```python
"""
date:20211209
describe:绘图
"""
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import os
import math

```

这个绘图的，不用管

```python
def PicSave(picPath="image", picName="defaultPic", figExtension="png", tight_layout=True, resolution=300):
    """
    describe: 该函数用于保存图片，默认保存在当前目录。
    parameters:
    picPath: 保存图片的目录名称
    picName: 保存图片名字
    figExtension: 保存图片扩展名，如png
    tight_layout: 自动调整子图参数，使之填充整个图像区域
    resolution: 分辨率，默认取值300
    :return: “Picture Save Successfully!”
    """
    myPicPath = os.path.join('.', picPath)
    os.makedirs(myPicPath, exist_ok=True)
    print("Saving Picture : ", picName)
    if tight_layout:
        plt.tight_layout()
    plt.savefig(os.path.join('.' + picPath + picName + '.' + figExtension), format=figExtension, dpi=resolution)
    print("Picture Save Successfully!")

class mySolution:
```

用于求空间某一点相对于线圈的磁场强度

```python

    # 定义问题参数
    def __init__(self, inputNMAX, inputZaShu, inputNeiJing, inputZaJianJU, inputDianLiu, inputCiDaoLiv, inputBeta, inputAlpha, inputXigema, inputR):
        self.NMAX = inputNMAX
        self.ZaShu = inputZaShu
        self.NeiJing = inputNeiJing
        self.ZaJianJiu = inputZaJianJU
        self.DianLiu = inputDianLiu
        self.CiDaoLiv = inputCiDaoLiv
        self.Beta = inputBeta
        self.Alpha = inputAlpha
        self.Xigema = inputXigema
        self.R = inputR
```

这个函数用于接收参数

```python

    # 求解B
    def solution(self):
        BX = 0
        BY = 0
        BZ = 0
```

这个函数用于求解，结果输出三个方向的磁场强度

```python
        for n in range(self.NMAX):
            t1 = (n - 0.5) * 2 * math.pi * self.ZaShu / self.NMAX

            a = self.ZaJianJiu * t1 / 2 * math.pi + self.NeiJing
            temp = math.sqrt(math.pow(self.R * math.sin(self.Alpha)*math.cos(self.Xigema) - a * math.sin(self.Beta) * math.cos(t1), 2) +
                             pow((self.R * math.sin(self.Alpha) * math.sin(self.Xigema) - a * math.sin(self.Beta) * math.sin(t1)), 2) +
                             pow((self.R * math.cos(self.Alpha) - a * math.cos(self.Beta)), 2))
            # otherParame = self.CiDaoLiv * self.DianLiu * a / 2 / self.NMAX
            BX = BX + (self.R * math.cos(self.Alpha) - a * math.cos(self.Beta)) * a * math.sin(t1)
            BY = BY + (self.R * math.cos(self.Alpha)  - a * math.cos(self.Beta)) * a * math.cos(t1)
            BZ = BZ +((self.R * math.sin(self.Alpha) * math.sin(self.Xigema) - a * math.sin(self.Beta) * math.sin(t1))*(-1 * a * math.cos(t1)) -
```

循环范围

```
                      (self.R * math.sin(self.Alpha) * math.cos(self.Xigema) - a * math.sin(self.Beta) * math.cos(t1))*( a * math.sin(t1)))

        BX = BX * self.CiDaoLiv * self.DianLiu / 2 / self.NMAX
        BY = BY * self.CiDaoLiv * self.DianLiu / 2 / self.NMAX
        BZ = BZ * self.CiDaoLiv * self.DianLiu / 2 / self.NMAX
        print(BX, BY, BZ)

inputNMAX = 1000
inputZaShu = 10
inputNeiJing = 50
inputZaJianJu = 2
inputDianLiu = 1
inputCiDaoLv = 4*math.pi*1e-7
inputBeta = math.pi / 2
inputAlpha = math.pi / 2
inputXigema = math.pi / 2
inputR = 50

# inputNMAX, inputZaShu, inputNeiJing, inputZaJianJU, inputDianLiu, inputCiDaoLiv, inputBeta, inputAlpha, inputXigema, inputR)
sh_solution = mySolution(inputNMAX, inputZaShu, inputNeiJing, inputZaJianJu, inputDianLiu, inputCiDaoLv, inputBeta, inputAlpha, inputXigema, inputR)
sh_solution.solution()
del sh_solution
```

这个增加测试的，目前不用管

```
# 在平面取一些点，进行测试
t = []
t.append(0)
minValue = 0
maxValue = 2 * math.pi * (inputZaShu + 2)
while True:
    minValue = minValue + (maxValue/1000)
    if minValue >= maxValue:
        break
    t.append(minValue)

x = [(inputZaJianJu * index / (2*math.pi) + inputNeiJing) * math.cos(index) for index in t]
y = [(inputZaJianJu * index / (2*math.pi) + inputNeiJing) * math.sin(index) for index in t]
z = [ 10 for index in t]
"""plt.plot(x,y)
plt.show()"""

rList = [48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68]
xigemaList = [(math.pi*2)*(index/99) for index in range(100)]


for r in rList:
    for xigema in xigemaList:
        z = 10
        inputR = math.sqrt(math.pow(z, 2) + math.pow(r, 2))
        inputXigema = xigema
```