Python语言基础与应用

Python程序设计风格

陈斌 北京大学 gischen@pku.edu.cn

Python程序设计风格

- Python程序设计
- 程序是写给人读的
- 程序员的精彩人生
- Python 哲学

Python 程序的风格：优雅、明确、简单

归并排序的不同版本

Python版：

```python
def merge_sort(lst):
    if len(lst) <= 1:
        return lst
    middle = int(len(lst) / 2)
    left = merge_sort(lst[:middle])
    right = merge_sort(lst[middle:])
    merged = []
    while left and right:
        merged.append(left.pop(0) if left[0]
    merged.extend(right if right else left)
    return merged
```

C语言版：

```c
void merge_sort_recursive(int arr[], int reg[], int start, int end) {
    if (start >= end)
        return;
    int len = end - start, mid = (len >> 1) + start;
    int start1 = start, end1 = mid;
    int start2 = mid + 1, end2 = end;
    merge_sort_recursive(arr, reg, start1, end1);
    merge_sort_recursive(arr, reg, start2, end2);
    int k = start;
    while (start1 <= end1 && start2 <= end2)
        reg[k++] = arr[start1] < arr[start2] ? arr[start1++] : arr[sta
    while (start1 <= end1)
        reg[k++] = arr[start1++];
    while (start2 <= end2)
        reg[k++] = arr[start2++];
    for (k = start; k <= end; k++)
        arr[k] = reg[k];
}
void merge_sort(int arr[], const int len) {
    int reg[len];
    merge_sort_recursive(arr, reg, 0, len - 1);
}
```

Python 程序设计

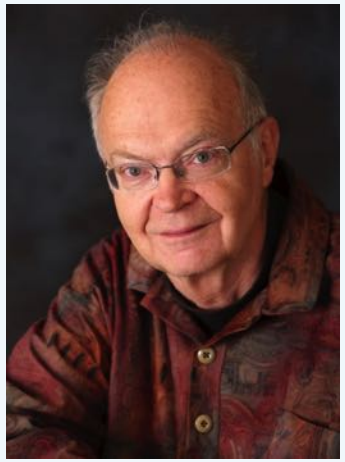
代码强制缩进

```
1  static OSStatus
2  SSLVerifySignedServerKeyExchange(SSLContext *ctx, bool isRsa, SSLBuffer signedPa
3                                   uint8_t *signature, UInt16 signatureLen)
4  {
5      OSStatus        err;
6      ...
7
8      if ((err = SSLHashSHA1.update(&hashCtx, &serverRandom)) != 0)
9          goto fail;
10     if ((err = SSLHashSHA1.update(&hashCtx, &signedParams)) != 0)
11         goto fail;
12         goto fail;
13     if ((err = SSLHashSHA1.final(&hashCtx, &hashOut)) != 0)
14         goto fail;
15     err = sslRawVerify(ctx,
16                        ctx->peerPubKey,
17                        dataToSign,            /* plaintext */
18                        dataToSignLen,         /* plaintext length */
19                        signature,
20                        signatureLen
21     if(err) {
22         sslErrorLog("SSLDecodeSigne
23                     "returned %d\n"
24         goto fail;
25     }
26
27 fail:
28     SSLFreeBuffer(&signedHashes);
29     SSLFreeBuffer(&hashCtx);
30     return err;
31 }
```

```
7
8       if ((err = SSLHashSHA1
9           goto fail;
10      if ((err = SSLHashSHA1
11          goto fail;
12          goto fail;
13      if ((err = SSLHashSHA1
14          goto fail;
15      err = sslRawVerify(ctx
```

程序是写给人读的

- **Python的强制缩进规范完成了关键部分**
- **我们还需要良好的编程规范**
  - 变量、函数、类命名
  - 注释和文档
  - 一些编程设计上的良好风格

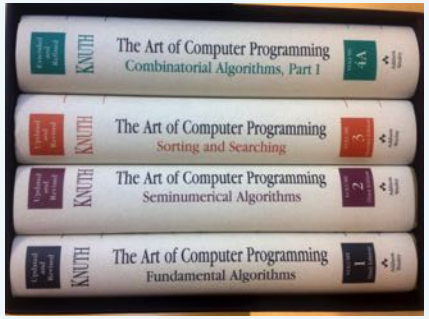


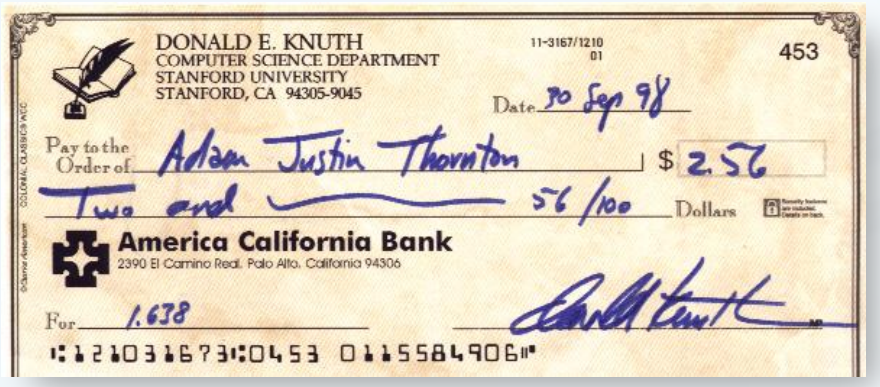
Programs are meant to be read by humans and only incidentally for computers to execute.

—Donald Ervin Knuth

程序员的精彩人生

- 鸿篇巨著《计算机程序设计艺术》
- 为了巨著印刷漂亮，开发了伟大的排版软件T_EX
- 字符串快速匹配KMP算法
- 1974年获得图灵奖





Python哲学

```
>>> import this
The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
```