**计算方法作业**

## 01. 二分法求利率（非线性方程求解）

【问题描述】如果在240个月内每月付款300美元，求解满足全部年金A为500000美元的利率I的近似值（精确到小数点后10位）

【输入形式】在屏幕上输入3个数，分别表示左端点值、右端点值和精确到小数点后的位数。各数间都以一个空格分隔。

【输出形式】第一行输出迭代次数，第二行输出利率（精确到小数点后11位）

【样例输入】

0.15 0.16 10

【样例输出】

27

0.15753931027

【样例说明】输入：左端点值为0.15，右端点值为0.16，求得的利率精确到小数点后10位。输出：第一行为迭代次数27次，第二行为求得的利率为0.15753931027

【评分标准】根据输入得到的输出准确

1. # -\*- coding: utf-8 -\*-
2. # 第一次作业 - 二分法求利率（非线性方程求解）
3. **import** math
5. **def** f(x):
6. p=300
7. n=240
8. A=12\*p\*((1+x/12)\*\*n-1)/x-500000
9. **return** A
11. **def** regula(a, b, accuracy):
12. delta = 0.5 \* 10\*\*(-accuracy)
13. n = math.floor( (math.log(b-a) - math.log(delta))\*1.0 / math.log(2) )
14. inter\_a = a
15. inter\_b = b
16. **if** f(inter\_a) \* f(inter\_b) > 0 :
17. **print**("ya,yb are not suitable ")
18. **return**
19. **for** k **in** range(1000000):
20. c = (inter\_a + inter\_b) / 2
21. **if** f(c) == 0:
22. **break**
23. **elif** f(inter\_a)\*f(c) < 0:
24. inter\_b = c
25. c = (inter\_a + inter\_b) / 2
26. **else**:
27. inter\_a = c
28. c = (inter\_a + inter\_b) / 2
29. err = abs(inter\_b - inter\_a) / 2
30. **if** err < delta:
31. **break**
32. # n = math.floor( (math.log(b-a) - math.log(0.5\*10\*\*(-delta)))\*1.0 / math.log(2) )
33. **return** (n, round(c, accuracy+1))
35. **def** main():
36. # left, right, accuracy = map(float, input().split())
37. left, right, accuracy = input().split()
38. left = float(left)
39. right = float(right)
40. accuracy = int(accuracy)
41. result = regula(left, right, accuracy)
42. **print**(result[0])
43. **print**(result[1])
45. **if** \_\_name\_\_ == '\_\_main\_\_':
46. main()

## 02. 试值法求利率（非线性方程求解）

【题目简述】试值法求利率（非线性方程求解）

【问题描述】如果在240个月内每月付款300美元，求解满足全部年金A为500000美元的利率I的近似值

【输入形式】在屏幕上输入3个数，分别表示左端点值、右端点值和由所求利率得到的年金误差。各数间都以一个空格分隔。

【输出形式】第一行输出迭代次数，第二行输出利率（精确到小数点后11位）

【样例输入】

0.15 0.16 0.01

【样例输出】

3

0.15753930922

【样例说明】输入：左端点值为0.15，右端点值为0.16，由所求利率得到的年金误差为0.01美元。输出：第一行为迭代次数3次，第二行为求得的利率为0.15753930922

【评分标准】根据输入得到的输出准确

1. # -\*- coding: utf-8 -\*-
2. # 第二次作业 - 试值法求利率（非线性方程求解）
3. **import** math
5. # 年金计算函数
6. **def** f(x):
7. p = 300
8. n = 240
9. A = 12\*p\*((1+x/12)\*\*n-1)/x-500000
10. **return** A
12. # a为左端点值，b为右端点值，accuracy为给定误差
13. **def** FalsePosition(a, b, accuracy):
14. # 如果f(a)\*f(b) > 0，此方法不适用
15. **if** f(a)\*f(b) > 0 :
16. **print**("This method is not suitable ")
17. **return**
18. err = 100000
19. n = 0
20. # while循环
21. **while** (err > accuracy):
22. c = b - f(b)\*(b-a) / ( f(b)-f(a) )
23. **if** f(c) == 0:
24. **break**
25. **elif** f(a)\*f(c) < 0:
26. b = c
27. c = b - f(b)\*(b-a) / ( f(b)-f(a) )
28. **else**:
29. a = c
30. c = b - f(b)\*(b-a) / ( f(b)-f(a) )
31. err = abs(f(c))
32. n = n + 1
33. **return** (n, round(c, 11))

36. # main函数
37. **def** main():
38. left, right, accuracy = map(float, input().split())
39. result = FalsePosition(left, right, accuracy)
40. **print**(result[0])
41. **print**(result[1])
43. **if** \_\_name\_\_ == '\_\_main\_\_':
44. main()

## 03. 不动点迭代法求函数根（非线性方程求解）

【问题描述】寻找方程x\*\*5-2\*x-1=0的根的近似值位置，然后使用不动点迭代法求方程的根。前后两次迭代的差的绝对值小于delta后停止迭代。

【输入形式】在屏幕上输入3个数，分别表示左端点值、右端点值和所求根的精度。各数间都以一个空格分隔。

【输出形式】每一行输出一个根（精确到小数点后3位）

【样例1输入】

-1.2 1.5 1

【样例1输出】

-1.012

-0.516

1.296

【样例1说明】输入：左端点值为-1.2，右端点值为1.5，前后两次迭代的差的绝对值小于0.1后停止迭代。输出：从小到大顺序输出三个根的值。

【样例2输入】

-1.2 1.5 3

【样例2输出】

-1.000

-0.519

1.291

【样例2说明】输入：左端点值为-1.2，右端点值为1.5，前后两次迭代的差的绝对值小于0.001后停止迭代。输出：从小到大顺序输出三个根的值。

【评分标准】根据输入得到的输出准确

【提醒】提醒下同学，在做这次作业时，approot程序中使用的epsilon为0.01，x = np.linspace(a, b, 9)。这次的作业先要用书上2.3节程序2.4粗略估算得到三个根的起始点p0，然后再用不动点迭代法求根

## 04. 牛顿法解投射体问题（非线性方程求解）

【问题描述】在考虑空气阻力情况下，求解投射体撞击地面时经过的时间和水平飞行行程，其中：y=f(t)=9600\*(1-e\*\*(-t/15.0)) - 480\*t；x=r(t)=2400\*(1-e\*\*(-t/15.0))。

【输入形式】在屏幕上输入3个数，分别表示起始值、前后两次迭代的差的绝对值精度和函数值的精度。各数间都以一个空格分隔。

【输出形式】第一行输出投射体撞击地面时经过的时间，第二行输出水平飞行行程，精确到小数点后5位。

【样例1输入】

8 1 1

【样例1输出】

9.08955

1090.69211

【样例1说明】输入：起始值为8、前后两次迭代的差的绝对值精度为0.1和函数值的精度为0.1。输出：第一行输出投射体撞击地面时经过的时间为9.08955秒，第二行输出水平飞行行程为1090.69211ft。

【样例2输入】

8 3 3

【样例2输出】

9.08790

1090.54798

【评分标准】根据输入得到的输出准确

## 05. 高斯消去法（线性方程组求解）

【问题描述】为求解一个线性方程组，首先构造增广矩阵[A|B]，采用偏序选主元策略的高斯消去法变换成上三角矩阵，再执行回代过程得到解。

【输入形式】在屏幕上依次输入方阵阶数n，系数矩阵A和常数矩阵B。

【输出形式】每一行输出X的一个解。

【样例1输入】

4

1 2 1 4

2 0 4 3

4 2 2 1

-3 1 3 2

13

28

20

6

【样例1输出】

[[3.]

[-1.]

[4.]

[2.]]

【样例1说明】输入：第1行为方阵阶数4，第2行至5行为系数矩阵A，第6行至9行为常数矩阵B。输出：每行依次输出方程解：x1, x2, x3, x4。

【评分标准】根据输入得到的输出准确

## 06. 三角分解法（线性方程组求解）

【问题描述】为求解一个线性方程组，首先采用偏序选主元策略的三角分解法构造矩阵L，U和P，再用前向替换法对方程组LY=PB求解Y，最后用回代法对方程组UX=Y求解X。

【输入形式】在屏幕上依次输入方阵阶数n，系数矩阵A和常数矩阵B。

【输出形式】先输出LU分解结果，再输出方程解。

【样例1输入】

4

1 2 4 1

2 8 6 4

3 10 8 8

4 12 10 6

21

52

79

82

【样例1输出】

[[ 4. 12. 10. 6. ]

[ 0.5 2. 1. 1. ]

[ 0.25 -0.5 2. 0. ]

[ 0.75 0.5 0. 3. ]]

[[1.]

[2.]

[3.]

[4.]]

【样例1说明】输入：第1行为方阵阶数4，第2行至5行为系数矩阵A，第6行至9行为常数矩阵B。输出：第1至第4行输出LU分解结果，第5行至第8行依次输出方程解：x1, x2, x3, x4。

【评分标准】根据输入得到的输出准确

## 07. 高斯赛德尔迭代法（线性方程组求解）

【问题描述】为求解一个线性方程组，使用高斯赛德尔迭代法，采用欧几里得距离判定是否收敛。精度delta为1E-9，最大迭代次数为20。

【输入形式】在屏幕上依次输入方阵阶数n，系数矩阵A，常数矩阵B和起始点P。

【输出形式】输出实际迭代次数，然后每一行输出一个根。

【样例1输入】

3

4 -1 1

4 -8 1

-2 1 5

7

-21

15

1

2

2

【样例1输出】

10

[[2.]

[4.]

[3.]]

【样例1说明】输入：第1行为方阵阶数3，第2行至4行为系数矩阵A，第5行至7行为常数矩阵B，第8行至10行为起始点。输出：实际迭代次数为10，然后每行依次输出方程解：x1, x2, x3。

【评分标准】根据输入得到的输出准确

## 08. 雅可比迭代法（线性方程组求解）

【问题描述】为求解一个线性方程组，使用雅可比迭代法，采用欧几里得距离判定是否收敛。精度delta为1E-9，最大迭代次数为20。

【输入形式】在屏幕上依次输入方阵阶数n，系数矩阵A，常数矩阵B和起始点P。

【输出形式】输出实际迭代次数，然后每一行输出一个根。

【样例1输入】

3

4 -1 1

4 -8 1

-2 1 5

7

-21

15

1

2

2

【样例1输出】

18

[[2.]

[4.]

[3.]]

【样例1说明】输入：第1行为方阵阶数3，第2行至4行为系数矩阵A，第5行至7行为常数矩阵B，第8行至10行为起始点。输出：实际迭代次数为18，然后每行依次输出方程解：x1, x2, x3。

【评分标准】根据输入得到的输出准确

## 09. 拉格朗日插值多项式（插值法）

【问题描述】考虑[0.0,1.2]内的函数y=f(x)=cos(x)。利用多个（2,3,4等）节点构造拉格朗日插值多项式。

【输入形式】在屏幕上依次输入在区间[0.0,1.2]内的一个值x\*，构造插值多项式后求其P(x\*)值，和多个节点的x坐标。

【输出形式】输出插值多项式系数矩阵，拉格朗日系数多项式矩阵和P(x\*)值（保留小数点后6位有效数字）。

【样例1输入】

0.3

0 1.2

【样例1输出】

[-0.53136854 1. ]

[[-0.83333333 1. ]

[ 0.83333333 0. ]]

P1(0.3)=0.840589

【样例1说明】输入：x\*为0.3，2个节点的x坐标分别为x0=0和x1=1.2。输出：插值多项式系数矩阵，表示P1(x)为-0.53136854x+1；拉格朗日系数多项式矩阵，表示：y0\*(-0.83333333x+1)+y1\*(0.83333333x+0)；当x\*为0.3时，P1(0.3)值为0.840589。

【评分标准】根据输入得到的输出准确

## 10. 牛顿插值多项式（插值法）

【问题描述】考虑[0,4]内的函数y=f(x)=cos(x)。利用多个（2,3,4等）节点构造牛顿插值多项式。

【输入形式】在屏幕上依次输入在区间[0,4]内的一个值x\*，构造插值多项式后求其P(x\*)值，和多个节点的x坐标。

【输出形式】输出牛顿插值多项式系数向量，差商矩阵和P(x\*)值（保留小数点后6位有效数字）。

【样例1输入】

0.3

0 1 2 3 4

【样例1输出】

4 3 2

-0.01466 x + 0.2345 x - 0.8493 x + 0.1697 x + 1

[[ 1. 0. 0. 0. 0. ]

[ 0.54030231 -0.45969769 0. 0. 0. ]

[-0.41614684 -0.95644914 -0.24837572 0. 0. ]

[-0.9899925 -0.57384566 0.19130174 0.14655916 0. ]

[-0.65364362 0.33634888 0.45509727 0.08793184 -0.01465683]]

P4(0.3)=0.980699

【样例1说明】输入：x\*为0.3，5个节点为(k, cos(k)),其中k = 0, 1, 2, 3, 4。输出：牛顿插值多项式系数向量，表示P4(x)-0.01466 x^4 + 0.2345 x^3 - 0.8493 x^2 + 0.1697 x + 1； 差商矩阵；当x\*为0.3时，P4(0.3)值为0.980699

【评分标准】根据输入得到的输出准确

## 11. 最小二乘拟合直线（曲线拟合）

【问题描述】根据N个数据点构造最小二乘拟合直线y=ax+b。

【输入形式】在屏幕上依次输入数据点的个数N，和N对数据点的x和y坐标。

【输出形式】输出最小二乘拟合直线y=ax+b和误差。

【样例1输入】

8

-1 10

0 9

1 7

2 5

3 4

4 3

5 0

6 -1

【样例1输出】

y=-1.6071429x+8.6428571

1.1801937

【样例1说明】输入：有8对数据点，后续每行是一对数据点的x和y坐标。输出：最小二乘拟合直线为y=-1.6071429x+8.6428571，误差（norm2范数，即欧式距离）为1.1801937（保留小数点后7位有效数字）

【评分标准】根据输入得到的输出准确

## 12. 最小二乘多项式拟合（曲线拟合）

【问题描述】根据N个数据点构造最小二乘多项式拟合。

【输入形式】在屏幕上依次输入数据点的个数N，和N对数据点的x和y坐标。

【输出形式】输出最小二乘多项式和误差。

【样例1输入】

4

-3 3

0 1

2 1

4 3

【样例1输出】

[0.17846248 -0.19249542 0.85051861]

0.2445252

【样例1说明】输入：有4对数据点，后续每行是一对数据点的x和y坐标。输出：最小二乘多项式为y=0.17846248x\*\*2-0.19249542x+0.85051861，误差（norm2范数，即欧式距离）为0.2445252（保留小数点后7位有效数字）

【评分标准】根据输入得到的输出准确