João Pedro Araujo Queiroz Barbosa
Ciência da computação

1-

a) Variáveis globais: x. Variáveis locais:

- Na função main: numero, outroNumero.
- Na função Teste1: numero.
- Na função Teste2: numero, valor.
- Na função Teste3: n, valor.

b) Na função Teste1, o parâmetro é passado por valor. Na função Teste2, o parâmetro é passado por referência. Na função Teste3, o parâmetro é passado por valor.

c)

- Início do programa: x = 20, numero = 10, outroNumero (indefinido).
- Após a chamada de Teste1: numero = 10, x = 21.
- Após a impressão "Valor de numero (após Teste1) = 10" e "X = 21": As variáveis permanecem iguais.
- Após a chamada de Teste2: numero = 110, x = 22.
- Após a impressão "Valor de numero (após Teste2) = 110" e "X = 22": As variáveis permanecem iguais.
- Após a chamada de Teste3: n = 110, valor = 200, x = 21. A variável outroNumero recebe o valor de n + valor = 310.
- Após a impressão "Valor de outro número (após Teste3) = 310" e "X = 21": As variáveis permanecem iguais.
- Fim do programa: As variáveis são liberadas da memória.

d) O que é impresso na tela:

Valor de numero (após Teste1) = 10 X = 21 Valor de numero (após Teste2) = 110 X = 22 Valor de outro numero (após Teste3) = 310 X = 21

2-

```
#include <iostream>
using namespace std;

const int TAM_MAX = 100;

bool inserirSemRepetir(int vetor[], int& n, int valor, int tam_max) {
    if (n >= tam_max) {
        cout << "Erro: vetor cheio." << endl;
        return false;
    }

    for (int i = 0; i < n; i++) {
        if (vetor[i] == valor) {
            cout << "Erro: elemento já existe no vetor." << endl;
            return false;
        }
    }
    vetor[n++] = valor;
    return true;
}

void listar(int vetor[], int n) {
    for (int i = 0; i < n; i++) {
```

```
        cout << vetor[i] << " ";
    }
    cout << endl;
}

void intercalar(int vetor1[], int n1, int vetor2[], int n2, int vetor3[], int& n3) {
    int i = 0, j = 0, k = 0;
    while (i < n1 && j < n2) {
        if (vetor1[i] < vetor2[j]) {
            vetor3[k++] = vetor1[i++];
        } else {
            vetor3[k++] = vetor2[j++];
        }
    }
    while (i < n1) {
        vetor3[k++] = vetor1[i++];
    }
    while (j < n2) {
        vetor3[k++] = vetor2[j++];
    }
    n3 = k;
}

void intersecao(int vetor1[], int n1, int vetor2[], int n2, int vetor3[], int& n3) {
    n3 = 0;
```

```
    for (int i = 0; i < n1; i++) {
        for (int j = 0; j < n2; j++) {
            if (vetor1[i] == vetor2[j]) {
                inserirSemRepetir(vetor3, n3, vetor1[i], TAM_MAX);
            }
        }
    }
}

void uniao(int vetor1[], int n1, int vetor2[], int n2, int vetor3[], int& n3) {
    n3 = 0;
    for (int i = 0; i < n1; i++) {
        inserirSemRepetir(vetor3, n3, vetor1[i], TAM_MAX);
    }
    for (int i = 0; i < n2; i++) {
        inserirSemRepetir(vetor3, n3, vetor2[i], TAM_MAX);
    }
}

void removerPeloIndice(int vetor[], int& n, int indice) {
    if (indice < 0 || indice >= n) {
        cout << "Erro: índice inválido." << endl;
        return;
    }
    for (int i = indice; i < n - 1; i++) {
```

```
        vetor[i] = vetor[i + 1];
    }
    n--;
}

int main() {
    return 0;
}
```