UNIVERSIDADE DE BRASÍLIA
Campus UnB Gama (FGA)

Curso: Engenharia de Software
Professor: Cristiane Loesch de Souza Costa **Data:** 07/05/2025
Disciplina: Matemática Discreta II **Turma:** T03

Aluno: _____ **Matrícula:** _____

Nota:_____

AVALIAÇÃO P1 (G1 e G2)
(PARTE 2)

QUESTÃO 9 (1,5 ponto)

Um estudante está desenvolvendo uma função de segurança para validar acessos em um sistema. Essa função baseia-se na ideia de "verificação modular", utilizando os conceitos de **congruência**, **primalidade**, **coprimalidade**, **Pequeno Teorema de Fermat**, **Teorema de Euler** e **exponenciação modular** eficiente.

Porém, o sistema da escola do professor utiliza um método especial para definir a base da potência. Ao invés desta fornecida diretamente, o valor da base a é definido como o resultado da divisão modular entre dois elementos de Z_n (Ex: $38 \oslash 79$ em Z_{252} , generalizando: $H \oslash G$ em Z_n)

Implemente um **programa em C** que tenha:

Dados de entrada

- três números inteiros positivos **H, G** e **n**, usados para calcular a base **a**
- um expoente **x**
- um módulo $\mathbf{n_1}$

  CALCULAR: $a^x \, mod \, n_1$

Etapas que o programa deve realizar:

1. **Verificar se** G **e** n **são primos utilizando o Algoritmo de Euclides**

   *** se não forem, justificar que a divisão não é possível**

2. **Calcular o inverso de** G **em** Z_n **utilizando divisões sucessivas**

   *** se** G^{-1} **não pertencer a** Z_n **verificar equivalência**

3. **Dividir** H **por** G **e encontrar** a

4. **Verificar se** a **e** n_1 **são coprimos.**

5. **Verificar se** n_1 **é primo.**

6. **Se** n_1 **for primo, aplicar o Pequeno Teorema de Fermat, e definir** $x_1 = n - 1$ **.**

7. **Caso contrário, aplicar o Teorema de Euler, e definir** $x_1 = \varphi(n)$ **(função totiente de Euler)**

***Implemente a função $\varphi(n_1)$ sem usar bibliotecas prontas, apenas com base nos fatores primos de n_1 .**

8. **Utilizar o teorema da divisão para decompor o expoente x na forma $x = x_1 * q + r$.**
9. **Reescrever a expressão $a^x \bmod n_1$ como: $a^x \equiv (((a^{x_1})^q \bmod n_1) \cdot (a^r \bmod n_1)) \bmod n_1$**
10. **Calcular os seguintes valores intermediários:**
    - $a^{x_1} \bmod n_1 = x_2$
    - $x_2^{\,q} \bmod n_1$
    - $a^r \bmod n_1$
11. **Combinar os resultados e imprimir o valor final da congruência: $((x_2^{\,q}) \cdot (a^r)) \bmod n_1$**

OBJETIVO:

Permitir que o programa execute todas as etapas acima e imprima os passos de forma clara, educativa e detalhada, como em uma explicação matemática.

A ENTREGA:

Data: 09/09/25

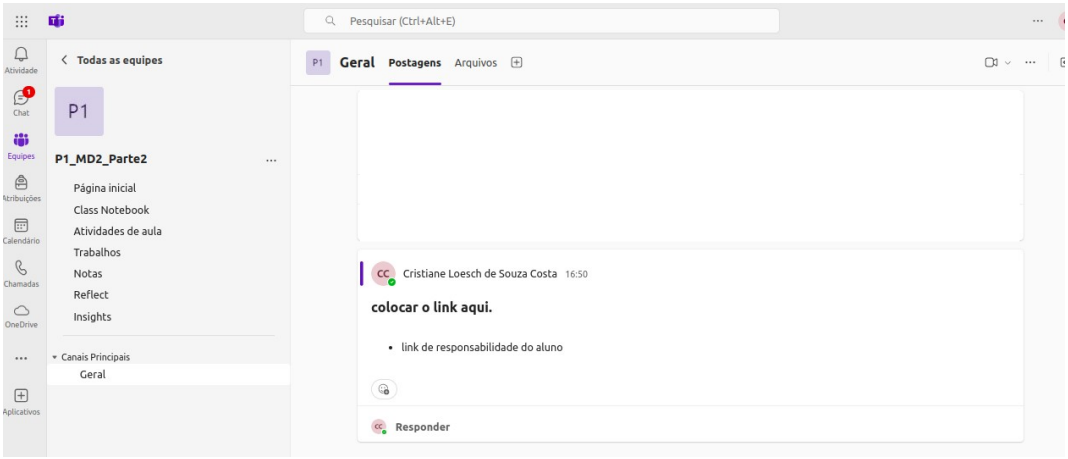
Hora: até as 18h

Local: Teams/UNB

O aluno poderá utilizar uma plataforma de sua escolha para desenvolver o código em linguagem C, (ex. One Compiler, Github, etc). Não serão aceitos códigos em outras linguagens.

Ao finalizar e testar o código o aluno deverá enviar o link do mesmo para o professor via post na plataformaTeams/UNB

Todos os alunos da turma foram adicionados à uma equipe para a publicação da atividade.



O link enviado é de responsabilidade exclusiva do aluno.

O professor deverá ter acesso liberado ao código, como usuário. Ao iniciar o código o mesmo deverá solicitar ao professor os números de entrada de sua preferência.

O professor poderá escolher os números que quiser para o teste.

Ao iniciar o teste o código deverá apresentar todo o passo a passo do problema como em uma explicação matemática.

O professor não fará alterações no código do aluno.

Se o teste não for bem sucedido por não encontrar o valor correto ou não funcionar, o aluno não receberá a pontuação referente a esta atividade

META:

A ideia da atividade é dar ao aluno a dimensão da matemática desenvolvida em sala de aula. Por isso é de suma importância que cada aluno seja responsável por sua atividade, especialmente no desenvolvimento matemático, considerando-se que esta será utilizada na próxima etapa da disciplina..

A discussão nos métodos de resolução, no entanto, é livre.