Python version :3.10.6??学校的是几阿

Starting with Python 3.1, Python (on most systems) is now able to choose the shortest of these and simply display 0.1.

有要求一定要参考多少本书吗

with open https://blog.csdn.net/weixin\_39613951/article/details/110785735

连续多个open

误差： 考虑数据向量是frame到frame的，要加上visual的高度吗，再与原来的作对比，看看原来的gripper会顶到rot plate吗

根据要求知道要修改的是什么标签了，但是在编写程序之前，我们需要先探究一下改动这些属性，对其他urdf标签有什么影响。以及相邻的link joint

是float64吗 double类型？

Getcontxt.pre 只会影响decimal对象

二进制不能准确的表示数:

print(0.0795000000000249+0.0795000000000246)

print('%.17f'%(10.0795000000000249+10.0795000000000246))

0.1590000000000495

20.15900000000004866

17位有效数字截断，https://help.uis.cam.ac.uk/system/files/documents/python3-notes.pdf

我们的最大的不超过17..

print(0.07750000000002338+0.07750000000002337)

0.15500000000004677

print('%.18f'%(0.07750000000002338+0.07750000000002337))

0.155000000000046767

print(0.07950000000000249+0.07950000000000246)

print('%.19f'%(0.0795000000000249+0.0795000000000242))

print('%.19f'%(10.0795000000000249+10.0795000000000242))

0.15900000000000494

0.1590000000000491021

20.1590000000000486580

print(0.07950000000002435)

0.07950000000002436

print(0.07950000000002435+0.23156418)

0.31106418000002434

由于这些 长的加长的不准，长的加短的也不准，所以我们干脆直接全用decimal

(decimal buneng + float)

localcontext

print(Decimal('0.07950000000002435')+Decimal('0.23156418'))

print(Decimal(0.23156418))

print('%.53f'%(0.23156418))

0.31106418000002435 是python3.11之后才能Deciamal(float)的吗

0.23156418000000000834148750072927214205265045166015625 实际显示的，我们看到的数不是他真实的数 但是我们要的就是看到的数。。

0.23156418000000000834148750072927214205265045166015625 python会三选1 但这不是以前版本的python才会这样吗

程序里print(getcontext()) pre是28,所以rounding参数不需要round\_down。看看别的参数

程序里的Decimal()里得要是str，float的话会出事：修改完的数据会变成28位的，比如1K\_Dispenser。

程序里格式化用%s不是%f或者%.17f，是因为会出事。 str（0.31641532...)里的是float，本身就是不对的，所以str完还是不对，但是这里%s可以，哦，str是将他转化成字符串, 不对，直接%s（。。。）了。