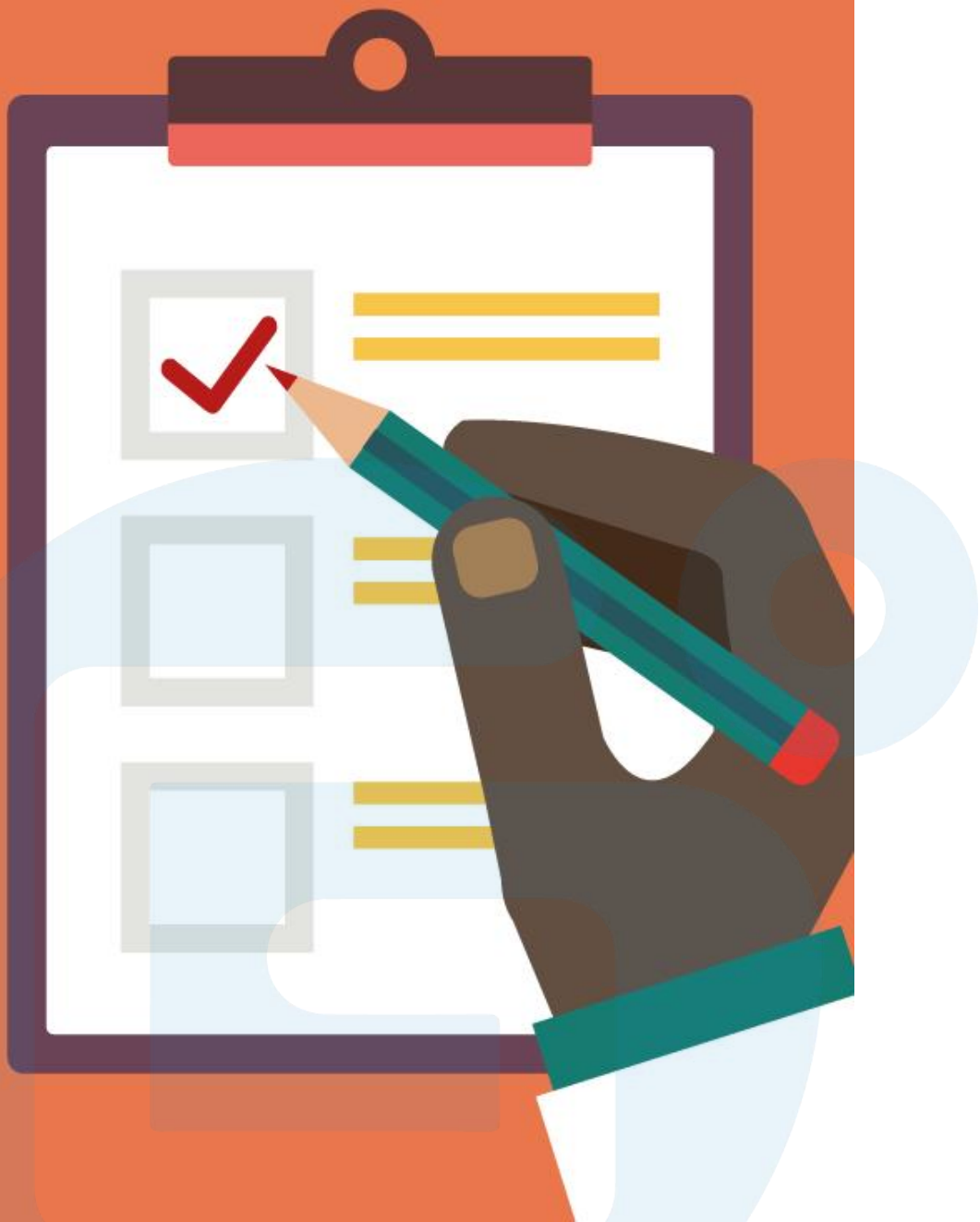


Talk is cheap, show me the code

第六课：Python数值计算练习课

Python初阶入门课程系列



OUTLINE

- 数值计算的概念
 - 随机数
 - 数列循环
 - 作业
-



一 数值计算的概念





数值计算的概念

数值计算的定义

- 数值计算指有效使用数字计算机求数学问题近似解的方法与过程，以及由相关理论构成的学科。数值计算主要研究如何利用计算机更好的解决各种数学问题，包括连续系统离散化和离散形方程的求解，并考虑误差、收敛性和稳定性等问题。
 - 数值运算的研究领域包括数值逼近、数值微分和数值积分、数值代数、最优化方法、常微分方程数值解法、积分方程数值解法、偏微分方程数值解法、计算几何、计算概率统计等。
 - 计算物理、计算力学、计算化学、计算经济学等都可归结为数值计算问题。
-

数值计算的概念

编程的一个重要应用方向就是数值计算。

- Python语言具有一系列的函数，库，包，模块用于实现数值计算。
 - 数值计算也是数据分析，机器学习，图像识别等高阶应用的基础。
-

数值计算的定义

数值计算具有以下5个重要特征：

- 数值计算的结果是离散的，并且一定有误差，这是数值计算方法区别与解析法的主要特征。
 - 注重计算的稳定性。控制误差的增长势头，保证计算过程稳定是数值计算方法的核心任务之一。
 - 注重快捷的计算速度和高计算精度是数值计算的重要特征。
 - 注重构造性证明。
 - 数值计算主要是运用有限逼近的思想来进行误差运算。
-

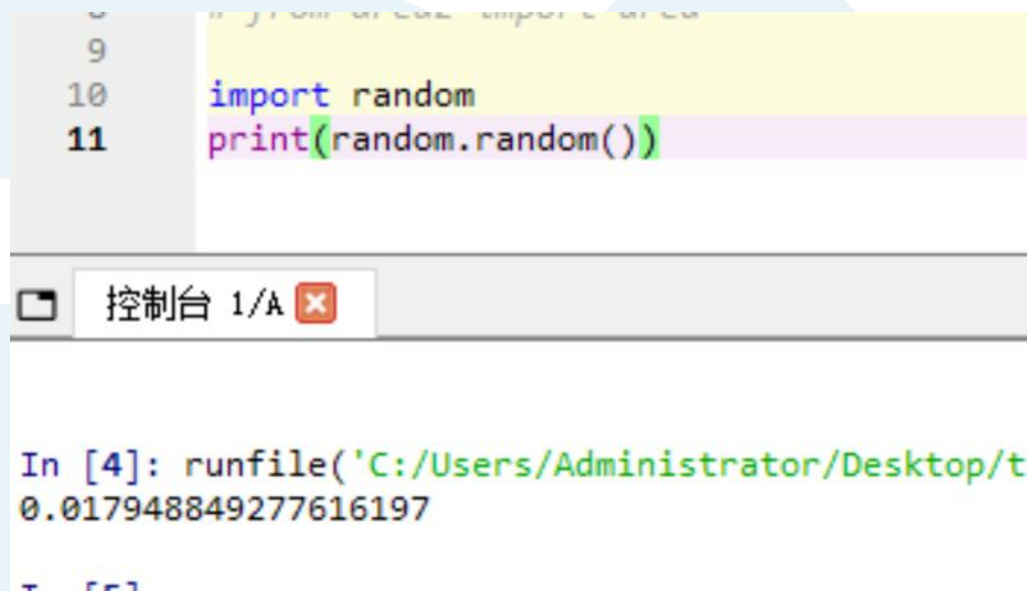


二 随机数



随机数用于模拟现象的不确定性特征

- 随机数在数值计算中起到筛选、初始化、实现算法等多重功能。
- Python里使用random模块实现生成随机数的功能。调用`random.random()`生成一个半开半闭区间 $[0,1)$ 内的随机数



The screenshot shows a Python IDE with a code editor and a console window. The code editor contains the following code:

```
9  
10 import random  
11 print(random.random())
```

The console window, titled "控制台 1/A", shows the output of the code:

```
In [4]: runfile('C:/Users/Administrator/Desktop/t  
0.017948849277616197
```


随机数

随机数种子

- 并非真正的随机数，而是用随机数种子按照某种特定算法指定的伪随机数。
- 也就是说，一旦指定了种子不变，则生成的随机数也不会变化。

```
10 import random
11 random.seed(2)
12 print(random.random())
13 print(random.random())
14 print('')
15
16 random.seed(3)
17 print(random.random())
18 print(random.random())
19 print(random.random())
20 print('')
21
22 random.seed(2)
23 print(random.random())
24 print(random.random())
25 print('')
26
27 random.seed(3)
28 print(random.random())
29 print(random.random())
```

控制台 1/A

```
In [12]: runfile('C:/Users/Administrator/Desktop/RandomNumber.py', wdir='C:/Users/Administrator/Desktop')
0.9560342718892494
0.9478274870593494

0.23796462709189137
0.5442292252959519
0.36995516654807925

0.9560342718892494
0.9478274870593494

0.23796462709189137
0.5442292252959519
```

随机数生成

- `random.random`生成的随机数是在0,1之间均匀分布的。
- 利用`random.uniform(a,b)`可以生成在[a,b)之间均匀分布的随机数。

```
8 import random
9 a=random.uniform(10, 100)
10 b=[random.uniform(10, 100) for i in range(20)]
11 print(a)
12 print(b)
```

控制台 1/A

```
In [19]: runfile('C:/Users/Administrator/Desktop/test/aaaa.py', wdir='C:/Users/Administrator/Desktop/test')
51.84044758875809
[38.66186150683097, 44.201342971064044, 90.26105120454586, 57.31774922314255, 60.4459324923849, 31.251106640355587,
12.147227122670397, 39.2628635885044, 22.302765368781998, 55.920146125348104, 99.88152113732968, 70.70317276112831,
26.365914714082994, 90.42143829246896, 81.70839292794756, 76.09615227045799, 91.59342849078048, 78.65969354497639,
81.0772873715587, 41.840828005744314]
```

随机数

随机数计算平均值和方差

- 利用while循环和if循环，计算生成的随机数的平均数和标准差。

```
7  # -*- coding: utf-8 -*-
8  import random as rd
9  from math import sqrt
10
11  sm=0;sv=0 #和与平方和
12  i=1
13  while i<5000:
14      a=rd.random()
15      sm=sm+a
16      sv=sv+a**2
17
18      mean=sm/i
19      smean=sqrt(sv/i-mean**2)
20
21      i=i+1
22
23      if (i%100)==0:
24          print(i,mean,smean)
```

控制台 1/A

```
1700 0.5014517545808844 0.28899908897110017
1800 0.5001105597247096 0.2894659163446706
1900 0.4991258305034659 0.28965791242406297
2000 0.4976031941845126 0.28972553490683567
2100 0.49897718392593615 0.28930301987739193
2200 0.49935582994339556 0.288398564710583
2300 0.4997560302307857 0.2879947360620657
2400 0.5024699853659027 0.2881733128882203
2500 0.5031959709689503 0.2875853573073713
2600 0.5042645324922418 0.28732031464282276
2700 0.505534687206593 0.28792200382271405
2800 0.5050095315311887 0.2875100529707975
2900 0.5052596715826909 0.2874636660755175
3000 0.5039472138635571 0.28744027713354153
3100 0.5043766938296864 0.28730759990834004
3200 0.5040417490366786 0.287341275066228
```

随机数

随机整数

- 一种方法是利用区间表示整数

```
8 import random as rd
9 for i in range(10):
10     a=rd.random()
11     if a<0.25:
12         b=1
13     elif a<0.5:
14         b=2
15     elif a<0.75:
16         b=3
17     else:
18         b=4
19     print(b)
```

控制台 1/A

```
In [39]: runfile('C:/Users/Administrator/D
3
2
2
1
4
3
4
3
2
3
```

随机整数

- 另外一种方法是利用randint函数
- 需要注意的是randint(a,b)的取值区间是[a,b],也就是说两端都是闭区间

```
7  #%%
8  import random as rd
9  for i in range(10):
10     a=rd.randint(1,10)
11     print(a)
```

控制台 1/A

```
In [44]: runfile('C:/Users/Administrat
6
7
10
4
2
1
6
9
10
5
```

模拟扔骰子

- 模拟丢单个骰子的函数，并且统计

出现6点的次数

```
7  ###
8  import random as rd
9  def six_eyes(N):#N为次数
10     S=0
11     for i in range(N):
12         a=rd.randint(1,6)
13         if a==6:
14             S=S+1
15     return S,S/N
16
17 print(six_eyes(1000))
18 print(1/6)
```

控制台 1/A

```
In [54]: runfile('C:/Users/Administrator/Desktop,
(168, 0.168)
0.16666666666666666
```

```
In [55]: runfile('C:/Users/Administrator/Desktop,
(159, 0.159)
0.16666666666666666
```

```
In [56]: runfile('C:/Users/Administrator/Desktop,
(158, 0.158)
0.16666666666666666
```

```
In [57]: runfile('C:/Users/Administrator/Desktop,
(155, 0.155)
0.16666666666666666
```

```
In [58]: runfile('C:/Users/Administrator/Desktop,
(153, 0.153)
0.16666666666666666
```

```
In [59]: runfile('C:/Users/Administrator/Desktop,
(165, 0.165)
0.16666666666666666
```

模拟扔骰子

- 利用大量随机数来估计概率的方法被称为蒙特卡洛模型，常用于评估科学问题

例子

两个骰子，一个是黑色的，另一个是红色的，同时扔两个骰子，黑骰子比红骰子大的几率是多少？

这个问题可以纯用概率学进行分析，也可以直接用蒙特卡洛法数值求解。

两个骰子，一个是黑色的，另一个是红色的，同时扔两个骰子，黑骰子比红骰子大的几率是多少？

模拟扔骰子

- 利用蒙特卡罗法

```
7  ###
8  import random as rd
9  N=10000
10 s=0
11 for i in range(1,N):
12     black=rd.randint(1,6)
13     red=rd.randint(1,6)
14     if black>red:
15         s=s+1
16
17     if i%100==0:
18         print(i,s/i)
19
```

```
7000 0.4133743307743307
7900 0.41417721518987344
8000 0.414375
8100 0.4141975308641975
8200 0.4146341463414634
8300 0.41493975903614455
8400 0.4158333333333333
8500 0.41623529411764704
8600 0.41627906976744183
8700 0.41620689655172416
8800 0.41670454545454544
8900 0.4166292134831461
9000 0.41744444444444445
9100 0.4174725274725275
9200 0.41706521739130437
9300 0.4174193548387097
9400 0.41712765957446807
9500 0.41778947368421054
9600 0.41739583333333335
9700 0.41835051546391755
9800 0.4176530612244898
9900 0.4174747474747475
```


两个骰子，一个是黑色的，另一个是红色的，同时扔两个骰子，黑骰子比红骰子大的几率是多少？

模拟扔骰子

- 利用嵌套循环进行直接计算

```
0
7   #%%
8   A=0;B=0 #计数器
9   for i in range(1,7):
10      for j in range(1,7):
11          if i>j:
12              A=A+1
13              B=B+1
14   print(A/B)
15
16
17
```

控制台 1/A

```
In [69]: runfile('C:/Users/Administ
0.4166666666666667

In [70]:
```

模拟人口变化

考虑到计划生育政策在执行的过程中，有人在生出女孩之后会继续生孩子，直到生出男孩为止。这种行为对性别比例的影响是多少？

```
7  ###
8  import random as rd
9  male=0;female=0;p=0
10 N=10000
11 for i in range(N):
12     A=rd.random()
13     while A<0.5: #小于0.5表示是女孩
14         female=female+1
15         A=rd.random()
16     male=male+1 #显然，按照现有策略，会稳定产出N个男孩
17 p=male+female
18 print(female/p)
19
```

控制台 1/A

```
In [83]: runfile('C:/Users/Administrator/Desktop/test/aaaa.py',
0.49708308187487427
```

```
In [84]: runfile('C:/Users/Administrator/Desktop/test/aaaa.py',
0.5015948963317385
```

```
In [85]: runfile('C:/Users/Administrator/Desktop/test/aaaa.py',
0.5016942395854096
```

```
In [86]: runfile('C:/Users/Administrator/Desktop/test/aaaa.py',
0.49713366187267427
```

```
In [87]: runfile('C:/Users/Administrator/Desktop/test/aaaa.py',
0.49680470990791525
```

模拟人口变化

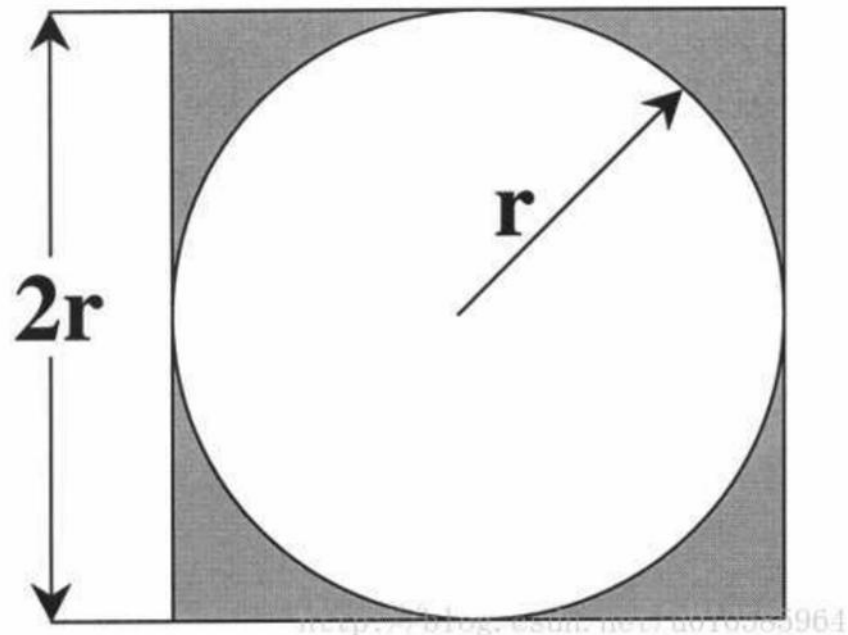
考虑到计划生育政策在执行的过程中，有人在生出女孩之后会继续生孩子，直到生出男孩为止。这种行为对性别比例的影响是多少？

- 可以看到这种行为并不会导致性别失调。
- 根据大数据显示，导致性别失调的主要因素是终止妊娠，人工选择性别，弃养女婴等非法行为。
- 要认识到重男轻女思想的危害和愚昧性质，关心和照顾妇女儿童，尤其是女童。

计算 π

如何利用随机数来计算 π 的近似值？

蒙特卡罗法，在方接圆里扔小球。



计算 π

```
8 import random as rd
9 N=1000000
10 A=0
11 for i in range(N):
12     x=rd.uniform(-1,1)
13     y=rd.uniform(-1,1)
14     S=x**2+y**2
15     if S<=1:
16         A=A+1
17 pai=A/N*4
18 print(pai)
```

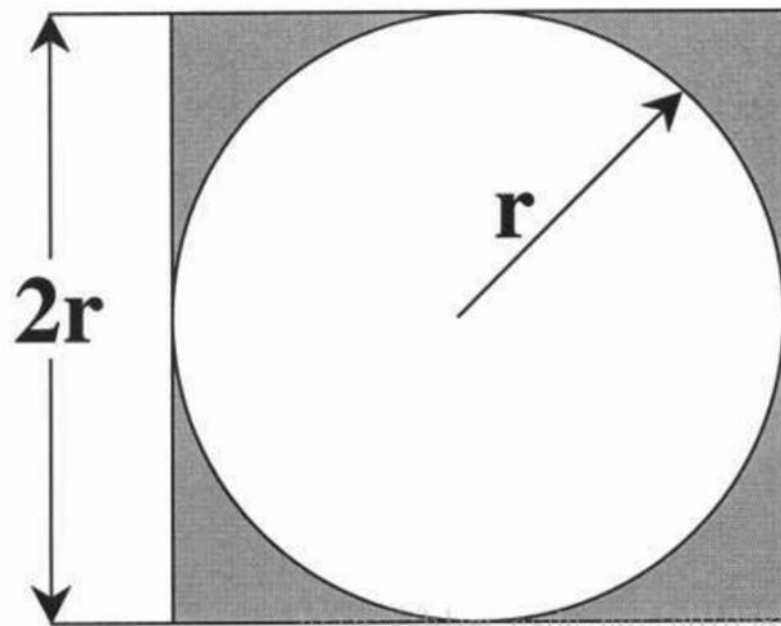
控制台 1/A

In [102]: runfile('C:/Users/Administrator/
3.14746

In [103]: runfile('C:/Users/Administrator/
3.143204

In [104]: runfile('C:/Users/Administrator/
3.140892

In [105]: runfile('C:/Users/Administrator/
3.143716





三 数列循环



数列循环

一个常见的例子

银行给定的一年期存款利息为2.25%，每年记息且自动转存。
如果不满一年取出，则按活期利息计算，年息0.35%，日息需要除365，而且不计复息。
设计一个计算程序，计算一笔存款在6年零215天后的总额。

```
7  ###
8  year=6
9  day=215
10 interest_rate=0.021
11 d_interest_rate=0.35/100/365
12 money=10000
13
14 for i in range(year):
15     money=money*(1+interest_rate)
16
17 money=money*(1+d_interest_rate*day)
18 print(money)
19
```

控制台 1/A

```
In [114]: runfile('C:/Users/Administrator/Desktop/
11351.385984608745
```

数列循环

- 实际上，你不会这样操作，因为如果下定决心存5年以上，你会选择5年定存，利息可以给到3.25%。因此更改程序。

```
7  ###
8  year=6
9  day=215
10 interest_rate_1=0.021
11 interest_rate_5=0.0325
12 d_interest_rate=0.35/100/365
13 money=10000
14
15 if year>=5:
16     money=money*(1+interest_rate_5)**5
17     money=money*(1+interest_rate_1)**(year-5)
18     money=money*(1+d_interest_rate*day)
19 else:
20     money=money*(1+interest_rate_1)**year
21     money=money*(1+d_interest_rate*day)
22
23 print(money)
```

控制台 1/A

```
In [111]: runfile('C:/Users/Administrator/Desktop/test/aa')
12005.229937964532
```


数列循环

- 如果你有足够的冒险精神，敢于挑战股票市场之类的高风险理财项目，复利的效应就会变得更加显著。
- 假设你每年都能拿到20%的高收益率。（其实也就是连拉两个涨停板，运气好的话就是两天的事）

```
7  ###
8  year=20
9  interest_rate=0.2
10 money=1e5 #十万元
11
12 for i in range(year):
13     money=money*(1+interest_rate)
14
15 print(money)
16
```

控制台 1/A

```
In [121]: runfile('C:/Users/Administrator/De
3833759.9924474712
```

10万元在20年后，
变成了383万元！

数列循环

- 现在，让我们不那么乐观一些，考虑到有赚有赔，假设每天涨和跌的几率是相等的。
- 可以看到，在维持了1000天的投资过程中，重复100次的结果显示，有80%的几率是赔本的。
- 所以，股市有风险，投资需谨慎。
- 想一想，为什么？
(赚10%和赔10%是一个概念吗？)

```
8 print(10000*1.1*0.9)
```

控制台 1/A

```
In [139]: runfile('C:/Users/Adminis
9900.0
```

```
7  ###
8  import random as rd
9  final=[]
10 for i in range(100):
11     money=1e5
12     for j in range(1000):
13         rate=rd.uniform(-0.1,0.1)
14         money=money*(1+rate)
15         final.append(money)
16
17 a=b=c=d=e=f=0
18 for i in range(100):
19     if final[i]<10000:
20         a=a+1
21     elif final[i]<50000:
22         b=b+1
23     elif final[i]<1e5:
24         c=c+1
25     elif final[i]<2e5:
26         d=d+1
27     elif final[i]<5e5:
28         e=e+1
29     else:
30         f=f+1
31 print(a,b,c,d,e,f)
```

控制台 1/A

```
In [135]: runfile('C:/Users/Administrator/Desktop
33 34 13 11 5 4
```

数列循环

两分法

考虑一个复杂的非线性函数的解。
 $\lg x + x - 3 = 0$

```
8 import math
9 x=1
10 y=math.log10(x)+x-3
11 print(y)
12
13 x=10
14 y=math.log10(x)+x-3
15 print(y)
16
17 a=1;b=10
18 c=(a+b)/2
19 y=math.log10(c)+c-3
20 while abs(y)>0.01:
21     c=(a+b)/2
22     y=math.log10(c)+c-3
23     if y<-0.01:
24         a=c
25     elif y>0.01:
26         b=c
27     else:
28         print(c)
29         print(y)
30
```

控制台 1/A

```
In [152]: runfile('C:/Users/Administrat
-2.0
8.0
2.58203125
-0.006007255826209423
```

```
8 import math
9 x=1
10 y=math.log10(x)+x-3
11 print(y)
12
13 x=10
14 y=math.log10(x)+x-3
15 print(y)
16
17 a=1;b=10
18 c=(a+b)/2
19 y=math.log10(c)+c-3
20 while abs(y)>0.0001:
21     c=(a+b)/2
22     y=math.log10(c)+c-3
23     if y<-0.0001:
24         a=c
25     elif y>0.0001:
26         b=c
27     else:
28         print(c)
29         print(y)
30
```

控制台 1/A

```
In [155]: runfile('C:/Users/Admin
-2.0
8.0
2.587249755859375
8.811049629420253e-05
```

数列循环

泰勒展开

$$\sin x = +\frac{x^1}{1!} - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \dots$$

```
7  ###
8  import math
9  N=50
10 x=math.pi/6
11 sum=0
12 for i in range(N):
13     a=math.factorial(2*i+1)
14     # print(a)
15     b=(-1)**i*x**(2*i+1)/a
16     # print(b)
17     sum=sum+b
18 print(sum)
```

控制台 1/A

```
In [175]: runfile('C:/Users/Administrat
0.49999999999999994
```

```
7  ###
8  import math
9  N=50
10 x=math.pi/3
11 sum=0
12 for i in range(N):
13     a=math.factorial(2*i+1)
14     # print(a)
15     b=(-1)**i*x**(2*i+1)/a
16     # print(b)
17     sum=sum+b
18 print(sum)
19 print(math.sqrt(3)/2)
```

控制台 1/A

```
In [178]: runfile('C:/Users/Administrator/Des
0.8660254037844385
0.8660254037844386
```

牛顿迭代法

$$f(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{f''(x_0)(x - x_0)^2}{2!} + \cdots + \frac{f^{(n)}(x_0)(x - x_0)^n}{n!} + R_n(x)$$

$$f(x_0) + f'(x_0)(x - x_0) = 0$$

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

数列循环

牛顿迭代法

```
7  """
8  def f(x):
9      return (x-3)**3      #定义 f(x) = (x-3)^3
10
11 def fd(x):
12     return 3*((x-3)**2)   #定义 f'(x) = 3*((x-3)^2)
13
14 def newtonMethod(n,assum): #n为计数器, assum为初值假设
15     time = n
16     x = assum
17     Next = 0
18     A = f(x)
19     B = fd(x)
20     print('A = ' + str(A) + ',B = ' + str(B) + ',time = ' + str(time))
21     if f(x) == 0.0:
22         return time,x
23     else:
24         Next = x - A/B
25         print('Next x = ' + str(Next))
26         if abs(A - f(Next)) < 1e-6:
27             print('Meet f(x) = 0,x = ' + str(Next)) #设置迭代跳出条件, 同时输出满足f(x) = 0的x值
28         else:
29             return newtonMethod(n+1,Next)
30
31 newtonMethod(0,4)
```

```
A = 0.020012294675/40913,B = 0.2033/4403390/0/0,time = 3
Next x = 3.197530864197531
A = 0.007707346629258938,B = 0.11705532693187012,time = 4
Next x = 3.1316872427983538
A = 0.002283658260521159,B = 0.052024589747497726,time = 5
Next x = 3.0877914951989025
A = 0.0006766394845988619,B = 0.023122039887776766,time = 6
Next x = 3.058527663465935
A = 0.0002004857732144761,B = 0.010276462172345176,time = 7
Next x = 3.0390184423106232
A = 5.9403192063548476e-05,B = 0.0045673165210423005,time = 8
Next x = 3.0260122948737487
A = 1.7600945796606656e-05,B = 0.002029918453796555,time = 9
Next x = 3.0173415299158326
A = 5.21509505084655e-06,B = 0.0009021859794651509,time = 10
Next x = 3.0115610199438883
A = 1.5452133483989184e-06,B = 0.0004009715464289457,time = 11
Next x = 3.007707346629259
A = 4.578409921182244e-07,B = 0.00017820957619064937,time = 12
Next x = 3.0051382310861725
Meet f(x) = 0,x = 3.0051382310861725
```

牛顿迭代法

- 复杂方程可能与遇到不收敛的问题，与两分法不一样，牛顿法并非绝对收敛的，但是牛顿法是二阶收敛，意味着其收敛速度比两分法快很多。
- 通常利用一些收敛速度较慢但稳定的方法，两分法或者割线法收敛至真解附近，再用该值作为初值用牛顿法进行快速迭代求解。



四 作业



写出一个非线性方程，分别用两分法和牛顿法求解，
利用python写出代码，比较两者之间的迭代速度。



感谢参与 下堂课见

