

Acertando na Mega Sena

Mário Leite

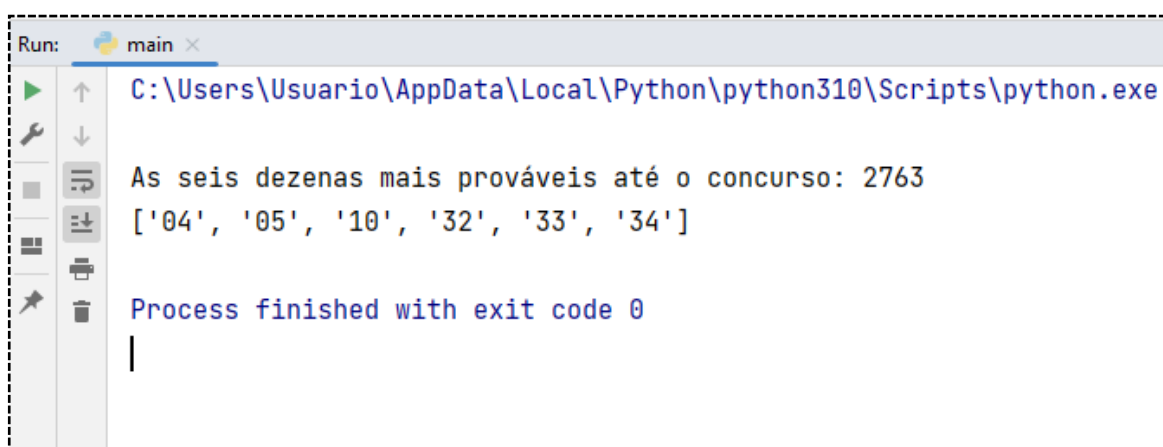
...

Existem na Internet muitos *sites* que apresentam soluções “mágicas” para acertar as seis dezenas da Mega Sena em um jogo simples. Muitos programas e aplicativos prometem resultados certos para os internautas; imaginem se fosse verdade! Com um prêmio total acumulado de 200 milhões de reais, o prêmio para cada apostador seria menos de **R\$ 1,30** (*um real e trinta centavos*) se todos fizessem o mesmo palpite indicado pelo *site*, considerando uma população de 170 milhões de brasileiros apostadores acima de 14 anos. A verdade é que “NÃO TEM COMO PREVER, COM EXATIDÃO, AS PRÓXIMAS SEIS DEZENAS QUE SERÃO SORTEADAS NA MEGA SENA”; esta deve ser a compreensão correta do assunto. Portanto, qualquer previsão de acerto nas seis dezenas da Mega Sena é pura especulação, pois não existe nenhum mecanismo lógico para essa previsão. Os sorteios das seis bolas numeradas de **01 a 60** são totalmente independentes um do outro.

“Acertando na Mega Sena” é apenas um título para esta postagem; não implica, necessariamente, que isto seja a verdade. Por outro lado, a Matemática oferece ferramentas que podem auxiliar na compreensão de eventos probabilísticos, como o famoso “Teorema do Valor Central”. Esse teorema pode mostrar a tendência de determinados valores agrupados em torno de uma média representativa de uma amostra grande de valores obtidos sob as mesmas condições. Em resumo, o “Teorema do Valor Central” é baseado em três pilares:

- **Independência:** As variáveis aleatórias devem ser independentes.
- **Identicamente distribuídas:** As variáveis devem ter a mesma distribuição.
- **Número suficiente de variáveis:** O número de variáveis deve ser grande o suficiente.

O banco de dados que contém os resultados dos sorteios da Mega Sena pode ser utilizado como base para utilizar essa ferramenta matemática. O programa “**DezenasMaisProváveis**” é uma solução de código em Python que utiliza esse teorema para mostrar as seis dezenas que mais foram sorteadas até o **Concurso 2763**, como mostra a **figura 1**. Essas informações poderiam ser usadas como base para futuras apostas, supondo que o alto número de repetições dessas seis dezenas possa ser uma “dica”.



```
Run: main x
C:\Users\Usuario\AppData\Local\Python\python310\Scripts\python.exe

As seis dezenas mais prováveis até o concurso: 2763
['04', '05', '10', '32', '33', '34']

Process finished with exit code 0
```

Figura 1 - Saída do programa “DezenasMaisProváveis” para os primeiros 2763 concursos da Mega Sena

```

'''
DezenasMaisProvaveis.py
-----
Acessa uma tabela de um banco de dados Access e mostra as seis dezenas médias
com o "Teorema do Valor Central".
-----
'''

import sys
import pyodbc
from collections import Counter

#Estabelece a conexão com o banco de dados Access
strConn = (
    r'DRIVER={Microsoft Access Driver (*.mdb, *.accdb)};'
    r'DBQ=D:\Livros\livro11\Dados\Loterias.mdb;'
)
objConn = pyodbc.connect(strConn)
cursor = objConn.cursor()

#Define o nome da tabela a ser acessada
tabela = "MegaSena"

#Obtém todos os registros da tabela
strSql = f"SELECT * FROM {tabela}"
cursor.execute(strSql)

#Carrega todos os registros
regs = cursor.fetchall()

#Divide as dezenas em dois blocos
LstDezBloco1 = []
LstDezBloco2 = []

#Itera pelos registros e divide as dezenas entre os dois blocos
for reg in regs:
    for i in range(6): #ajuste o range conforme o número de colunas que contêm as dezenas
        valor = reg[i]
        if(valor is None):
            continue #pula valores nulos

        try:
            dezena = int(valor) #converte a string para inteiro
        except ValueError:
            continue #pula valores que não podem ser convertidos para inteiro

        if(1 <= dezena <= 30):
            LstDezBloco1.append(dezena)
        elif(31 <= dezena <= 60):
            LstDezBloco2.append(dezena)

#Calcula as três dezenas mais frequentes em cada bloco de três dezenas
contadorBloco1 = Counter(LstDezBloco1)
contadorBloco2 = Counter(LstDezBloco2)
LstMaisFreqBloco1 = [dez for dez, freq in contadorBloco1.most_common(3)]
LstMaisFreqBloco2 = [dez for dez, freq in contadorBloco2.most_common(3)]

#Combine as duas listas e mantenha apenas seis dezenas
LstDezMaisProv = LstMaisFreqBloco1 + LstMaisFreqBloco2
LstDezMaisProv = list(set(LstDezMaisProv)) #remove duplicatas, se houver

```

```
#Ordena as dezenas e limita a seis
LstDezMaisProv.sort()
if(len(LstDezMaisProv) > 6):
    LstDezMaisProv = LstDezMaisProv[:6]

#Conta o total de registros na tabela
cursor.execute(f'SELECT COUNT(*) AS totalRegs FROM {tabela}')
totalRegs = cursor.fetchone()[0]

#Formata as dezenas para exibir com zero à esquerda se necessário
LstDezMaisProvFormatada = [f'{dez:02d}' for dez in LstDezMaisProv]

#Exibe as seis dezenas mais prováveis
print()
print(f"As seis dezenas mais prováveis até o concurso: {totalRegs}")
print(LstDezMaisProvFormatada)

#Fecha a conexão com o banco de dados
cursor.close()
objConn.close()
#Fim do programa "DezenasMaisProvaveis" -----
```