关于 `.gitignore`

```
**/*.zip
**/__pycache__
```

`git` 不需要维护 `压缩包` 和 `缓存文件`

Lab 0: Getting Started

实验链接：Lab 0: Getting Started
关于实验测评：Using Ok

实验环境配置

`CS 61A` 的实验需要：

- `Terminal（终端）`
- `Python3`
- `Text Editor（代码编辑器）`

命令行工具的使用

实验会可能用到的命令：

- `ls`
- `cd`
- `mkdir`
- `unzip`
- `mv`

Python Basic

Primitive Expressions（原始表达式）

`原始表达式` 只进行一步计算：

```
>>> 3
3
>>> 12.5
12.5
>>> True
True
```

Arithmetic Expressions（算术表达式）

算术表达式由以下运算符组成：

- 加法：`+`
- 减法：`-`
- 乘法：`*`

- 乘方：`**`
- 浮点数除法：`/`
- 下取整除法：`//`
- 取余：`%`

```
>>> 7 / 4
1.75
>>> (2 + 6) / 4   # Floating point division
2.0
>>> 7 // 4        # Floor division (rounding down)
1
>>> 7 % 4         # Modulus (remainder of 7 // 4)
3
```

Assignment statements（赋值语句）

```
>>> a = (100 + 50) // 2
```

```
>>> a
75
```

Doing the assignment（实验任务）

Unlocking tests（解锁测试）

完成正式实验前要进行一次测试，检测学生对 `python` 的基础知识的掌握程度，使用如下命令（`--local` 参数使得测评在本地运行）：

```
python3 ok -q python-basics -u --local
```

测试过程如下：

```
===================================================================
Assignment: Lab 0
OK, version v1.18.1
===================================================================

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Unlocking tests

At each "? ", type what you would expect the output to be.
Type exit() to quit

---------------------------------------------------------------------
Python Basics > Suite 1 > Case 1
(cases remaining: 2)

What would Python display? If you get stuck, try it out in the Python
interpreter!

>>> 10 + 2
? 12
-- OK! --

>>> 7 / 2
? 3.5
```

```
-- OK! --

>>> 7 // 2
? 3
-- OK! --

>>> 7 % 2                       # 7 modulo 2, the remainder when dividing 7 by
2.
? 1
-- OK! --

---------------------------------------------------------------------
Python Basics > Suite 2 > Case 1
(cases remaining: 1)

What would Python display? If you get stuck, try it out in the Python
interpreter!

>>> x = 20
>>> x + 2
? 22
-- OK! --

>>> x
? 20
-- OK! --

>>> y = 5
>>> y = y + 3
>>> y * 2
? 16
-- OK! --

>>> y = y // 4
>>> y + x
? 22
-- OK! --

---------------------------------------------------------------------
OK! All cases for Python Basics unlocked.

Cannot backup when running ok with --local.
```

Understanding problems（了解问题）

> In `twenty_twenty_one`,
>
> - The docstring tells you to "come up with the most creative expression that evaluates to 2021," but that you can only use numbers and arithmetic operators `+` (add), `*` (multiply), and `-` (subtract).
> - The `doctest` checks that the function call `twenty_twenty_one()` should return the number 2021.

代码如下：

```
def twenty_twenty_one():
    """Come up with the most creative expression that evaluates to 2021,
    using only numbers and the +, *, and - operators.

    >>> twenty_twenty_one()
    2021
    """
    return 20 * 100 + 21
```

Running tests（运行测试）

使用如下命令测评代码：

```
python3 ok --local
```

测评结果如下：

```
=====================================================================
Assignment: Lab 0
OK, version v1.18.1
=====================================================================

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Running tests

---------------------------------------------------------------------
Test summary
    3 test cases passed! No cases failed.

Cannot backup when running ok with --local.
```

Appendix: Useful Python command line options（有用的Python命令行选项）

- `-i` : The `-i` option runs your Python script, then opens an interactive session. In an interactive session, you run Python code line by line and get immediate feedback instead of running an entire file all at once. To exit, type `exit()` into the interpreter prompt. You can also use the keyboard shortcut `Ctrl-D` on Linux/Mac machines or `Ctrl-Z Enter` on Windows.

  If you edit the Python file while running it interactively, you will need to exit and restart the interpreter in order for those changes to take effect.

  ```
  python3 -i
  ```

- `-m doctest` : Runs doctests in a particular file. Doctests are surrounded by triple quotes (`"""`) within functions.

  Each test in the file consists of `>>>` followed by some Python code and the expected output (though the `>>>` are not seen in the output of the doctest command).

  ```
  python3 -m doctest
  ```