Function

Python 基础入门

回顾

函数的 声明

函数的 调用

形参 和 实参

返回值

作用域

质数 的判断

递归

函数的 递归

- ✓ 在函数中 调用自身，称作 递归
- ✓ 大而化小，分而治之
- ✓ 递归函数必须有 结束条件，不能 无限递归

```
def my_print():
    print('Hello My Print')
    my_print()
```

```
def my_print(N):
    print('Hello My Print')
    if N > 0:
        my_print(N - 1)
```

累加和

一个 简单 的例子

- ✓ 计算从 1 到 N 的 累加和，1 + 2 + … + N
- ✓ 先写个函数，告诉自己它可以 做什么
- ✓ 在必要的时候，进行 递归
  - ✓ N 等于 1
  - ✓ N 大于 1

```
def cum_sum(N):
    if N == 1:
        return 1
    else:
        return ?
```

```
def cum_sum(N):
    if N == 1:
        return 1
    else:
        return N + cum_sum(N - 1)
```

阶乘

又一个 简单 的例子

- ✓ 计算 N 的 阶乘，1 * 2 * … * N
- ✓ 先写个函数，告诉自己它可以 做什么
- ✓ 在必要的时候，进行 递归
  - ✓ N 等于 1
  - ✓ N 大于 1

```
def factorial(N):
    if N == 1:
        return 1
    else:
        return N * factorial(N - 1)
```

练习

1 斐波那契

- ✓ 编写一个函数
- ✓ 返回 斐波那契 数列中的第 N 个数
- ✓ 1, 1, 2, 3, 5, 8, 13, 21…

练习

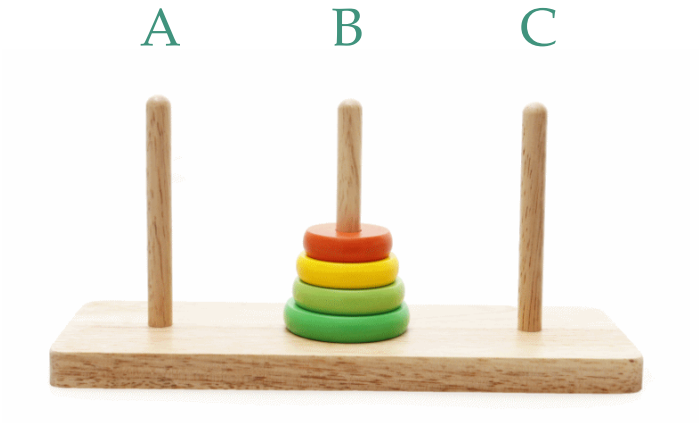
2 走路

- ✓ 从家里到学校共 20 米
- ✓ 每步可以走 1 米或 2 米
- ✓ 一共有多少种 不同的 走路方案
- ✓ 例如，走 20 个 1 米，走 10 个 2 米

练习

3 汉诺塔

- ✓ 三根 柱子，N 个从小到大的 圆盘
- ✓ 移动圆盘，大的只能在小的 下面
- ✓ 通过移动，将圆盘全部移到 另一根 柱子上



练习

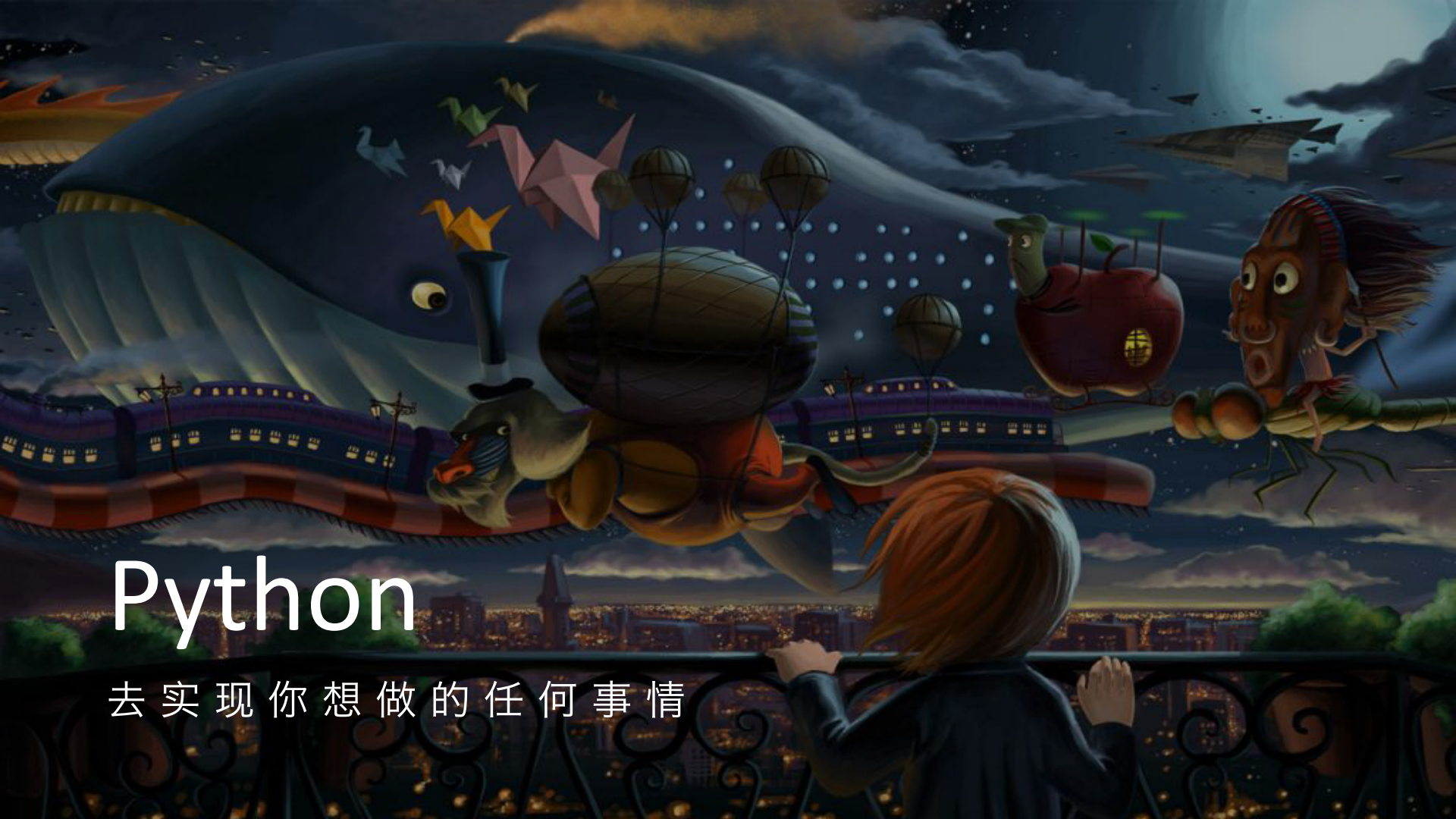
3 汉诺塔

- ✓ 三根 柱子，N 个从小到大的 圆盘
- ✓ 移动圆盘，大的只能在小的 下面
- ✓ 通过移动，将圆盘全部移到 另一根 柱子上

```
def hano(N, source, target):
    if N == 1:
        print(1, source, '=>', target)
    else:
        pillars = ['A', 'B', 'C']
        pillars.remove(source)
        pillars.remove(target)
        media = pillars[0]

        hano(N - 1, source, media)
        print(N, source, '=>', target)
        hano(N - 1, media, target)

hano(4, 'B', 'C')
```


Python
去 实 现 你 想 做 的 任 何 事 情