A Slice of Pi

Nota

Este é um problema razoavelmente complicado! Como tal, será importante fazer um plano antes de começar a escrever qualquer código. Você pode querer ler a seção sobre estratégia, que fornece algumas dicas para gerenciar a complexidade desse problema, antes de começar (mas depois de ler a descrição do problema).

---

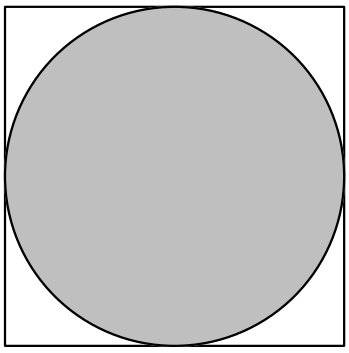
1) Descrição do Problema

Neste problema, consideraremos um método interessante para aproximar o valor de π.

Imagine que você não soubesse o valor de π, mas conhecesse a relação entre a área de um círculo e π, especificamente que:

$$A = \pi r^2$$

Poderíamos então usar a equação de área e o método a seguir para aproximar π computacionalmente. Em primeiro lugar, considere um círculo inscrito em um quadrado:



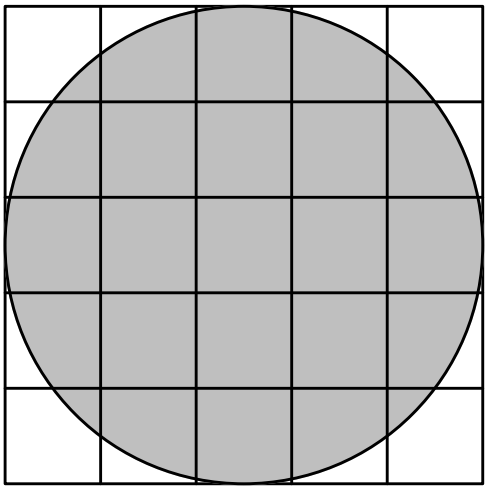
Tente agora:

Qual fração da área do quadrado (A_q) é ocupada pelo círculo (A_c), em termos de π e do raio do círculo r?

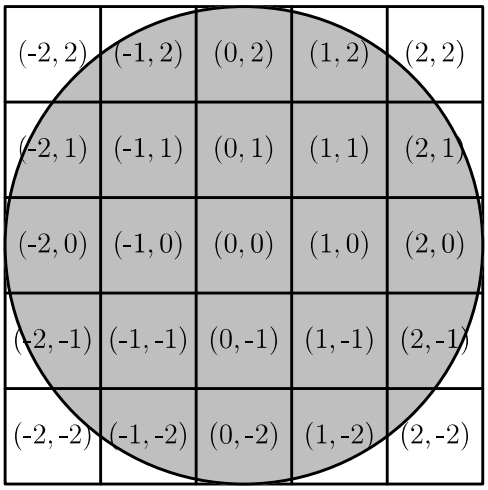
▶ Mostrar/Esconder

Medir essa proporção nos daria um caminho claro para calcular π com base na fórmula acima (podemos apenas reorganizar a fórmula para encontrar π). Mas acontece que a proporção em si é difícil de medir. No entanto, podemos pensar em uma maneira de aproximar isso.

Para aproximar essa proporção, podemos primeiro cortar o quadrado em uma grade com base em um parâmetro de discretização de valor ímpar p_d, que nos diz quantas "células de grade" deve haver ao longo de cada eixo. Abaixo está o mesmo desenho sobreposto com uma grade correspondente a $p_d = 5$:



Nesta visão, se rotularmos o centro da célula do meio como $(0, 0)$ e assumirmos que cada célula tem comprimento unitário, então podemos rotular cada célula com uma localização (x, y):



Quadrados cujo centro está dentro ou exatamente na borda do círculo podem ser contados testando se, para cada célula (x, y), a distância do centro do círculo ao centro dessa célula é menor que o raio:

$$\sqrt{x^2 + y^2} \leq \frac{p_d}{2}$$

Tente agora:

Por que a distância é comparada com $p_d/2$?

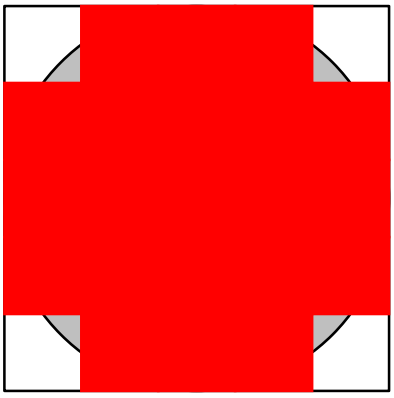
▶ Mostrar/Esconder

O número de células que satisfazem essa condição dividido pelo número total de células na grade se aproxima da proporção A_c/A_q, que, como vimos acima, podemos usar para aproximar π. Conforme p_d muda, a razão (em termos de número de células da grade) se aproxima da razão teórica, e assim nossa estimativa de π se aproxima do valor verdadeiro de π também!

Nos desenhos abaixo, os quadrados que satisfazem esta condição são coloridos em vermelho, para $p_d = 5, p_d = 13, p_d = 101$, e algumas estatísticas são mostradas para cada caso (que você pode usar para ajudar a testar seu código enquanto trabalha!).

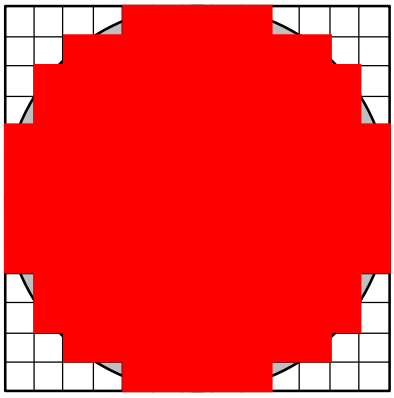
Como você pode ver, conforme p_d aumenta, a região sombreada parece ficar cada vez mais perto da área real do círculo, e nossa estimativa fica cada vez mais perto de π! (demora um pouco para obter uma aproximação de π, talvez próximo a $p_d \to 5000$).

$p_d = 5$



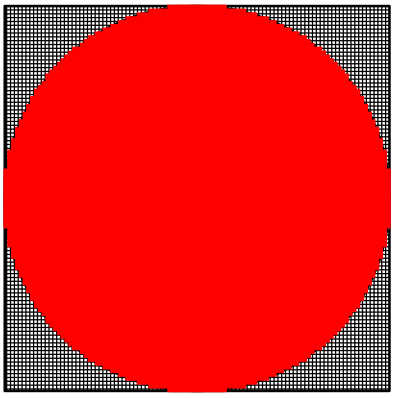
Cells in circle: 21
Total cells in square: 25
Estimate of A_c/A_s: 0.84
Estimate of π: 3.36

$p_d = 13$



Cells in circle: 137
Total cells in square: 169
Estimate of $A_c/A_s \approx 0.81065$
Estimate of $\pi \approx 3.24260$

$p_d = 101$



Cells in circle: 8021
Total cells in square: 10201
Estimate of $A_c/A_s \approx 0.78630$
Estimate of $\pi \approx 3.14518$

Tente agora:

Que valor este método produziria quando $p_d = 1$?

▶ Mostrar/Esconder

---

2) Especificação do Programa

Escreva um programa que estime π usando o método acima para um valor particular de p_d que não seja um dos três valores acima e então atribua esse valor a uma variável `out`. A primeira linha em seu programa deve definir uma variável `p_d` (com o underscore!) para conter o valor de p_d a se usar. Por exemplo:

```
p_d = 10
```

e deve atribuir a `out` a estimativa de π obtida pelo método acima e seu valor de p_d. Não arredonde nenhum valor!

3) Estratégia e Design

Este é um problema bastante complicado e, portanto, é uma ótima oportunidade para falar um pouco sobre o *design* de programas. Frequentemente, podemos pensar em como gerenciar a complexidade de um programa dividindo o programa em etapas para ajudar a tornar o processo um pouco mais gerenciável. Às vezes, é mais fácil no longo prazo começar não escrevendo o problema que realmente queremos resolver, mas sim um problema semelhante e menor que nos ajuda a realizar uma das etapas que estamos interessados em resolver.

Neste problema, há várias coisas que temos que fazer:

1. temos que considerar cada célula da grade
2. temos que calcular as distâncias entre as células da grade e a célula central
3. temos que usar esse resultado para verificar se a célula em questão está dentro do círculo
4. temos que acompanhar quantas células estão no círculo e quantas estão no quadrado

Isso é bastante, mas algumas dessas peças podem ser abordadas por meio de uma abordagem *iterativa* para resolver esse problema. Uma dessas estratégias é descrita abaixo (que você pode seguir, ou não, conforme achar adequado!). Ao longo do caminho, a seção de leituras desta tarefa sobre debugging (http://web.mit.edu/~armelin/python_intro_iap/Notes/IPL%201.html#7)-Debugging) pode ser útil!

- Pode fazer sentido começar com um programa que usa `p_d` para simplesmente imprimir os pares (x, y) de todas as células que queremos verificar. Você pode testar isso com `p_d = 5` para ter certeza de obter todas as mesmas coordenadas do diagrama acima.
    - Mesmo este não é necessariamente um problema fácil! Você pode querer quebrar isso em partes e começar escrevendo um programa que use `p_d` para imprimir apenas os valores x das células que queremos verificar. Depois de ter isso, como você pode expandir essa ideia para imprimir os pares (x, y)?
- Assim que tiver isso funcionando, você pode modificar seu programa para que ele imprima não apenas o par (x, y) associado a esse ponto, mas também a distância do centro a esse ponto (que você pode verificar por mão para cada célula – use simetria para reduzir o trabalho quando estiver fazendo a mão!).
- Em seguida, você pode adicionar a verificação se o centro da célula está dentro do círculo ou não e imprimir esse resultado (`True` ou `False`) em vez de imprimir a distância diretamente. Novamente, você pode verificar esta etapa certificando-se de que todas as células vermelhas no diagrama `p_d = 5` acima imprimem `True` em seu código quando `p_d = 5` .
- Em seguida, você pode modificar seu código para controlar o número de células dentro e fora do círculo e para imprimir esses valores uma vez que seja feito considerando todas as células. Novamente, você pode compará-los com os desenhos acima para vários valores de `p_d` mostrados acima.
- Finalmente, você pode remover todas as instruções `print` úteis que você usou para debugging e certificar-se de que seu programa apenas atribua o valor pedido a `out` .

Submissão

Quando estiver pronto (depois de ter simulado manualmente e testado em sua própria máquina e estiver convencido de que seu programa fará a coisa certa), faça upload do seu arquivo Python no **Problema 1.6**