74+戴一帆+211205102388

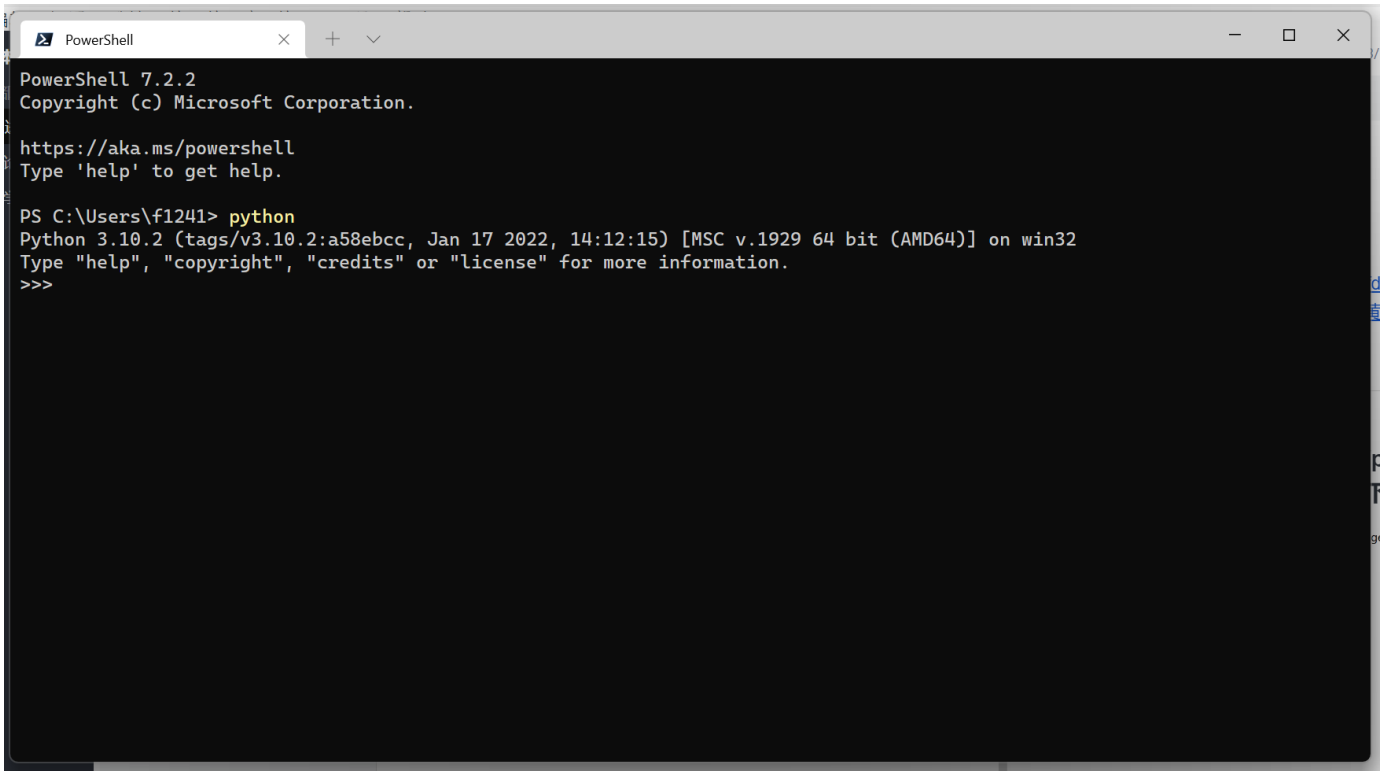
正则表达式4

戴一帆

2022年3月27号

1、Python模块本地安装

1、根据 python 版本，通过https://www.lfd.uci.edu/~gohlke/pythonlibs/下载对应的本地模块安装包（图片标黄为所下载的模块包）



Annoy: approximate nearest neighbors optimized for memory usage and loading/saving to disk.

- annoy-1.17.0-pp37-pypy37_pp73-win_amd64.whl
- annoy-1.17.0-cp310-cp310-win_amd64.whl
- annoy-1.17.0-cp310-cp310-win32.whl
- annoy-1.17.0-cp39-cp39-win_amd64.whl
- annoy-1.17.0-cp39-cp39-win32.whl
- annoy-1.17.0-cp38-cp38-win_amd64.whl
- annoy-1.17.0-cp38-cp38-win32.whl
- annoy-1.17.0-cp37-cp37m-win_amd64.whl
- annoy-1.17.0-cp37-cp37m-win32.whl
- annoy-1.17.0-cp36-cp36m-win_amd64.whl
- annoy-1.17.0-cp36-cp36m-win32.whl

2、进行本地安装

```
PowerShell
PowerShell 7.2.2
Copyright (c) Microsoft Corporation.

https://aka.ms/powershell
Type 'help' to get help.

PS C:\Users\f1241> pip install C:\Users\f1241\Downloads\annoy-1.17.0-cp310-cp310-win_amd64.whl
```

```
PowerShell
PowerShell 7.2.2
Copyright (c) Microsoft Corporation.

https://aka.ms/powershell
Type 'help' to get help.

PS C:\Users\f1241> pip install C:\Users\f1241\Downloads\annoy-1.17.0-cp310-cp310-win_amd64.whl
>>
Looking in indexes: https://pypi.tuna.tsinghua.edu.cn/simple
Processing c:\users\f1241\downloads\annoy-1.17.0-cp310-cp310-win_amd64.whl
Installing collected packages: annoy
Successfully installed annoy-1.17.0
PS C:\Users\f1241>
```

3、调用模块

```
import annoy
[1]  ✓ 0.4s                                Python

[ ]                                        Python
```

2、两个类的编写练习

1. 第一个类是单位转换方法有：（1）kg转g、（2）kg转t、（3）t转g

```python
class UnitConverter: #建立类
    function = 'UnitConverter' # 类的属性
    def __init__(self,function,unit ):
        self.function = function
        self.unit = unit
    def kg_g(self, x):
        result = x*1000
        self.unit = 'g'
        return str(result) + self.unit
```

```
    def kg_t(self, x):
        result = x/1000
        self.unit = 't'
        return str(result) + self.unit
    def t_g(self, x):
        result = x*1000*1000
        self.unit = 'g'
        return str(result) + self.unit
y = UnitConverter('UnitConverter','unit')#调用
print(y.function)
print(y.kg_g(2))#取值
print(y.kg_t(2000))
print(y.t_g(2))
#x = input()# 无法实现数值合理转化
#y = UnitConverter('UnitConverter','unit')
#print(y.function)
#print(y.kg_g(x))#
#print(y.kg_t(x))
#print(y.t_g(x))
```

结果

> UnitConverter
> 2000g
> 2.0t
> 2000000g

2. 第二个类是数学计算方法有：（1）$y=x^2$、（2）$y=x^3$、（3）$y=x^4$；

```
class ChengFang:  # 建立类
    Function = 'Function'  # 类的属性
    def __init__(self,function,form):
        self.function = function
        self.form = form
    def cifang2(self,x):
        result = x*x
        self.form = 'x^2 = '
        return self.form + str(result)
    def cifang3(self,x):
        result = x*x*x
        self.form = 'x^3 = '
        return self.form + str(result)
    def cifang4(self,x):
```

```
        result = x*x*x*x
        self.form = 'x^4 = '
        return self.form + str(result)
y = ChengFang('Function','form')#调用
print(y.function)
print(y.cifang2(2)) #数字2为计算的基底
print(y.cifang3(2))
print(y.cifang4(2))
```

结果

> Function
> x^2 = 4
> x^3 = 8
> x^4 = 16