Desenvolvendo o Jogo do NIM

Joao Lucas da Silva

https://joao-lsilva1198.medium.com/desenvolvendo-o-jogo-do-nim-9126a87f66d5

Oct 9, 2020·4 min read

```
Quantas peças? 5
Limite de peças por jogada? 3

Computador começa!

O computador tirou uma peça
Agora restam, 4 peças no tabuleiro.

Quantas peças você vai tirar? 2

Você tirou 2 peças
Agora restam, 2 peças no tabuleiro.


O computador tirou 2 peças
Fim do jogo! O computador ganhou!
```

No jogo do NIM, *n* peças são dispostas numa mesa ou tabuleiro. Dois jogadores se alternam, retirando no mínimo 1 e no máximo *m* peças cada um. Quem tirar as últimas peças possíveis ganha o jogo.

Este programa é a tarefa de programação pedida na sexta semana do curso Introdução a Ciência da Computação com Python. Neste tutorial mostrarei como desenvolvi este jogo,

para que um usuário posso jogar contra o computador na linguagem Python.

Inicialmente, existe uma estratégia para se ganhar o jogo, sempre deixar um números de peças múltiplos do máximo mais um (m+1) para o oponente. A “inteligência” do computador deve ser fundamentada nessa estratégia.

A primeira função é a em que o computador escolhe a jogada, o computador deve remover no mínimo uma peça, e no máximo o número máximo permitido, porém, para que o computador possa vencer a partida, deve-se deixar ao usuário um número de peças múltiplos do máximo mais um (m + 1); caso isso não seja possível o computador deve retirar o máximo (m) de peças.

Esta função receberá via parâmetro (n)o número de peças, e o número máximo que se possa retirar na partida (m). E deverá retornar o numero de peças que o computador removeu do tabuleiro. O laço de repetição indicará quando o número de peças que o computar irá (*computadorRemove*) remover for maior que o máximo permitido, sendo *n menos computadorRemove* múltiplo de m + 1 o computador irá retornar o número de peças que o computador removerá.

```
def computador_escolhe_jogada(n,m):
    computadorRemove = 1while computadorRemove != m:
        if (n - computadorRemove) % (m+1)== 0:
            return computadorRemove
        else:
            computadorRemove += 1
    return computadorRemove
```

while computadorRemove != m:
if (n — computadorRemove) % (m+1)== 0:
return computadorRemove
else:
computadorRemove += 1
return computadorRemove

A próxima função é a em que o usuário removerá as peças. Nesta, basta pedir que o usuário informe o número de peças que quer remover, e verificar se esse número é válido, caso não seja, requisitar ao usuário que informe o número novamente.

```
def usuario_escolhe_jogada(n,m):
    op_errada = False
    while not op_errada:
        pecas_retirar= int(input("Quantas peças você vai
tirar? "))
        if pecas_retirar > m or pecas_retirar < 1:
            print("\nOops! Jogada inválida! Tente de
novo.\n")
            op_errada = False
        else:
            op_errada = True
    return pecas_retirar
```

Agora a função partida do jogo, no inicio esta função requisita que o usuário informe a quantidade de peças (n) e o número máximo de peças que se pode retirar na partida (m). Deve-se verificar se m não é maior que n, caso for, requisitar que o usuário digite m novamente.

```
def partida():
    pecas_validas = False
    while not pecas_validas:
        n = int(input("Quantas peças? "))
        m = int(input("Limite de peças por jogada? "))
        if m > n:
            print("\nOops! Número de peças a retirar
```

```
inválido! Tente de novo.\n")
        else:
            pecas_validas = True
```

Agora, deve-se definir quem iniciará cada partida. Se n for múltiplo de m + 1 o usuário deve começar, caso contrário, o computador irá começar. Para isso será utilizado a variável *vezDoPC* que receberá um valor booleano.

```
vezDoPC = False
    if n % (m+1) == 0:
        print()
        print("Voce começa!\n")
    else:
        print("Computador começa!\n")
        vezDoPC = True
```

A partida ocorrerá enquanto houver um número de peças no tabuleiro (*n*) maior que 0.

Se for *vezDoPC* a função em que o computador escolhe a jogada será chamada. Aí entra a verificação para mostrar quantas peças o computador removeu. Se após o computador ter removido as peças, *n* for 0, então o computador venceu a partida.

```
if vezDoPC:
            computadorRemove = computador_escolhe_jogada(n,
m)
            n = n - computadorRemove
            if computadorRemove == 1:
                print('O computador tirou uma peça\n')
            else:
                print('O computador tirou', computadorRemove,
'peças\n')
            if n == 0:
                print('Fim do jogo! O computador ganhou!')
            vezDoPC = False
```

Caso não seja *vezDoPC*, será chamada a função que o usuário remove as peças. Deve-se também verificar quantas peças o usuário removeu, e se *n* for 0, o usuário venceu a partida.

```
else:
            jogadorRemove = usuario_escolhe_jogada(n, m)
            n = n - jogadorRemove
            if jogadorRemove == 1:
                print('Você tirou uma peça\n')
            else:
                print('Você tirou', jogadorRemove, 'peças\n')
            if n == 0:
                print('Fim do jogo! Voce ganhou!')
                return 1
            vezDoPC = True
```

Enquanto a partida estiver em andamento e as funções para o usuário e o computador forem chamadas, deve-se mostrar quantas peças ainda restam no tabuleiro.

```
if n ==1:
            print('Agora resta apenas uma peça no
tabuleiro.\n')
        else:
            if n != 0:
                print('Agora restam,', n, 'peças no
tabuleiro.\n')
```

Com a função partida pronta, o jogo está funcional.

Este foi o meu primeiro artigo, e ficou bem longo, mas consegui explicar como desenvolvi esse jogo. Se você se interessou o código que fiz está no meu GitHub: https://github.com/joao-lucasilva/Jogo-do-Nim.

Se você chegou até aqui, obrigado pela leitura e deixe seu feedback para que eu possa melhorar em futuras publicações.