

代码

周潇翔

摘要. 这里总结自己学过的代码供查阅。为啥不用英文？英文的参考文献浩如烟海，也不差我一个啊。对数学系的同学而言，代码的逻辑并不难，大家不会的只是格式而已。第一节横向介绍我们需要啥，之后纵向对每一种语言给出对应的代码。

在这份文档的编译中第一次学 python 和 Github, 可以说是紧跟潮流
23333

1. 代码需求

大部分的语言都需要：

- 安装 + 初始代码 (Halloworld)
- 基本逻辑
- 调试
- 参考文档
- (想保留的) 例子

以下是具体需求：

1.1. 安装 + 初始代码.

- 简要说明该语言的目的
- 说明自己使用何种编译器
- 解释该语言的结构 (基本框架)
- 使用该语言在屏幕中打出”Halloworld”
- 必要时给出英文注释

1.2. 基本逻辑.

- 数据结构类型 (数字、字符串、其他结构)
- 基本四则运算 + mod (若数据结构中包含矩阵，则需要矩阵的各类运算；)
- 条件语句
- 循环语句

- 函数

1.3. 调试.

- 快捷键
 - 运行代码
 - 注释方式及快捷键 (单行注释 + 多行注释)
 - 自动补全功能
 - 自动对齐功能
 - 其他快捷键
- 如何获得帮助
- 控制输出
- 如何设置断点
- 控制输入

1.4. 参考文档.

- 官方文档
- 民间优秀文档

超出科大 C 语言的知识: 编程范式 (Programming paradigm)、方法 (method)

对于图形化输出的语言, 我们还想总结如下知识:

- 坐标轴: 确定位置
- 基本 2D 图形: 圆、线段、圆弧、钢笔工具
- 几何计算: 距离的计算, 交点、中垂线、平行线等
- 3D 作图
- 背景, 控制边界长度
- 页面布局: 盒子模型
- 数学文本输入: 字体

2. PYTHON

2.1. 安装 + 初始代码. Python 是一门高级的编程语言。他有许多的标准模块 (standard module).

2.2. 基本逻辑. 与 C 语言不同, Python 不需要声明变量。当有赋值时不输出结果。

数据类型在这里看到。¹

¹你需要知道那些类型是可改变的 (mutable);

对于数字, Python 不仅有 int 和 float, 还有 Decimal, Fraction and complex numbers 这些奇葩的变量。Task: 学会 Decimal, Fraction.

```
1 >>> complex('1+2j')*complex('1+3j')
```

四则计算像自然计算一样自然, 不过带余除法用//, 余数用%, 幂次用**.(好符号)²

```
>>> a,b=8,13 # a++ is not allowed in python
>>> a ** (b-1) % b # verify the Fermat's little theorem
```

计算器上的 Ans 记为_, round(0.142857,1)给出 0.1

python 中的逻辑运算符如下: 与 (and &), 或 (or |)

条件语句和循环语句的书写规范详见这里。以下是计算素数的例子。

```
import math # Compute square root
def isPrime(n): # return true when n is a prime
    for x in range(2, math.isqrt(n)+1):
        if n % x == 0:
5         return False;
    else:
        # loop fell through without finding a factor
        return True;
isPrime(57)
```

用库可以更加容易地计算素数:

```
1 from sympy.ntheory import isprime
  isprime(10000019)

# find primes by sieve method
from sympy import sieve
3 sieve._reset() # this line for doctest only
# 10000019 in sieve #10000019 is a prime
sieve.extend(100)
sieve._list
```

2.3. 调试. 控制输入使用函数input()。

2.4. 参考文档. 官方文档 SymPy

2.5. (想保留的) 例子. 这个例子计算 gl_n 幂零轨道的维数.

```
# this computes dimension of orbits of nilpotent elements
young_diagram = [2,1,1,1,1] # Here is the input
n=len(young_diagram)
a,b=0,0
```

²请小心使用负数的带余除法。

```

5  for i in range(n):
    a=a+young_diagram[i]
    for j in range(n):
        if young_diagram[i]<young_diagram[j]:
            b=b+young_diagram[i]
10         else:
            b=b+young_diagram[j]
    c=a**2 -b
    print("the dimension of this orbit is", str(c)+".") # one small trick for
    printing result

```

这个例子想要验证某个猜想。据说是 BSD 猜想的推论。

```

from sympy.ntheory import isprime
2  import math      # Compute square root
def isQuart(k):
    for i in range(k//2):
        if i**4 % k == 3:
            return True
7  return False
def solution(m):
    n=math.isqrt(m)+1
    num=0
    for i in range(n):
12        for j in range(n):
            for k in range(n):
                if 6* i**2+j**2+18* k**2 == m:
                    #print("find the solution", i, j, k)
                    if i*j*k==0:
17                        if k % 2 ==0:
                            num=num+4
                        else:
                            num=num-4
                    else:
22                        if k % 2 ==0:
                            num=num+8
                        else:
                            num=num-8
    return num
27 for l in range(25, 10000, 24):
    if isprime(l) & (isQuart(l)==False):
        #print(l,"find the number")
        if solution(l) % 16 ==8:
            print("the conjecture is true for ", l)
32 else:
    print("the conjecture is not true for ", l)

```

这个例子计算 affine quiver 正根的可能情形 (real positive root + regular simple).

```

# Usage of format
2 # E6 case
orderedCouple1 = [(a1,a2,a3,a4,a5,a6,a7) for a1 in range(2) for a2 in
    range(3) for a3 in range(4) for a4 in range(3) for a5 in range(2) for
    a6 in range(3) for a7 in range(2)]
k=0 # compute the number of positive real root which is possible regular
    simple
for a1,a2,a3,a4,a5,a6,a7 in orderedCouple1: # do everything in one line*
    if a1**2+a2**2+a3**2+a4**2+a5**2+a6**2+a7**2-a1*a2-a2*a3-a3*a4-a4*a5-
        a3*a6-a6*a7 ==1:
7         k=k+1
        print('                                {0}'.format(a7)) #ugly code
        have better output
        print('                                {0}'.format(a6))
        print('The positive real root is {0}, {1}, {2}, {3}, {4}.'.format(
            a1,a2,a3,a4,a5))
        print("")
12 print('There are {0} results'.format(k))

# E7 case
orderedCouple2 = [(a1,a2,a3,a4,a5,a6,a7,a8) for a1 in range(2) for a2 in
    range(3) for a3 in range(4) for a4 in range(5) for a5 in range(4) for
    a6 in range(3) for a7 in range(2) for a8 in range(3)]
k=0 # compute the number of positive real root which is possible regular
    simple
17 for a1,a2,a3,a4,a5,a6,a7,a8 in orderedCouple2: # do everything in one line
    *
    if a1**2+a2**2+a3**2+a4**2+a5**2+a6**2+a7**2+a8**2-a1*a2-a2*a3-a3*a4-
        a4*a5-a5*a6-a6*a7-a4*a8 ==1:
        k=k+1
        print('                                {0}'.format(a8)) #ugly
        code have better output
        print('The positive real root is {0}, {1}, {2}, {3}, {4}, {5},
            {6}.'.format(a1,a2,a3,a4,a5,a6,a7))
22         print("")
        print('There are {0} results'.format(k))

# E8 case

```

```

orderedCouple3 = [(a1,a2,a3,a4,a5,a6,a7,a8,a9) for a1 in range(2) for a2
    in range(3) for a3 in range(4) for a4 in range(5) for a5 in range(6)
    for a6 in range(7) for a7 in range(5) for a8 in range(3) for a9 in
    range(4)]
27 k=0 # compute the number of positive real root which is possible regular
    simple
    for a1,a2,a3,a4,a5,a6,a7,a8,a9 in orderedCouple3: # do everything in one
        line*
        if a1**2+a2**2+a3**2+a4**2+a5**2+a6**2+a7**2+a8**2+a9**2-a1*a2-a2*a3-
            a3*a4-a4*a5-a5*a6-a6*a7-a7*a8-a8*a9 ==1:
            k=k+1
            print('                                {0}'.format(a9)) #
            ugly code have better output
32     print('The positive real root is {0}, {1}, {2}, {3}, {4}, {5},
        {6}, {7}'.format(a1,a2,a3,a4,a5,a6,a7,a8))
        print("")
    print('There are {0} results'.format(k))

    # E6 case, subspace case
37 orderedCouple1 = [(a1,a2,a3,a4,a5,a6,a7) for a1 in range(2) for a2 in
    range(3) for a3 in range(4) for a4 in range(3) for a5 in range(2) for
    a6 in range(3) for a7 in range(2)]
    k=0 # compute the number of positive real root which is possible regular
        simple
        for a1,a2,a3,a4,a5,a6,a7 in orderedCouple1: # do everything in one line*
            if a1**2+a2**2+a3**2+a4**2+a5**2+a6**2+a7**2-a1*a2-a2*a3-a3*a4-a4*a5-
                a3*a6-a6*a7 == 1 and a1+a2-3*a3+a4+a5+a6+a7==0:
                k=k+1
42         print('                                {0}'.format(a7)) #ugly code
            have better output
            print('                                {0}'.format(a6))
            print('The positive real root is {0}, {1}, {2}, {3}, {4}'.format(
                a1,a2,a3,a4,a5))
            print("")
        print('There are {0} results'.format(k))

```

3. SAGE

3.1. 安装 + 初始代码. Sage 是专为数学家设计的程序, 基于 python, 通过集成大量数据库, 使用其中的数学函数来简化编程难度。我使用的是网上的编辑器CoCalc, 在 New 中生成 Sage worksheet. 你可以在每一行前添加sage: 或者不加。在最开始输入需要的宏包, 然后直接进行计算即可。

```
sage: print("Halloworld!233") #The code is the same as
python
```

3.2. 基本逻辑. 本节的基本内容参见[这里](#)。

3.3. 参考文档. 官方文档

Sage V9.1 中文文档

3.4. (想保留的) 例子. 这个例子能帮助我计算 Dynkin quiver 的所有不可约表示.

```
sage: Q = DiGraph({1:{2:['a1']},2:{3:['a2']},4:{3:['a3']},5:{4:['a4']},6:{3:['a5']}}).path_semigroup()
sage: M = Q.I(GF(11),3)
sage: M
4 sage: tauM = M.AR_translate()
sage: tauM
```

这个例子能帮助我画出 $\Gamma(5)$ 对应的基本区域.

```
G5 = Gamma(5)
A = FareySymbol(G5).fundamental_domain(show_pairing =
true)
show(A,figsize=10,fontsize=10)
```

4. LATEX

4.1. 本文参考文献. listings 的具体设置

fancy 版本的 Mathematica 代码没学会。

L^AT_EX 中如何自动补全代码? 不容易。

本文 python 代码格式来源

4.2. 参考文档. 刘海洋的书: L^AT_EX 入门

5. MATHEMATICA

5.1. 安装 + 初始代码. Mathematica 是收费的数学计算软件, 参考文档量多但是不够有结构性, 导致我的代码往往是临时性的, 每次计算都需要重新学习代码。(而且我还没有结构性地保存它们) 不过现在可以在这份文档中储存代码了。

5.2. (想保留的) 例子. 这个例子能帮助我计算 affine quiver 相关的矩阵.

```

2  Cp = ({
    {1, 1, 1, 0, 0, 0},
    {0, 1, 1, 0, 0, 0},
    {0, 0, 1, 0, 0, 0},
    {0, 0, 1, 1, 0, 0},
    {0, 0, 1, 1, 1, 0},
7   {0, 0, 1, 0, 0, 1}
    });
    Ci = Transpose[Cp];
    Phi = -Ci.Inverse[Cp]
    MatrixPower[Phi, 6]
12 A = Transpose[Inverse[Ci]] + Inverse[Ci] (*symmetric form*)

```

6. 电脑快捷键 (中级)

部分参见Chrome 快捷键。

- ctrl+win+left: 切换桌面
- alt+(shift)+tab: 切换任务栏上的程序
- ctrl+(shift)+tab: 切换标签页

REFERENCES

SCHOOL OF MATHEMATICAL SCIENCES, UNIVERSITY OF SCIENCE AND TECHNOLOGY
OF CHINA, HEFEI, 230026, P.R. CHINA,
Email address: email