Laboratório de Programação Competitiva

Profa. Silvia Brandão

2024.1

Aula de hoje

- ➢ Elaboração de códigos
- ➢ Teoria dos Números x Teoria dos Números em Python
    - ➢ Teorema Chinês do Resto,
    - ➢ Crivo de Eratóstenes.

MOMENTO N2:

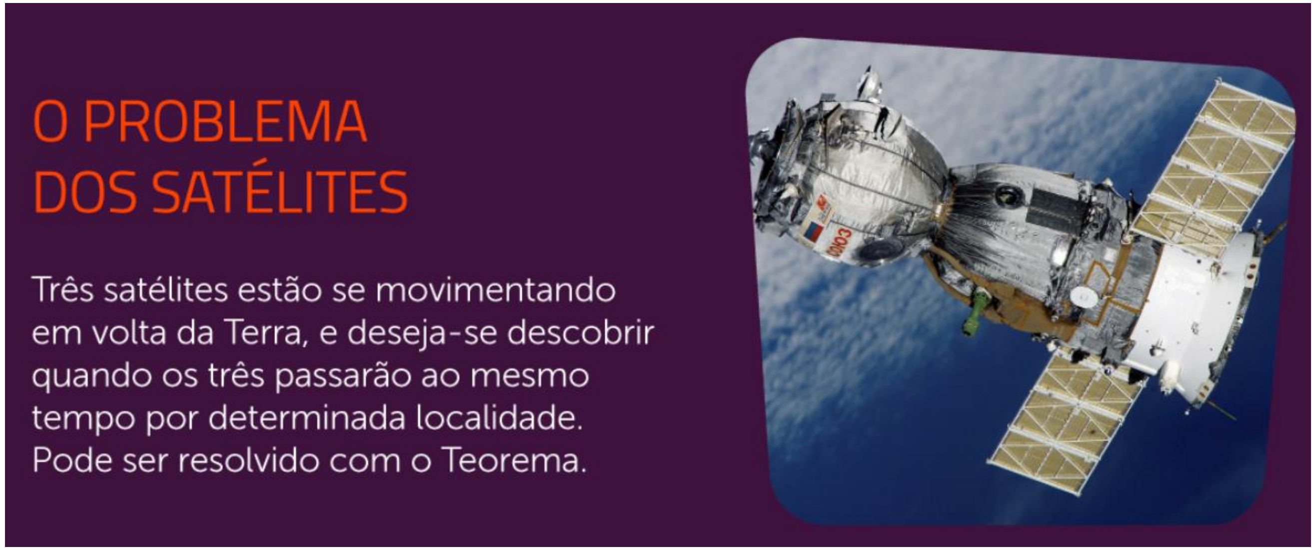
- Crie seu portifólio (notebook), no Google Colab, para o momento de avaliação N2. Qualquer exercício proposto, neste momento, deverá ser implementado no portifólio para posterior correção no valor de 10 pontos. Não se esqueça de incluir NOME e RA. Compartilhar comigo: silvia.brandao@uniube.br
- Lista de exercícios no Beecrowd, valor de 5 pontos.
- Avaliação prática, valor de 10 pontos.

Teoria dos Números em Python

➢Teorema Chinês do Resto

- Descoberto pelos chineses no início século XIII. Há indícios de que ele tenha surgido de problemas práticos enfrentados na época, problemas que envolviam astrologia; ligados ao movimento dos corpos celestes, entre outros.
- É um teorema muito importante da Álgebra, com aplicações interessantes.
- Para que o teorema possa ser enunciado, é importante relembrar alguns conceitos da Álgebra, como a divisibilidade e a congruência.

Aplicação: Teorema Chinês do Resto



Divisibilidade e Congruência

DIVISIBILIDADE

Dados dois inteiros *d* e *a*, diz-se que:

- *d* divide *a* ou que *d* é um divisor de *a*.
- Ou ainda que *a* é um múltiplo de *d* e escreve-se $d|a$ se existir $q \in Z$ com $a = qd$.
- Caso contrário, escreve-se $d \nmid a$.

CONGRUÊNCIA

Sejam $a, b, m \in Z$, diz-se que *a* é congruente a *b* módulo *m*, quando *a* e *b* deixam o mesmo resto quando dividido por *m*.

Denota-se por: $a \equiv b \ (mod\ m)$.

Definição

Dados três números inteiros a, b e m, *dizemos que a* **é congruente a** *b* **módulo** m *se* m$|$ $a - b$, *isto é, se* $a - b$ *é múltiplo de* m *Notação:* $a \equiv b \bmod(m)$.

Primos entre si ou coprimos

Lema 1. *Se $mdc(a, m) = 1$, então existe um inteiro x tal que:*

$$ax \equiv 1 \pmod{m}.$$

Tal inteiro é único módulo m. Se $mdc(a, m) > 1$, não existe x satisfazendo tal equação.

Demonstração. Pelo teorema de Bachet-Bézout, existem inteiros x e y tais que $ax+my = 1$. Analisando essa congruência módulo m, obtemos $ax \equiv 1 \pmod{m}$. Se y é outro inteiro que satisfaz a mesma congruência, temos $ax \equiv ay \pmod{m}$. Pelo primeiro lema, $x \equiv y \pmod{m}$. Se $d = mdc(a, m) > 1$, não podemos ter $d \mid m$ e $m \mid ax - 1$ pois $d \nmid ax - 1$.

Na teoria dos números, dois inteiros a e b são **primos entre si** ou **coprimos** se o único divisor comum a ambos é 1. Consequentemente, qualquer número primo que divide a não divide b, e vice e versa. Isso é equivalente a dizer que o mdc(a,b) é 1.

Exemplo: Os número 8 e 9 não são primos, porém eles são primos entre si, visto que 1 é o único divisor comum. Por outro lado, 6 e 9 não são primos entre si, pois ambos são divisíveis por 3.

TEOREMA CHINÊS DO RESTO

Sejam $n_1, n_2, n_3, \ldots, n_k$ números inteiros positivos tais que $MDC(n_i, n_j) = 1$, para $i \neq j$. O sistema de congruência linear

$$
\begin{aligned}
x &\equiv a_1 \pmod{n_1} \\
x &\equiv a_2 \pmod{n_2} \\
x &\equiv a_3 \pmod{n_3} \\
&\ldots \\
x &\equiv a_k \pmod{n_k}
\end{aligned}
$$

Admite uma solução simultânea, que é única no módulo do inteiro $n = n_1 n_2 n_3 \ldots n_k$.

Algoritmo: cálculo do Teorema do resto Chinês

Exemplo: Encontre x inteiro tal que: $x \equiv 1 \pmod{11}$ e $x \equiv 2 \pmod{7}$.

Para resolver o problema dado, temos m1=11 e m2=7 que são coprimos, **também conhecidos como números primos entre si**. Assim, podemos usar o Teorema Chinês do Resto para encontrar o valor de x. Siga os passos do algoritmo:

1. Calcule M=m1*m2:

- M=11*7=77

2. Calcule M1 = M/m1 e M2 = M/m2:

- M1=77/11=7
- M2=77/7=11

3. Calcule os inversos multiplicativos y1 e y2, usando Euclides e o Lema 1:

- Para m1=11, temos $7y1 \equiv 1 \pmod{11} \rightarrow y1 = 8$.
- Para m2=7, temos $11y2 \equiv 1 \pmod{7} \rightarrow y2 = 2$.

Aplicando Euclides:
11 | (7y1 - 1)
(7y1 - 1) = 11k
7.8 - 1 = 11.5

Aplicando Euclides:
7 | (11y2 - 1)
(11y2 - 1) = 7k
11.2 - 1 = 7.3

4. Calcule x=(a1*M1*y1 + a2*M2*y2)%M:

- x=((1*7*8)+(2*11*2))%77=(56+44)%77=100%77=23

Portanto, o valor de x inteiro que satisfaz as congruências dadas é x=23.

Teorema Chinês do Resto

Exercício 1: Usando o teorema Chinês do Resto, encontre o menor inteiro positivo x tal que:

$x \equiv 5 \pmod{7}$

$x \equiv 7 \pmod{11}$

$x \equiv 3 \pmod{13}$

R. a menor solução positiva é 887.

Exercício 2: Usando o teorema Chinês do Resto, encontre o menor inteiro positivo x tal que:

$x \equiv 2 \pmod{3}$

$x \equiv 3 \pmod{5}$

$x \equiv 2 \pmod{7}$

R. a menor solução positiva é 23.

Exercício - Teorema Chinês do Resto

5. Implemente em Python, a solução para o sistema de congruências lineares usando o Teorema Chinês do Resto. Para isso, use o sistema de congruências abaixo, como exemplo, para encontrar o valor de x.

$x \equiv 2 \pmod 3$
$x \equiv 3 \pmod 5$
$x \equiv 2 \pmod 7$

Sugestão: A função teorema_chines_do_resto() recebe duas listas restos e mod, onde rem contém os restos das congruências, [2,3,2] e mod contém os módulos correspondentes, [3,5,7].

Entrega pelo portifólio (notebook), no Google Colab.

Exercício - Solução

```python
'''
Teorema Chinês do Resto (CRT - Chinese Remainder Theorem) - fornece uma solução para sistemas de
congruências lineares
'''
import math

def euclid_estendido(a, b):
    if a == 0:
        return (b, 0, 1)
    else:
        mdc, x, y = euclid_estendido(b % a, a)
        return (mdc, y - (b // a) * x, x)

def teorema_chines_do_resto(restos, mod):
    n = len(restos)

    # Calcular o módulo total - produtos
    M = math.prod(mod)
    # Calcular Xi
    X = [M // m for m in mod]
    # Calcular inversos modulares
    inversos = [euclid_estendido(X[i], mod[i])[1] for i in range(n)]
    print(inversos)
    # Calcular a solução
    resultado = sum(restos[i] * X[i] * inversos[i] for i in range(n)) % M

    return resultado
```

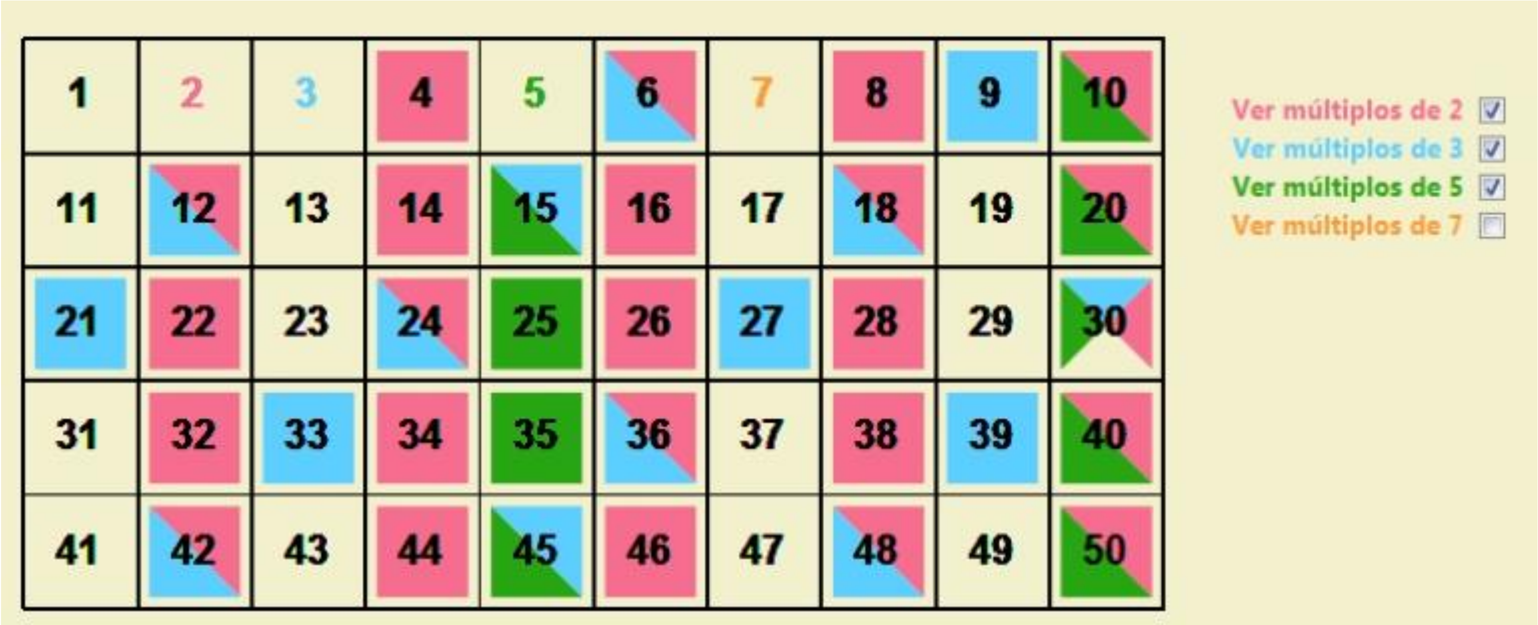
```python
# Exemplo de uso
#restos = [1,2]
#mod = [11,7]
#restos = [5,7,3]
#mod = [7,11,13]
restos = [2,3,2]
mod = [3,5,7]
print("Solução: ",
teorema_chines_do_resto(restos, mod))
```

Teoria dos Números em Python

➢ Crivo de Eratóstenes

- Eratóstenes foi um matemático grego que viveu entre os anos 276 a.C. até 194 a.C. Ele desenvolveu uma tabela, chamada de "Crivo de Eratóstenes", onde conseguiu determinar, não com uma fórmula, mas com uma tabela os números naturais primos.
- Na teoria pode ser feito para todos os números primos; porém, o inconveniente é que quanto maior for o nº primo, mais difícil de aplicar o Crivo de Eratóstenes, pois o esforço aliado ao tempo gasto começará a aumentar incrivelmente
- Crivo de Eratóstenes é um método para determinar todos os números primos menores ou iguais a um certo número.
- A palavra "crivo" refere-se a um utensílio que serve para separar diferentes componentes de uma mistura, retendo as substâncias maiores e deixando passar as substâncias de dimensões mais reduzidas. Usando o Crivo de Eratóstenes iremos separar os números primos dos números não primos (ou seja, do número um e dos números compostos).

Crivo de Eratóstenes

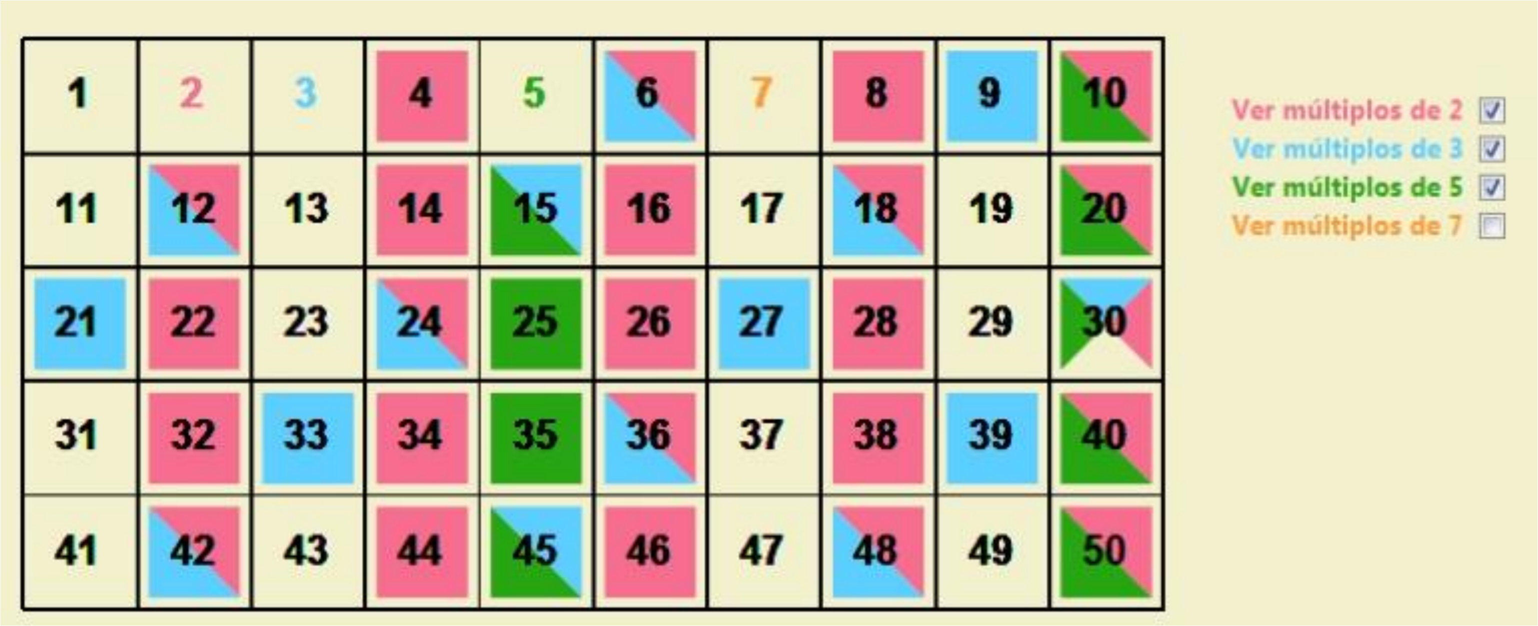


- Tente seguir os passos indicados na aplicação do aplicativo Geogebra:

https://www.geogebra.org/m/gNpjubWw

Exercício – Crivo de Eratóstenes

6. Implemente em Python, a solução para o Crivo de Erastótenes dado um n >= 0.



Entrega pelo portifólio (notebook), no Google Colab.

Exercício - Solução

```python
def crivo_eratostenes(n):
    if n <= 1:
        return []

    # Inicialmente, assumimos que todos os números de 2 até n são primos
    primo = [True] * (n + 1)
    primo[0] = primo[1] = False

    # Percorrendo os múltiplos dos números primos conhecidos
    for p in range(2, int(n ** 0.5) + 1):
        if primo[p]:
            # Marcando todos os múltiplos de p como não primos
            for i in range(p * p, n + 1, p):
                primo[i] = False

    # Coletando os números primos restantes
    primos = []
    for p in range(n + 1):
        if primo[p]:
            primos.append(p)

    return primos
```

```python
# Exemplo de uso:
n = int(input("Digite um número inteiro positivo n: "))
if n >= 0:
    primos = crivo_eratostenes(n)
    print("Números primos até", n, ":", primos)
else:
    print("Este número não é inteiro positivo.")
```

RESULTADO

```
Digite um número inteiro positivo n: 100
Números primos até 100 : [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97]
```

Exercícios

7. Escreva um programa que lê n e uma matriz A de inteiros de dimensão n x n, e:

a) verifica se A é simétrica
b) soma os elementos da diagonal principal de A
c) soma os elementos da diagonal secundária de A
d) exibe os números negativos e sua localização na matriz A
e) exibe os valores de máximo e mínimo da matriz A
f) lê uma outra matriz, a matriz B, de dimensão n x p e imprime a matriz C de dimensão n x p que é o produto de A por B

Utilize funções como: leia_matriz(), imprima_matriz(), eh_simétrica(), soma_diagP(), multiplica_matriz() , etc

Leituras:

- **ENQ 2020.1 👉 TEOREMA CHINÊS DOS RESTOS - PROFMAT (QUESTÃO 1) -** https://www.youtube.com/watch?v=YZbH4BslZaU&t=156s
- **POTI** - https://poti.impa.br/index.php/modulo/ver?modulo=5
- https://portaldaobmep.impa.br/index.php/modulo/ver?modulo=55
- **Como encontrar os primos através do Crivo de Eratóstenes** - https://matematicando.net.br/o-crivo-de-eratostenes-numeros-primos/
- Continue estudando:
  - https://www.w3schools.com/python/default.asp

Próxima Aula

Não se esqueçam do Uniube+

Tópicos Avançados em Algoritmos - Grafos e Estruturas de Dados.

-Grafos em Python: Busca em largura (BFS), busca em profundidade (DFS), árvores, grafos ponderados.

-Estruturas de Dados em Python: Pilhas, filas, listas ligadas, árvores, heaps, segment trees. Abordagem de árvores e grafos ponderados.