Tutorial 2: Code Style and Code Quality

Kyran Fang

SIST

2022

Outlines

- Bonus
- Coding style: Why important?
- Core Ideas
- Naming
- Whitespaces and Indentation
- Comments

Bonus

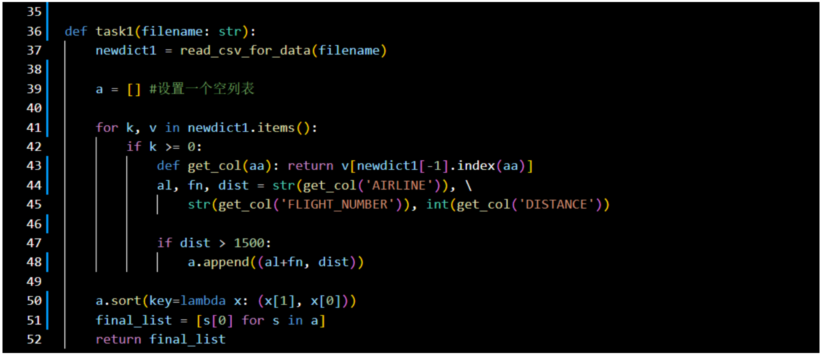
- Deep-learning-based MRI Image Denoising
- Encapsulated matrix calculator
- Aircraft Games
- Code Converter

Coding style: Why important?

- Makes code easier to read for others.
- Easier to read for yourself.
- Also makes code aesthetically pleasing.

Coding style: Why important? Cont'd

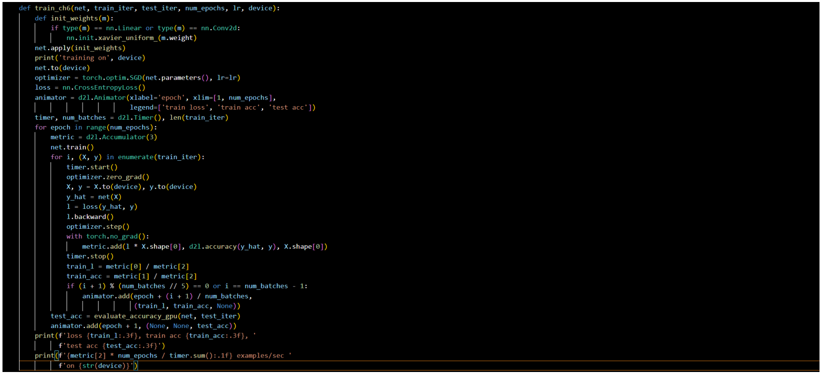
My programming assignments1 code of SI100B:



```
def task1(filename: str):
    newdict1 = read_csv_for_data(filename)

    a = [] #设置一个空列表

    for k, v in newdict1.items():
        if k >= 0:
            def get_col(aa): return v[newdict1[-1].index(aa)]
            al, fn, dist = str(get_col('AIRLINE')), \
                str(get_col('FLIGHT_NUMBER')), int(get_col('DISTANCE'))

            if dist > 1500:
                a.append((al+fn, dist))

    a.sort(key=lambda x: (x[1], x[0]))
    final_list = [s[0] for s in a]
    return final_list
```

Coding style: Why important? Cont'd

My recent code:



```python
def train_ch6(net, train_iter, test_iter, num_epochs, lr, device):
    def init_weights(m):
        if type(m) == nn.Linear or type(m) == nn.Conv2d:
            nn.init.xavier_uniform_(m.weight)
    net.apply(init_weights)
    print('training on', device)
    net.to(device)
    optimizer = torch.optim.SGD(net.parameters(), lr=lr)
    loss = nn.CrossEntropyLoss()
    animator = d2l.Animator(xlabel='epoch', xlim=[1, num_epochs],
                            legend=['train loss', 'train acc', 'test acc'])
    timer, num_batches = d2l.Timer(), len(train_iter)
    for epoch in range(num_epochs):
        metric = d2l.Accumulator(3)
        net.train()
        for i, (X, y) in enumerate(train_iter):
            timer.start()
            optimizer.zero_grad()
            X, y = X.to(device), y.to(device)
            y_hat = net(X)
            l = loss(y_hat, y)
            l.backward()
            optimizer.step()
            with torch.no_grad():
                metric.add(l * X.shape[0], d2l.accuracy(y_hat, y), X.shape[0])
            timer.stop()
            train_l = metric[0] / metric[2]
            train_acc = metric[1] / metric[2]
            if (i + 1) % (num_batches // 5) == 0 or i == num_batches - 1:
                animator.add(epoch + (i + 1) / num_batches,
                             (train_l, train_acc, None))
        test_acc = evaluate_accuracy_gpu(net, test_iter)
        animator.add(epoch + 1, (None, None, test_acc))
    print(f'loss {train_l:.3f}, train acc {train_acc:.3f}, '
          f'test acc {test_acc:.3f}')
    print(f'{metric[2] * num_epochs / timer.sum():.1f} examples/sec '
          f'on {str(device)}')
```

Core Ideas

♠ **READABLE**

Not just for you but also for other guys! (and that guy may be yourself three months later)

Core Ideas Cont'd

♠ **REASONABLE**

There is no rule for code style, but you should always code with your logic.

Naming

Naming scheme

1. ♠ Snake Case
   - ♠ find_location
   - ♠ train_acc
2. ♠ Little Camel Case
   - ♠ evaluateAccuracyGpu
   - ♠ numEpochs
3. ♠ Big Camel Case
   - ♠ InitWeight
   - ♠ CrossRntropyLoss
     - ∗ Do not use this for variables.

Naming Cont'd

Naming variables

- Good variables names:
  - Reflect its value or function.
  - Eliminate ambiguity.
  - Fit the enviroment and its function.
- Examples:
  - Good names:
    read_from_csv, FibSeries, countAllMoves ...
  - AWFUL names:
    o0OO0o,I1L111Il, I_HATE_TA, xx, *a,x,r,i,k,j* ...

Naming Cont'd

♠ Use as less magic number as you can!

♠ If you have to, remember to write a comment about the magic number you use.

Whitespaces

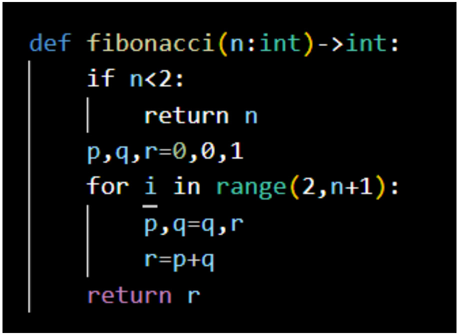
♠It is impropiate to use whitespaces after a punctuation mark as you are writing a English article, like:

"What+ does? spring== look, like. on@ Jupiter"

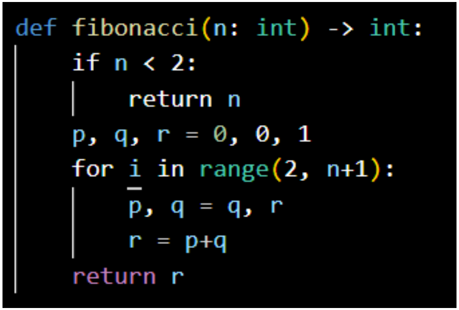
♠You SHOULD use whitespaces before & after some operators like $+, -, ==, >,$ and $=$.

For example:
1 + 1, ans += 1

Whitespaces cont'd



```python
def fibonacci(n:int)->int:
    if n<2:
        return n
    p,q,r=0,0,1
    for i in range(2,n+1):
        p,q=q,r
        r=p+q
    return r
```



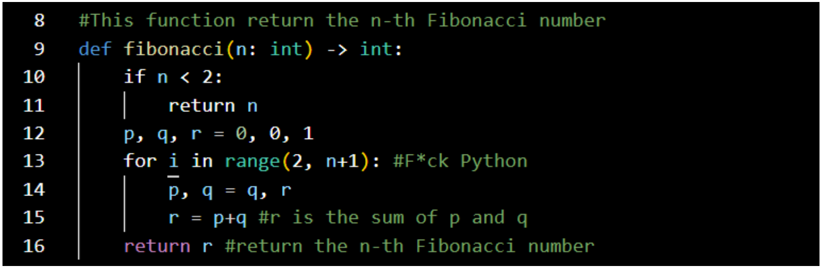
```python
def fibonacci(n: int) -> int:
    if n < 2:
        return n
    p, q, r = 0, 0, 1
    for i in range(2, n+1):
        p, q = q, r
        r = p+q
    return r
```

Comments

- Meaningful comments are
  - Complicate calculus/control flow/binary magic/magic number
  - Regular expressions
  - No nonsense
  - Better in English
- Awful comments are
  - Transliteral of your code
  - Hard to read
  - Unrelated to the content

Comments cont'd

Which one is meaningful comment?



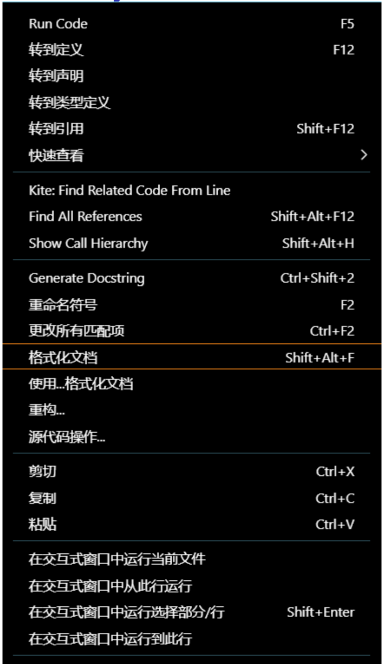
```
#This function return the n-th Fibonacci number
def fibonacci(n: int) -> int:
    if n < 2:
        return n
    p, q, r = 0, 0, 1
    for i in range(2, n+1): #F*ck Python
        p, q = q, r
        r = p+q #r is the sum of p and q
    return r #return the n-th Fibonacci number
```

Comments cont'd

♠ Good naming reduce the need of commenting!

♠ Most inexperienced developers do not know how to do commenting.

Master Key


Run Code F5
转到定义 F12
转到声明
转到类型定义
转到引用 Shift+F12
快速查看 >
Kite: Find Related Code From Line
Find All References Shift+Alt+F12
Show Call Hierarchy Shift+Alt+H
Generate Docstring Ctrl+Shift+2
重命名符号 F2
更改所有匹配项 Ctrl+F2
格式化文档 Shift+Alt+F
使用...格式化文档
重构...
源代码操作...
剪切 Ctrl+X
复制 Ctrl+C
粘贴 Ctrl+V
在交互式窗口中运行当前文件
在交互式窗口中从此行运行
在交互式窗口中运行选择部分/行 Shift+Enter
在交互式窗口中运行到此行