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

对于图形化输出的语言，我们还有很多额外的需求：

- 图形
  - 基本几何图形 (尽可能多样化)
  - Bézier 曲线 (钢笔工具)
  - 参数化曲线
  - 直线，箭头等基本图形库，电路，棋盘，乐器等专业库
  - 几何变换 (平移，旋转，伸缩)
  - 集合运算 (如交并补)
  - 几何运算 (如平行，垂直，切线，交点)
  - 度量衡转换 (数值，距离单位，向量运算，弧度制与角度制)
  - 测量
- 图形进阶
  - 光滑化
  - 锐化
- 坐标系
  - 直角坐标系
  - 其他类型坐标系
- 其他细节
  - 颜色设置
  - 背景与边界设置
  - 透明背景设置
  - 图片嵌入设置
  - 材质设置
  - 数学字体嵌入设置
  - 动画 + 声音嵌入设置 (可选)

对于主要用于排版的语言，我们还希望能了解页面布局的相关知识，如盒子模型。

1.3. 调试.

- 快捷键 (Keyboard shortcut)
  - 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)

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]

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.

```
>>> 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:
            return False;
```

[1]你需要知道那些类型是可改变的 (mutable);

[2]请小心使用负数的带余除法。

```
    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：

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

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

2.3. **调试.** 单行注释使用井号#, 多行注释使用'''注释'''or """注释"""。快捷键为 Ctrl+/.

控制输入使用函数input()。

2.4. **参考文档.** 官方文档 SymPy PKU CS 自学指南

2.5. **(想保留的) 例子.** 这个例子计算正整数的各位数之和，用到了把数转化为字符串的技巧：

```
print(sum([int(d) for d in str(int(input("number:")))]))
```

这个例子计算 $\mathfrak{gl}_n$ 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
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]
        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
```

```
import math    # Compute square root
def isQuart(k):
    for i in range(k//2):
        if i**4 % k == 3:
            return True
    return False
def solution(m):
    n=math.isqrt(m)+1
    num=0
    for i in range(n):
        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:
                        if k % 2 ==0:
                            num=num+4
                        else:
                            num=num-4
                    else:
                        if k % 2 ==0:
                            num=num+8
                        else:
                            num=num-8
    return num
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)
        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
# 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:
            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("")
    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
    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))
            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)]
    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-a6*a9 ==1:
            k=k+1
            print('                                        {0}'.format(a9)) #
        ugly code have better output
            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
    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
            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。我使用的是网上在线的编辑器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. **调试.** 大部分同 python.

3.4. **参考文档.** 官方文档

Sage V9.1 中文文档

3.5. **(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
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。

LaTeX 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：LaTeX 入门

4.3. 非初等但是值得学习的宏包.

- Tikz: 数学系画图专属宏包，其子宏包 Tikzcd 也是相当实用[3]，为了简化制图流程大佬们甚至做了两个网站tikzcd-editor和quiver: a modern commutative diagram editor, 都非常实用。
- tcolorbox: 把文档变得花里胡哨的宏包 (你甚至可以做海报)

4.4. (想保留的) 例子. 这个例子可以画文本宽度的水平直线，并把公式拉长到文本宽度：

```
\hrule % show text width
\resizebox{\textwidth}{!}{ %show the formula very big
        $\displaystyle \frac{1}{\pi}=\frac{2\sqrt{2}}{99^2}
            \sum_{k=0}^{\infty} \frac{(4k)!}{k!^4}\frac
           {26390k+1103}{396^{4k}} $
}
\begin{gather*} % or small
3.14159265358979323846264338327950288419716939937510
58209749445923078164062862089986280348253421170679
82148086513282306647093844609550582231725359408128 4
81117450284102701938521105559644622948954930381 96
\end{gather*}
```

[3] tikzcd is in fact only an improvement of matrix in TikZ. Cited from here.

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。(而且我也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.

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

这个例子计算四元数 $\mathbb{H}$ 上的 non-reduced norm.

```
Det[({
   {x, -y, -z, -w},
   {y, x, -w, z},
   {z, w, x, -y},
   {w, -z, y, x}
  })]
Factor[w^4 + 2 w^2 x^2 + x^4 + 2 w^2 y^2 + 2 x^2 y^2 + y^4
   +
  2 w^2 z^2 + 2 x^2 z^2 + 2 y^2 z^2 + z^4]
```

这个例子画彩色的点.(255 位 RGB)

```
ListPlot[{Style[{{1, 1}, {2, 3}, {2, 4}},
   Interpreter["Color"]["RGB 255 0 0"]],
  Style[{{1.5, 1}, {2.3, 3}, {2.4, 4}},
   Interpreter["Color"]["RGB 0 255 0"]]}]
```

6. POV-Ray

6.1. **安装 + 初始代码.** POV-Ray 是一款依赖代码的 3D 建模软件，免费开源，目前个人感觉是"3D 版本的 Tikz"." 一款能够在短时间内持续追踪光线视觉，产生高质量计算机图像的工具。" from here.

程序对大小写敏感。

一些特殊符号：

```
o=<0,0,0>, x=<1,0,0>, y=<0,1,0>, z=<0,0,1>, u=<1,0>, v
    =<0,1>.
sqrt(a), pi, pow(a,n) sin(radians(90)), degrees(atan2(1,1))
```

6.2. **基本逻辑.** 基本几何体请参考这里。

透明背景设置

superellipsoid 可以画圆角圆柱和方体。

6.3. **调试.** 单行注释使用两斜杠//, 多行注释使用/*注释*/。对于.ini 文档，单行注释使用分号;.

关于输出请参考此文档。

6.4. **参考文档.** 官方文档、视频教程

动画效果

POV-Ray 的经典实例是维度：数学漫步。

关于 3D 建模还有很多优秀的软件，比如:

- solidwork: 曾经尝试过，后来删了
- 3DMax: 实例可以参考GM 的建模基地。但是收费。
- blender: 免费开源，似乎是 3DMax 的高级替代品，依赖快捷键。

如果我有一台优质的电脑、时间 (没书读时) 和精力那我也会尝试下。现在就算了。

6.5. **(想保留的) 例子.** 这个例子实现了杯子的旋转动画，并制作了一个好的坐标轴。

```
// POV-Ray Scene File "begin.pov"
// by Friedrich A. Lohmueller, Jan-2013, now Xiaoxiang Zhou
//------------------------------------------
global_settings{assumed_gamma 1.0}
//#default{ finish{ ambient 0.1 diffuse 0.9 }}
//------------------------------------------

#include "colors.inc"
#include "textures.inc"
#include "woods.inc"
//------------------------------------------
// camera -----------------------------------
 camera{
  location <25,25,25>    //best: <25,25,25> <10,-25,25>
```

```
  right -x*image_width/image_height
  sky   <0,0,1>    //see the sky
  look_at <0,0,0>
  angle 22
 }
 // sun ----------------------------------
 light_source{
    <25,25,25>,
    White
    shadowless
 }
 // sky ----------------------------------
 sky_sphere{
  pigment{
    gradient z-3
    color_map {
     [0 color White]
     [0.95 color Blue]
     [1 color White]
    }

    scale 2
    translate <0,0,1>
  }
 }
   // axis arrows ----------------------------------
 cone {
    6*x, .1, 6.5*x, 0
 }
cone {
    6*y, .1, 6.5*y, 0
  }
cone {
    6*z, .1, 6.5*z, 0
   }

  // axis ----------------------------------
cylinder {
    -6*x, 6*x, .021
    pigment{ Black}
    finish {ambient .3}
}
cylinder {
    -6*y, 6*y, .021
    pigment{ Black}
```

```
    finish {ambient .3}
}
cylinder {
    -6*z, 6*z, .021
    pigment{ Black}
    finish {ambient .3}
}
  // grid -------------------------------------
#for (i,-5,5,1)
cylinder {
    <i,-5,0>, <i,5,0>, .02
    pigment{ Grey}
    finish {ambient .3}
}
cylinder {
    <-5,i,0>, <5,i,0>, .02
    pigment{ Grey}
    finish {ambient .3}
    }
#end


 text {
    ttf "timrom.ttf" "x" 0.05, 0
    pigment { Black }
    translate -0.25*x
    rotate<90,0,180>
    translate 6*x-0.8*z
  }
 text {
    ttf "timrom.ttf" "y" 0.05, 0
    pigment { Black }
    translate -0.25*x
    rotate<90,0,90>
    translate 6*y-0.8*z
  }
 text {
    ttf "timrom.ttf" "z" 0.05, 0
    pigment { Black }
    translate -0.25*x
    rotate<90,0,135>
    translate 6*z-0.8/sqrt(2)*x+0.8/sqrt(2)*y
 }

 difference{
```

```
union {
sphere_sweep {
    b_spline
    7,
    <.7,2.2,0>, .8
    <1.5,.5,0>,.3
    <3.1,.5,0>,.3
    <3.8,2.4,0>,.3
    <3.1,4.1,0>,.3
    <1.5,4.1,0>,.3
    <.7,2.6,0>,.8
        texture{
   pigment{ Grey}
   finish { phong 1}
 }
    finish {ambient .3}
    scale <1,1,2.5>
}  //cup
cylinder{
    <0,0,0>, <0,4.75,0>,2

    scale 1/4.75

        texture{
   pigment{ Grey}
   finish { phong 1}
 }
    scale 4.75
    finish {ambient .4}
     rotate<0,-60,0>
    }
} //end of union

cylinder {
    <0,.2,0>, <0,4.9,0>, 1.8
        texture{
   pigment{ Grey}
   finish { phong 1}
 }
    finish {ambient .3}

}
rotate<90,0,360*clock>
} //end of difference
```

我们还需要.ini 文档才能够生成动画。

```
; Persistence Of Vision raytracer version 3.7 example file.
Antialias=On

Antialias_Threshold=0.1
Antialias_Depth=2
Input_File_Name=begin.pov

Initial_Frame=1
Final_Frame=30
Initial_Clock=0
Final_Clock=1

Cyclic_Animation=on
Pause_when_Done=off
```

6.6. **Tasks.** 制造一个基本对象集，以及基本操作。

7. 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
- win+D: 显示桌面
- win+left: 页面占左半边
- win+R: 调出运行窗口

8. 正则表达式

正则表达式可以大幅提升搜索和修改代码的效率。

```
.{0,100}(?=;\d{0,3};\d{0,3}m\n);(\d{0,3});(\d{0,3})m\n
```

8.1. **参考文档.** 推荐B 站视频。与之配套的练习。

你可以在regexr中测试。

9. 热键设置 (AutoHotkey)

这是一款开源软件。为了 Hackergame 的脚本，同时我猜会大幅度提高打代码的速度。

9.1. **调试.** 注释使用分号;。

9.2. **参考文档.** 官方文档. ;

9.3. **(想保留的) 例子.** 这个例子模拟鼠标和键盘，使用了条件和循环语句。

```
^j::
Sleep,300
loop, 1
{
    Random, rand, 1, 999
        CoordMode, ToolTip
        Click, 1743 81 1
        Sleep, 100
        Click, 36 64 2
        Sleep, 100
        Send, {rand}.0.0.0
        Sleep, 100
        Click, 433 323 2
        Sleep, 100
        Send, 0.0.0.0
        Sleep, 100
        Click, 452 321 1
        Sleep, 100
        if GetKeyState("F10")
        {
        break
        }
        Sleep,1000
}
Return
```

这是第二版本，能够 95% 成功提交。

```
#MaxThreadsPerHotkey 3
^j::  ; 热键(可根据您的喜好改变此热键).
#MaxThreadsPerHotkey 1
if KeepWinZRunning  ;  这说明一个潜在的线程正在下面的循环中
    运行.
{
    KeepWinZRunning := false  ; 向那个线程的循环发出停止的
        信号.
    return  ; 结束此线程, 这样才可以让下面的线程恢复并得知
        上一行所做的更改.
}
; 否则:
KeepWinZRunning := true
rand := 0
Loop
```

```
{
    ; 以下四行是您要重复的动作(可根据您的需要修改它们):
    MyNumber :=  rand . ".233.233.233"
        Click, 1743 81 1
        Sleep, 70
        Click, 74 74 2
        Sleep, 70
        Send, %MyNumber%
        Sleep, 70
        Click, 429 346 2
        Sleep, 80
        Send, %MyNumber%
        Sleep, 90
        Click, 1073 31 1
        Send ^{Click 432 360 1}
        Sleep, 190
        rand += 1
    ; 但请不要修改下面剩下的内容.
    if not KeepWinZRunning  ; 用户再次按下 Win-Z 来向循环发
        出停止的信号.
        break  ; 跳出此循环.
}
KeepWinZRunning := false  ; 复位, 为下一次使用热键做准备.
return
```

第三个版本可以提交 254 个，还剩 2 个。。。

```
#MaxThreadsPerHotkey 3
^j::
#MaxThreadsPerHotkey 1
if KeepWinZRunning
{
    KeepWinZRunning := false
    return
}
KeepWinZRunning := true
rand := 0
Loop
{
    MyNumber :=  rand . ".233.233.233"
        Click, 1751 91 1
        Sleep, 120
        Click, 74 74 2
        Sleep, 50
        Send, %MyNumber%
        Sleep, 70
```

```
        Click, 429 346 2
        Sleep, 80
        Send, %MyNumber%
        Sleep, 50
        Click, 587 747 1
        ;Click, 1073 31 1
        Send ^{Click 387 437 1}
        Sleep, 60
        rand += 1
    if not KeepWinZRunning
        break
}
KeepWinZRunning := false
return
```

最后成功的版本：

```
#MaxThreadsPerHotkey 3
^j::
#MaxThreadsPerHotkey 1
if KeepWinZRunning
{
    KeepWinZRunning := false
    return
}
KeepWinZRunning := true
rand := 0
Loop
{
    MyNumber :=  rand . ".233.233.233"
        Click, 1751 91 1
        Sleep, 120
        Click, 74 74 2
        Sleep, 50
        Send, %MyNumber%
        Sleep, 70
        Click, 429 346 2
        Sleep, 80
        Send, %MyNumber%
        Sleep, 40
        Click, 587 747 1
        ;Click, 1073 31 1
        Send ^{Click 387 437 1}
        Sleep, 60
        rand += 1
    if not KeepWinZRunning
```

```
        break
}
KeepWinZRunning := false
return
```

9.4. **Tasks.** 通过该软件创建一个 LaTeX 的脚本，要求：

- 自动翻译 iff, SES, LES, mfld, rep...
- 长段代码如 equ+aligned
- 只在打开 TeXstudio 时生效 (使用 IfWinActive+Window Spy)

通过该软件创建一个脚本，要求：

- 自动打开日常软件
- 将其放入开机启动项

已完成，代码如下：

```
#IfWinActive, ahk_class Qt5QWindowIcon
::SES::short exact sequence

::LES::long exact sequence

::TFAE::the following are equivalent:

::iff::if and only if

::st::such that

::eg::for example

::geo::geometry

::cont::continuous

::diff::differential

::sm::smooth

::mfld::manifold

::lb::line bundle

::mlb::metrized line bundle

::vb::vector bundle
```

```
::cplx::complex

::RS::Riemann surface

::AV::abelian variety

::ell::elliptic curve

::proj::projective ;be careful for projection

::inj::injective

::fct::function

::iso::isomorphism

::irr::irreducible

::ind::indecomposable

::adm::admissible

::rep::representation

::Archi::Archimedean field

::sign::Best wishes,`n`nXiaoxiang Zhou

::sym::symmetric

::corrg::corresponding ;be careful for correspondence

::corre::correspondence

::align:: ;超长变换,我的Texstudio还是有些问题
(
\begin{equation*}
\begin{aligned}
  =\;&  \\
)

::draw:: ;Texstudio如何原样打出?
(
\begin{figure}[ht]
```

```
  \vspace{0cm}
  \centering
  \includegraphics[width=12cm]{}
  \label{fig:}
  \caption{}
\end{figure}
)

::minipage::
(
\begin{figure}[th]
  \begin{minipage}[t]{.48\textwidth}
    \centering
    \includegraphics[width=\textwidth]{}
    \label{fig1}
  \end{minipage}
  \begin{minipage}[t]{.48\textwidth}
    \centering
    \includegraphics[width=1\textwidth]{}
    \label{fig2}
  \end{minipage}
\caption{}
\end{figure}
)

^+c:: ;注释
send ^c
sleep, 200
clipboard=%clipboard%
tooltip, %clipboard%
sleep, 500
tooltip,
return

::||:: ;临时符号
(
\norm{\cdot}
)

::CLm::Chambert-Loir measure

::show::\displaystyle

::comm diag::commutative diagram
```

```
::reso::resolution

::pfv::partial flag variety

::qgr::quiver Grassmannian


#IfWinActive
^j::
Run, "F:\hide\Snipaste\Snipaste.exe"
Run, "F:\hide\WiFi共享大师\WiFiMaster.exe"
return
```

10. 其他

这里收集乱七八糟的材料，以后说不定可以单独搞个文件。

10.1. **数学的可视化.** 这里以四维空间为例。

可以参考Andrew J. Hanson 的主页，也可以参考Github 中的项目(这里面有很多可以了解到的就不引用了)。

脑洞：参考四维截面游戏，我们是否可以做个四维版本的小球进洞？

10.2. **数学与游戏.** 脑洞：搞个 Riemann surface 的世界，在介绍这个世界的时候以神的旨意来介绍相关概念

```
1.安装Unity
2.用sublime配置unity
3.制作横版跳跃游戏，包含：
场景/人物/交互/动作(左右动，跳跃，蹲)/人物动画
4.制作贪吃蛇游戏，熟悉代码(无边框，不蠢，带格子)
5.制作扭结+贪吃蛇游戏
6.制作黎曼面游戏(两点)
方程
游戏类型:找不同？横版跳跃？迷宫？五子棋(4/3子棋,粘附5/4/3子
    棋)？围棋(扭羊头游戏)?跳棋？空间爬行？扭结+贪吃蛇？塔
    防？数独？数回？华容道？(mini metro？连连看？泡泡龙？消
    砖块？2048？推箱子？其他的Nikoli Puzzles？)
简易示意图
可能引入的概念:Laurent展开/同调群、基本群的概念(在数回中，
    请使得最后画出的曲线表示基本群不平凡但是同调群不平凡)/覆
    叠，万有覆叠，deck transformation(华容道简化难度:允许
    deck transformation将区块变换至另一个基本区域中相同的位
    置；或者，我们允许上下粘接)/曲率/曲率驱动游戏/descent(数
    独解何时能成为更小的黎曼面的解)
```

```
视角：收敛区域视角(遮罩圆形)、星形区域视角、基本区域视角、
    万有覆叠视角、第一人称视角、二维3D视角(视角需要通过"培训
    +考试"解锁)

虚拟对话
-宇宙到底是啥？(永远都到达不了的点是否真的存在？-black hole
    宇宙是几维的？2+3(颜色)+1(时间) 如何测量时间，or时间的
    流逝是否是均匀的？)
-自由意志(我可以自由地说我想说的东西，而你的输入却受到限制)
-表达自己想做却做不到的事
-对数学的热爱
```

脑洞：利用扭结来制作游戏，例如“贪吃蛇”。需要的时候考虑离散化，也可以考虑离散版本的 Reidemeister moves。游戏制作的难点在于如何判定成功 (好的 Goal 是啥。很难去说明一个扭结和另一个扭结等价) 扭结的相关知识可参考这个视频。

10.3. **数学相关句子模板.** 这里收集“啊我之前不知道咋表达，这就是我当时卡住的时候可以用的句子!”的英文句子。

```

notations:
Beginning with..., only ... are considered.
shorthand
By an aesthentically desirable abuse of notation,
One word about the notation: for simplicity we often omit
    ...
For notational reasons we usually extend the ... to (
    negative indices) by defining ...

refer to:
The main references for the material covered in this
    section are ...
A more detailed overview of this chapter may be obtained by
     reading the introductions to the various sections.


example and generalization:
prototypical example
By the same argument as the special case above, ...
The arguments in ... carry over mutatis mutandis.

Disclamer:
```

```
I am by far not a person with serious knowledge/
    understanding of ..., thus in the ... I may oversimplify
    /overcomplicate things, be inaccurate, or even wrong,
    and miss subtelties.

equivalently(replace i.e.)
encyclopaedic knowledge of ...
In fact, the arguments used in the final part of this proof
    , give the following result.
Let us unravel this definition a little.(after complicated
    definition)
We point out the similarities and the differences whenever
    appropriate.

brackets:
{}:curly set brackets
():round brackets
[]:square brackets
<>:angle brackets

beginning words: Goals, Pros and Cons
```

另外关于英语语法，可以参考天文物理类英文科技论文写作的常见问题.

References

School of Mathematical Sciences, University of Science and Technology of China, Hefei, 230026, P.R. China,

Email address: xx352229@mail.ustc.edu.cn