   部分格式化符号：字符串%s，整数%d，浮点数%f，百分号%%。

   ```
   a = 'Alice'
   print('Hello, %s!' % a) # => Hello, Alice!
   print('%s has %d pens' % (a, 3)) # => Alice has 3 pens
   ```

7. f-string

   ```
   x = 1
   print(f'{x}') # => 1
   print(f'{x=}') # => x=1
   ```

小图灵社 Little Turings

9. 控制结构

1. 流程图

2. 分支

   if 语句的格式:

   ```
   if condition1:
      statement1
   elif condition2:
      statement2
   else:
      statement3
   ```

   其中 elif 部分和 else 部分不是必须的。

3. 循环

   1. while 语句

      while 语句的格式:

      ```
      while condition:
         statement1
      else:
         statement2
      ```

      else 部分不是必须的。

   2. for 语句

      for 语句的格式:

      ```
      for variable in sequence:
         statement1
      else:
         statement2
      ```

      else 部分不是必须的。

小小图灵社 Little Turings

3. break 语句

   break 语句用于跳出循环。

   使用 break 语句跳出循环后 else 中的内容

4. continue 语句

   continue 语句用于进入下一次循环。

4. pass 语句

   pass 语句什么也不干。

小小图灵社 Little Turings

10. 函数

1. def 语句

   def 语句的格式：

   ```
   def function(parameters):
      statement
   ```

2. return 语句和函数的返回值

   return 语句可以在函数内部直接退出函数。

   若给予 return 语句一个表达式，则这个函数被调用时也是一个表达式，其返回值就是 return 的表达式。

3. 库（模块）

   使用 import 语句导入库，这将使库中的函数能够被调用。

   可以自己编写库让别人来使用。

小小图灵社 Little Turings

11. 我还能学什么?

如果你觉得以上内容都很简单，恭喜你，你可能有很高的编程天赋!

但这些内容实际上并没有让你触及到真正的编程。就 Python 这门语言本身，你还有很多东西可以学:

- 其他的运算符
- 其他的数据类型及其用法
- 嵌套的控制结构
- 迭代器与生成器
- 函数的参数传递
- 包和库的管理
- 输入、输出
- 异常处理
- 面向对象
- 命名空间和作用域
- 常用的标准库
- 正则表达式
- 多线程

如果你能熟练掌握以上内容并且用这些知识写出了规模较大的程序，那你或许已经精通 Python 语言了！这通常需要一个新手数月甚至数年的学习。

小小图灵社 Little Turings