

---

**DEPARTAMENTO DE SISTEMAS E COMPUTAÇÃO**  
**Faculdade de Engenharia - UERJ**  
**Laboratório de Programação I (FEN06-04049)**

Tempo: 2 horas

Prova P1 - 2025-2

05 de novembro de 2025

Prof. João Araujo

---

1. Escreva um programa em **C** que leia um número qualquer e diga se ele é ou não primo. **(2pts)**

**2. Criptografia dos Primos Felizes**

Em alguns sistemas de criptografia, os números são transformados antes de serem usados como chaves. Um pesquisador propôs um método curioso: usar apenas **números primos felizes** para codificar mensagens, pois eles “sempre chegam à felicidade (1)” e simbolizam **chaves seguras**.

Números **primos** são números maiores que 1 que só são divisíveis por 1 e por eles mesmos.

Números **felizes** são números que, ao substituir o número pela soma dos quadrados de seus dígitos repetidamente, o processo eventualmente resulta em 1.

Por exemplo, 19 é um número feliz porque:

- a.  $1^2 + 9^2 = 1 + 81 = 82$
- b.  $8^2 + 2^2 = 64 + 4 = 68$
- c.  $6^2 + 8^2 = 36 + 64 = 100$
- d.  $1^2 + 0^2 + 0^2 = 1$

Números que não chegam a 1 são chamados de números **infelizes**. Por exemplo, 20 ou 89 não são números felizes porque o processo entra em um ciclo infinito sem nunca chegar a 1.

**Cifra de César:**

Um método de criptografia simples que consiste em deslocar cada letra do texto por um número fixo de posições no alfabeto.

**Tarefa**

Escreva um programa em C que realize as seguintes etapas:

- a. **Leitura do Número-Chave:** Leia um número inteiro  $N$  do teclado, no intervalo  $[1, 10000]$ .
- b. **Verificação de Primo Feliz:** Verifique se  $N$  é um Primo Feliz. Se  $N$  não for um Primo Feliz, o programa deve imprimir a mensagem “Não é primo feliz. Fim do programa.” e encerrar.
- c. **Criação da Chave de Criptografia:** Se  $N$  for um Primo Feliz, ele será usado para gerar uma chave de criptografia

$K$ :

$$K = N \pmod{26}$$

- O programa deve, em seguida, ler uma mensagem de texto (uma string, apenas letras maiúsculas A-Z, com no máximo 100 caracteres).
- Cada letra da mensagem deve ser deslocada  $K$  posições no alfabeto. Se o deslocamento ultrapassar ‘Z’, ele deve continuar a partir de ‘A’.
- Espaços ou outros caracteres diferentes de letras na mensagem original devem ser mantidos na mensagem criptografada.
- O programa deve imprimir a mensagem
  - “N é primo feliz! Chave K = [valor de K]”.
  - A mensagem criptografada.
- Exemplo de Entrada e Saída:

– Entrada:

- \* Número-Chave: 19
- \* Mensagem: CELACANTO PROVOCA MAREMOTO
- \* Saída:
- \* “19 é primo feliz! Chave  $K = 19$ ”
- \* “VXETVTGMH IKHOHVT FTKXFHMH”

Dica: Todo número infeliz entra em um ciclo que inclui o número 4. Portanto, se durante o processo de verificação de felicidade você encontrar o número 4, pode concluir que o número não é feliz.

- (a) Lógica Principal 1.0 pt. Leitura da entrada  $N$ , chamada correta das funções para testar primo e testar se é feliz para determinar se é um “Primo Feliz”, e impressão da mensagem inicial (seja positiva ou negativa).
- (b) Função que calcula se o número é feliz correta 3.0 pts. Cálculo correto da soma dos quadrados dos dígitos e, criticamente, tratamento eficiente de ciclos infinitos para números infelizes (evitando loops infinitos).
- (c) Preparação da Criptografia 1.5 pt. Cálculo correto da chave de deslocamento  $K$  e leitura da mensagem de texto de forma segura.
- (d) Função/Lógica de Cifra de César 2.0 pts. Implementação correta do deslocamento (Cifra de César) para a mensagem, tratando apenas letras maiúsculas e garantindo a circularidade do alfabeto (ex:  $Z + K = A, B$ , etc.).
- (e) Boas Práticas de Programação 0.5 pt. Código modularizado em funções, legibilidade, uso de constantes (se aplicável), e manuseio correto dos dados.

Boa Sorte, mas cada variável global desconta um ponto...

Envie os arquivos, sem o executável, para o classroom.

Não esqueça de colocar seu nome em cada arquivo fonte.

Os programas serão corrigidos usando o compilador **gcc** em ambiente **Linux**.

Obs.: Qualquer tentativa de **desonestidade intelectual**, conhecida popularmente como cola, será considerada falta grave e punida com nota **zero** na prova. Provas com **erros de sintaxe** (ou seja, erros de compilação) receberão nota **ZERO**. Seu programa deve ser compilável sem erros. Evite também warnings, apesar de não serem descontados pontos, pois eles podem indicar na realidade erros lógicos de seu programa.

Seu programa deve estar bem indentado, documentado e organizado. A indentação deve deixar clara a estrutura de subordinação dos comandos. Os comentários devem ser esclarecedores.