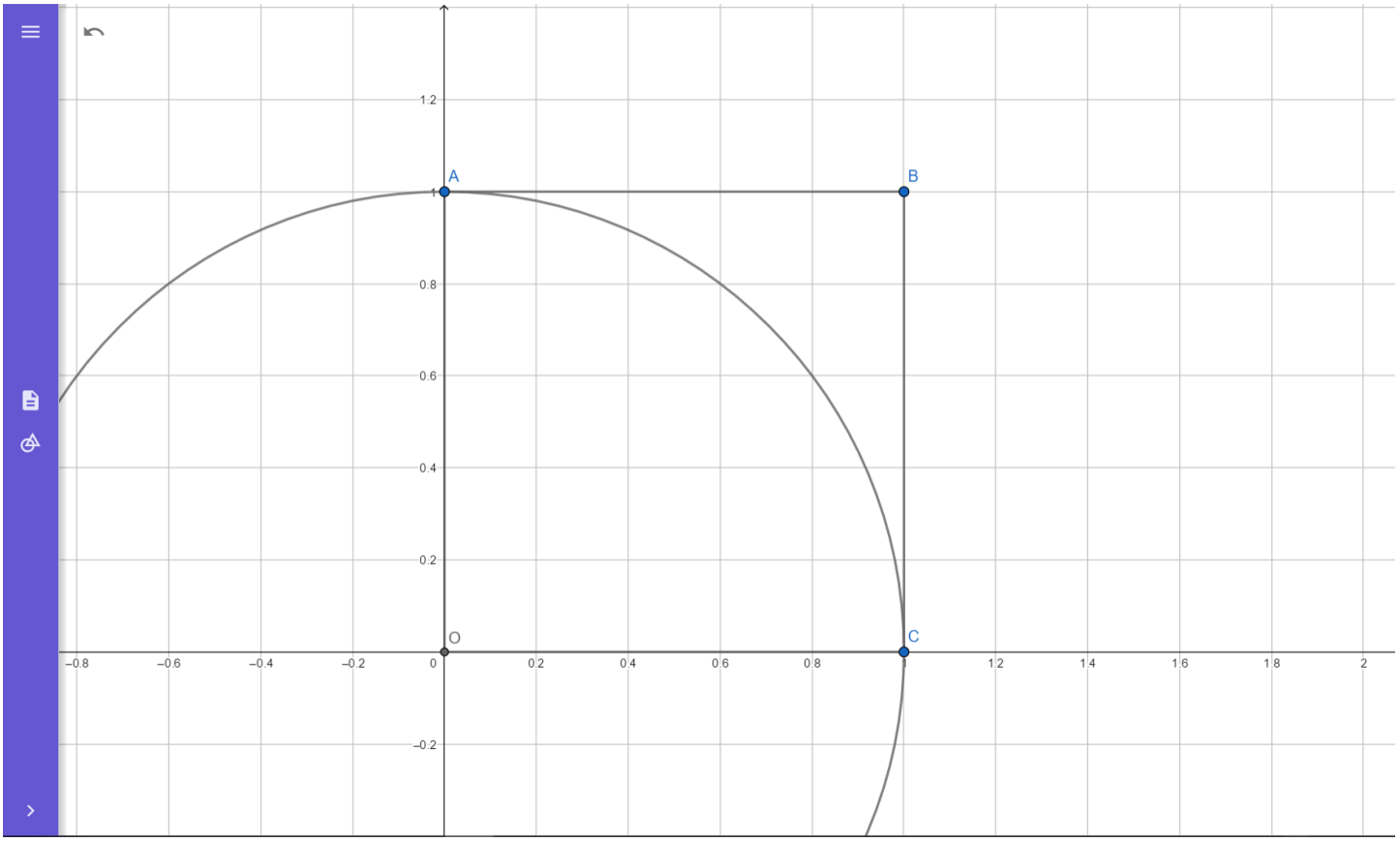
π的计算

在预习初三圆的内容时，突然对π的值是如何求出的产生了疑问，经过一番思考和查阅资料，得出了一种较为简单的计算方法。

根据圆面积计算公式`S=π*r^2`，易得`π=S/r^2`,也就是说只要知道圆的面积和半径，就可以求出π的值。我想到了用频率估算概率的



如图，在正方形OABC中任意取一点，点在扇形AOC上的几率为`S/4=π*r^2/4=π/4`，因此只要在正方形ABCD中随机取m个点，若在扇形AOC内，则`π=n/m*4`

为了结果的准确性和偷懒，我决定借助python来完成这个任务：

```
from random import choice #choice函数，从一个列表中随机选取一个数
from decimal import Decimal #高精度浮点数类
from fractions import Fraction #分数类，避免浮点数导致的精度损失
```

首先，从模块中导入所需的类和函数

```
def rand_point(m,pre):
    count=0
    while count<=m:
        x=Fraction(f'{choice(range(pre))}/{pre}')
        y=Fraction(f'{choice(range(pre))}/{pre}')
        #随机生成两个0-1之间的整数，精度为10^-pre
        res=sum((x**2,y**2)) #计算点(p1,p2)到原点距离的平方
        count+=1
        yield res<=1 #返回点到原点距离的平方是否小等于1，即是否在扇形内
```

定义一个生成器，按需生成点并计算到原点的距离 其中`f'{}'`表示一个字符串,`{}`中填写一个表达式(f-string,python3.6后加入),数用于生成一个数列：0,1,2,...,n-1。整个字符串表示一个分数。

```
n,m=0,2<<25
```

n表示点落在扇形内的频数，m表示实验总次数，`<<`为左移符号(在二进制中补零,`如0b10(2)>>1==0b100(4)`)，等价于*2，`2<<25`等于`2^26`(python写法`2**26`) ~~这样写似乎效率更高~~

```
r=rand_point(m,pre=11**18)#初始化生成器
```

(模拟)生成一个坐标系，单位长度为11^-18

```
for x in r:
    n+=x
```

迭代生成器，x值只能为True(1)(点在扇形内)或False(0)(点在扇形外)
~~这种写法在一些语言(如Java)中被视为非法，且降低可读性，因此不推荐~~

```
print(f'π={Decimal(4*n)/Decimal(m)},times={m}')#输出结果
```

最终执行(两次)的结果为:

```
π=3.141416072845458984375,times=8388608
π=3.141605854034423828125,times=8388608
```

(使用CPython运行,理论上可以使用PyPy,可以大幅提高执行效率)
精度为`0.001`,由于语言限制,很难进一步提高精度。

事实上,求π值还有很多更好更快的算法，但是如果让我用一个目前无法理解的方法，还不如

```
import math
print(math.pi)
```

常熟市外国语初级中学初二(3)班 朱博文