PROVA SUBSTITUTIVA

MAC 2166 – INTRODUÇÃO À COMPUTAÇÃO (ELÉTRICA E COMPUTAÇÃO)

PRIMEIRO SEMESTRE DE 2026

Nome completo: _____

NUSP: _____ **Turma:** _____

Assinatura: _____

Instruções:

1. Não destaque as folhas deste caderno.
2. A prova pode ser feita a lápis. Cuidado com a legibilidade.
3. A prova consta de **10 páginas** (incluindo a atual), totalizando **3 questões**. Verifique, antes de começar a prova, se o seu caderno de questões está completo.
4. Preencha o cabeçalho acima e os campos NUSP e Turma das folhas seguintes.
5. Não é permitido o uso de folhas avulsas para rascunho.
6. Não é permitido o uso de artefatos eletrônicos de qualquer natureza.
7. Não é permitida a consulta a livros, apontamentos ou colegas.
8. Não é necessário apagar rascunhos no caderno de questões.

DURAÇÃO DA PROVA: 2 horas

Questão	Nota
1	
2	
3	
Total	

Q1. [**3 pontos**] Considere o seguinte código.

```
#include <stdio.h>
#include <stdlib.h>
void imprima_vec(int N, int v[]) {
    for (int i = 0; i < N; i++)
        printf("%d ", v[i]);
    printf("\n");
}
void f(int N, int v[]) {
    int c[10];
    for (int i = 0; i < 10; i++)
        c[i] = 0;
    for (int i = 0; i < N; i++)
        c[v[i]]++;
    imprima_vec(10, c);

    int x = 0;
    for (int i = 0; i < 10; i++) {
        int qtd = c[i];
        for (int j = 0; j < qtd; j++) {
            v[x] = i;
            x++;
        }
    }
}
int main(int argc, char *argv[])
{
    int N;
    scanf("%d", &N);

    int n_digs = 0;
    int *digs = malloc(15 * sizeof(int));
    for (int i = 0; i < 15; i++)
        digs[i] = 0;

    while (N > 0) {
        digs[n_digs] = N % 10;
        n_digs++;
        N = N / 10;
    }
    imprima_vec(n_digs, digs);
```

```
    f(n_digs, digs);
    imprima_vec(n_digs, digs);

    int (*D)[3] = (int (*)[3])digs;
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++)
            printf("%d ", D[i][j]);
        printf("\n");
    }

    free(digs);
    return 0;
}
```

Determine a saída impressa pelo programa acima, supondo que a entrada é seu número USP.

Saída:

Rascunho:

NUSP: _____ **Turma:** _____

Q2. **[3 pontos]** Seja A uma matriz $M \times N$ com entradas a_{ij}. Os jogadores Lina e Carlos jogam o seguinte jogo: Lina escolhe uma linha i de A e Carlos escolhe uma coluna j de A (sem saber qual é a escolha do outro jogador), e revelam suas escolhas simultaneamente. O resultado do jogo é que Lina obtém a_{ij} pontos e Carlos obtém $-a_{ij}$ pontos.

Exemplo. Considere a seguinte matriz:

$$A = \begin{bmatrix} 0 & -1 & 1 \\ 1 & 0 & -1 \\ -1 & 1 & 0 \end{bmatrix}.$$

Suponha que as linhas de índice 0, 1 e 2 de A correspondam a *pedra*, *papel* e *tesoura*, respectivamente, e que as colunas 0, 1 e 2 de A correspondam também a *pedra*, *papel* e *tesoura*, respectivamente.

Se, por exemplo, Lina escolher a linha 1 (papel) e Carlos escolher a coluna 0 (pedra), Lina obtém $a_{10} = 1$ ponto e Carlos obtém $-a_{10} = -1$ ponto. Por outro lado, se Lina escolher a linha 1 (papel) e Carlos escolher a coluna 2 (tesoura), Lina obtém $a_{12} = -1$ ponto e Carlos obtém $-a_{12} = 1$ ponto.

Escolhas irracionais. Dada uma matriz A, a escolha de uma linha i por Lina é "irracional" se existir uma outra linha i' que é claramente melhor: vamos dizer aqui que a linha i é *irracional* se existe i' tal que $a_{ij} \leq a_{i'j}$ para todo j e, ademais, vale que $a_{ij} < a_{i'j}$ para algum j. Por exemplo, se a matriz fosse

$$B = \begin{bmatrix} 0 & -1 & 1 & 1 & 2 \\ 0 & 2 & -1 & -1 & -2 \\ 1 & 0 & 1 & 2 & 3 \\ -1 & 1 & -1 & -1 & -2 \end{bmatrix},$$

a linha 0 seria irracional, pois ela é "dominada" pela linha 2: qualquer que seja a escolha j de Carlos, o resultado do jogo é melhor ou igual para Lina se ela escolher a linha 2 em vez de escolher a linha 0 e, além disso, há escolhas de Carlos para as quais o resultado de Lina é estritamente melhor se ela escolher a linha 2. Na matriz do pedra-papel-tesoura, não há linhas irracionais.

Nesta questão, você deve escrever um programa que lê inteiros positivos M e N e as entradas de uma matriz $M \times N$ inteira A, e lista as linhas irracionais de A.

(a) Escreva inicialmente uma função

```
bool domina(int N, int v[], int w[])
```

que recebe um inteiro `N` e dois vetores `v` e `w` com `N` entradas, e que devolve `true` se `v[j]` $\geq$ `w[j]` para todo `j` e, ademais, `v[j]` $>$ `w[j]` para algum `j`.

(*b*) Escreva aqui a função `main` de seu programa. Ela deve ler inteiros M e N e as entradas de uma matriz inteira $M \times N$ e deve imprimir os índices das linhas irracionais da matriz lida, separados por espaços. Você deve usar a função `domina` do item (*a*), mesmo que você não a tenha escrito.

Por exemplo, se a entrada for B, a matriz 4×5 acima, então a saída deve ser
`0 3`
Se a entrada for a matriz 3×3 do pedra–papel–tesoura, seu programa não deve imprimir nada.

NUSP: _____ **Turma:** _____

Q3. **[4 pontos]** Ana é um colecionadora de bugigangas e decidiu vender N itens de sua coleção. As bugigangas são numeradas de 0 a $N-1$, e Ana atribuiu um preço a cada bugiganga. Zé também coleciona bugigangas, e quer comprar algumas das bugigangas de Ana. Zé tem um orçamento de exatamente V reais. Nesta questão, você escreverá as três funções principais de um programa que permitirá a Zé responder à seguinte pergunta:

Pergunta. É possível escolher algum subconjunto das N bugigangas da Ana que, no total, custam exatamente V reais?

(a) Escreva uma função

```
int custo_total(int N, int precos[N], int comprado[N])
```

que recebe o número de bugigangas `N` e vetores com os seguintes significados:

- `precos[i]` é o preço da i-ésima bugiganga ($0 \leq i <$ `N`);
- `comprado[i]` vale 0 ou 1. Se `comprado[i]` for 1, então Zé decidiu comprar a i-ésima bugiganga; se `comprado[i]` for 0, decidiu não comprá-la.

A função deve devolver quanto Zé pagaria no total. Isto é, esta função serve para Zé saber o custo de comprar os itens especificados no vetor `comprado`.

(b) Escreva uma função

```
void digitos(int b, int num, int N, int v[N])
```

que recebe um inteiro positivo `b`, um inteiro não-negativo `num`, um inteiro `N` e um vetor de inteiros com `N` posições `v`, e que coloca em `v` os dígitos de `num` na base `b`. Mais precisamente, dados `num` e `b`, a função deve preencher `v` de forma que valha o seguinte:

$$\texttt{num} = \texttt{v[0]}\texttt{b}^{0} + \texttt{v[1]}\texttt{b}^{1} + \texttt{v[2]}\texttt{b}^{2} + \cdots + \texttt{v[N-1]}\texttt{b}^{\texttt{N-1}},$$

onde os `v[i]` são inteiros entre 0 e `b` − 1. Por exemplo, se `b` = 10, `num` = 2345 e `N` = 6, sua função deve preencher `v` com os valores `{5, 4, 3, 2, 0, 0}`. Lembre-se que sua função deve funcionar para outros valores de `b`.

Observação. É garantido que `N` será sempre grande o suficiente para que os dígitos de `num` caibam em `v`.

(*c*) Escreva uma função

```
bool valor_exato(int V, int N, int precos[N])
```

que recebe um inteiro `V` (o orçamento de Zé), um inteiro `N` e um vetor `precos`. Ela deve devolver `true` se existe um subconjunto de bugigangas cuja soma dos preços é exatamente igual a V, e `false` caso contrário.

Observações. (1) Note que esta função responde à pergunta de Zé destacada no começo desta questão. (2) Você deve usar as funções de (*a*) e (*b*), mesmo que você não as tenha implementado. (3) Ademais, use neste item a função

```
int pot2(int N)
```

que recebe um inteiro não-negativo `N` e que devolve o valor de $2^{\texttt{N}}$.

Sugestão. Suponha, por exemplo, que `N = 3`. Escreva, na base 2, cada um dos números $0, 1, \ldots, 2^{\texttt{N}} - 1 = 2^3 - 1 = 7$.

NUSP: _____ **Turma:** _____

Espaço para a solução do item (c):

Observação: Nessa questão, você não deve escrever a função `main`.

(Rascunho)

NUSP: _____ **Turma:** _____

(Rascunho)