Lista de Exercícios de Algoritmos – 1º Semestre 2023

Igor Natan Silva Ferreira - 1º SEM - DSM

EXERCÍCIO 1:

```
#include <iostream>

using namespace std;

int main() {
   int numCavalos;
   int numFerraduras;

   cout << "Informe o número de cavalos: ";
   cin >> numCavalos;

   numFerraduras = numCavalos * 4;

   cout << "O número total de ferraduras necessárias é: " << numFerraduras << endl;

   return 0;
}
```

EXERCÍCIO 2:

```
#include <iostream>
#include <locale.h>

using namespace std;

int main() {
   setlocale(LC_ALL, "Portuguese");
   int qtdPaes, qtdBroas;
   float valor_pao = 0.12, valor_broa = 1.50;
   float total, poupanca;

   cout << "Informe a quantidade de pães vendidos: ";
   cin >> qtdPaes;

   cout << "Informe a quantidade de broas vendidas: ";
   cin >> qtdBroas;

   total = (qtdPaes * valor_pao) + (qtdBroas * valor_broa);
   poupanca = total * 0.1;
```

```
    cout << "Total arrecadado: R$ " << total << endl;
    cout << "Valor a ser guardado na poupança: R$ " << poupanca << endl;

    return 0;
}
```

EXERCÍCIO 3:

```
#include <iostream>
#include <locale>

using namespace std;

int main() {
    setlocale(LC_ALL, "Portuguese");

    int quantidadePequenas, quantidadeMedias, quantidadeGrandes;
    float valorPequena = 10.0, valorMedia = 12.0, valorGrande = 15.0;
    float valorTotal;

    cout << "Informe a quantidade de camisetas pequenas vendidas: ";
    cin >> quantidadePequenas;

    cout << "Informe a quantidade de camisetas médias vendidas: ";
    cin >> quantidadeMedias;

    cout << "Informe a quantidade de camisetas grandes vendidas: ";
    cin >> quantidadeGrandes;

    valorTotal = (quantidadePequenas * valorPequena) + (quantidadeMedias * valorMedia) +
(quantidadeGrandes * valorGrande);

    cout << "Valor total arrecadado: R$ " << valorTotal << endl;

    return 0;
}
```

EXERCÍCIO 4:

```
#include <iostream>
#include <locale>

using namespace std;

int main() {
    setlocale(LC_ALL, "Portuguese");

    int pratoEscolhido, sobremesaEscolhida, bebidaEscolhida;
```

```
    int caloriasPrato, caloriasSobremesa, caloriasBebida;
    int caloriasTotal;
    int caloriasPratos[] = {180, 230, 250, 350};
    int caloriasSobremesas[] = {75, 110, 170, 200};
    int caloriasBebidas[] = {20, 70, 100, 65};

    cout << "Escolha o prato (1- Vegetariano, 2- Peixe, 3- Frango, 4- Carne): ";
    cin >> pratoEscolhido;

    cout << "Escolha a sobremesa (1- Abacaxi, 2- Sorvete Diet, 3- Mousse Diet, 4- Mousse
Chocolate): ";
    cin >> sobremesaEscolhida;

    cout << "Escolha a bebida (1- Chá, 2- Suco de Laranja, 3- Suco de Melão, 4- Refrigerante
Diet): ";
    cin >> bebidaEscolhida;

    caloriasPrato = caloriasPratos[pratoEscolhido - 1];
    caloriasSobremesa = caloriasSobremesas[sobremesaEscolhida - 1];
    caloriasBebida = caloriasBebidas[bebidaEscolhida - 1];

    caloriasTotal = caloriasPrato + caloriasSobremesa + caloriasBebida;

    cout << "Calorias da refeição: " << caloriasTotal << " calorias." << endl;

    return 0;
}
```

EXERCÍCIO 5:

```
#include <iostream>
#include <locale>

using namespace std;

int main() {
    setlocale(LC_ALL, "Portuguese");

    string destino;
    char opcaoRetorno;
    float precoPassagem;

    cout << "Informe o destino da viagem (Norte, Nordeste, Centro-Oeste ou Sul): ";
    cin >> destino;

    cout << "A viagem inclui retorno? (S para Sim / N para Não): ";
    cin >> opcaoRetorno;
```

```
    if (destino == "Norte") {
        if (opcaoRetorno == 'S') {
            precoPassagem = 900.0;
        } else {
            precoPassagem = 500.0;
        }
    } else if (destino == "Nordeste") {
        if (opcaoRetorno == 'S') {
            precoPassagem = 650.0;
        } else {
            precoPassagem = 350.0;
        }
    } else if (destino == "Centro-Oeste") {
        if (opcaoRetorno == 'S') {
            precoPassagem = 600.0;
        } else {
            precoPassagem = 350.0;
        }
    } else if (destino == "Sul") {
        if (opcaoRetorno == 'S') {
            precoPassagem = 550.0;
        } else {
            precoPassagem = 300.0;
        }
    } else {
        cout << "Destino inválido!" << endl;
        return 0;
    }

    cout << "O preço da passagem é: R$ " << precoPassagem << endl;

    return 0;
}
```

EXERCÍCIO 6:

```
#include <iostream>
#include <locale>

using namespace std;

int main() {
    setlocale(LC_ALL, "Portuguese");

    int idade;
    float peso;
    float dosagem;
    int gotas;
```

```
    cout << "Informe a idade do paciente: ";
    cin >> idade;

    cout << "Informe o peso do paciente (em quilos): ";
    cin >> peso;

    if (idade >= 12) {
        if (peso >= 60) {
            dosagem = 1000.0;
        } else {
            dosagem = 875.0;
        }
    } else {
        if (peso >= 5 && peso <= 9) {
            dosagem = 125.0;
        } else if (peso >= 9.1 && peso <= 16) {
            dosagem = 250.0;
        } else if (peso >= 16.1 && peso <= 24) {
            dosagem = 375.0;
        } else if (peso >= 24.1 && peso <= 30) {
            dosagem = 500.0;
        } else if (peso > 30) {
            dosagem = 750.0;
        } else {
            cout << "Peso inválido!" << endl;
            return 0;
        }
    }

    gotas = (dosagem / 500.0) * 20;

    cout << "Receita:" << endl;
    cout << "Dosagem: " << dosagem << " mg" << endl;
    cout << "Quantidade de gotas por dose: " << gotas << " gotas" << endl;

    return 0;
}
```

EXERCÍCIO 7:

```
#include <iostream>
#include <locale>

using namespace std;

int main() {
    setlocale(LC_ALL, "Portuguese");
```

```
    int n;
    int termoAnterior = 0;
    int termoAtual = 1;
    int proximoTermo;

    cout << "Informe a quantidade de termos da série de Fibonacci: ";
    cin >> n;

    cout << "Série de Fibonacci:" << endl;

    cout << termoAnterior << " ";

    for (int i = 1; i < n; i++) {
        cout << termoAtual << " ";

        proximoTermo = termoAnterior + termoAtual;
        termoAnterior = termoAtual;
        termoAtual = proximoTermo;
    }

    cout << endl;

    return 0;
}
```

EXERCÍCIO 8:

```
#include <iostream>
#include <locale>

using namespace std;

int main() {
    setlocale(LC_ALL, "Portuguese");

    int numQuadros = 64;
    unsigned long long int totalGraos = 0;
    unsigned long long int graosNoQuadro = 1;

    for (int i = 1; i <= numQuadros; i++) {
        totalGraos += graosNoQuadro;
        graosNoQuadro *= 2;
    }

    cout << "A somatória do número de grãos de trigo em um tabuleiro de xadrez é: " <<
totalGraos << endl;
```

```
    return 0;
}
```

EXERCÍCIO 9:

```
#include <iostream>
#include <locale>

using namespace std;

int main() {
    setlocale(LC_ALL, "Portuguese");

    int N;
    cout << "Informe o tamanho dos vetores A e B: ";
    cin >> N;

    int A[N], B[N], C[N], D[N];

    // Ler os vetores A e B
    cout << "Informe os elementos do vetor A:" << endl;
    for (int i = 0; i < N; i++) {
        cin >> A[i];
    }

    cout << "Informe os elementos do vetor B:" << endl;
    for (int i = 0; i < N; i++) {
        cin >> B[i];
    }

    // Calcular a soma dos elementos de A
    int somaA = 0;
    for (int i = 0; i < N; i++) {
        somaA += A[i];
    }

    // Calcular a soma dos elementos de B
    int somaB = 0;
    for (int i = 0; i < N; i++) {
        somaB += B[i];
    }

    // Obter o vetor C (soma dos vetores A e B)
    for (int i = 0; i < N; i++) {
        C[i] = A[i] + B[i];
    }

    // Obter o vetor D (subtração de B de A)
```

```
    for (int i = 0; i < N; i++) {
        D[i] = A[i] - B[i];
    }

    // Calcular o produto escalar de A por B
    int produtoEscalar = 0;
    for (int i = 0; i < N; i++) {
        produtoEscalar += A[i] * B[i];
    }

    // Imprimir os resultados
    cout << "Soma dos elementos de A: " << somaA << endl;
    cout << "Soma dos elementos de B: " << somaB << endl;

    cout << "Vetor C (A + B): ";
    for (int i = 0; i < N; i++) {
        cout << C[i] << " ";
    }
    cout << endl;

    cout << "Vetor D (A - B): ";
    for (int i = 0; i < N; i++) {
        cout << D[i] << " ";
    }
    cout << endl;

    cout << "Produto escalar de A por B: " << produtoEscalar << endl;

    return 0;
}
```

EXERCÍCIO 10:

```
#include <iostream>
#include <locale>

using namespace std;

void bubbleSort(int vetor[], int tamanho) {
    for (int i = 0; i < tamanho - 1; i++) {
        for (int j = 0; j < tamanho - i - 1; j++) {
            if (vetor[j] > vetor[j + 1]) {
                int temp = vetor[j];
                vetor[j] = vetor[j + 1];
                vetor[j + 1] = temp;
            }
        }
    }
```

```
}

int main() {
    setlocale(LC_ALL, "Portuguese");

    int n;
    cout << "Informe o tamanho do vetor: ";
    cin >> n;

    int vetor[n];

    cout << "Informe os elementos do vetor:" << endl;
    for (int i = 0; i < n; i++) {
        cin >> vetor[i];
    }

    bubbleSort(vetor, n);

    cout << "Vetor ordenado em ordem crescente: ";
    for (int i = 0; i < n; i++) {
        cout << vetor[i] << " ";
    }
    cout << endl;

    return 0;
}
```

EXERCÍCIO 11:

```
#include <iostream>
#include <locale>

using namespace std;

int main() {
    setlocale(LC_ALL, "Portuguese");

    const int linhas = 10;
    const int colunas = 10;
    int matriz[linhas][colunas];

    cout << "Informe os elementos da matriz 10x10:" << endl;
    for (int i = 0; i < linhas; i++) {
        for (int j = 0; j < colunas; j++) {
            cin >> matriz[i][j];
        }
    }
```

```
    int maiorValor = matriz[0][0];
    int linhaMaiorValor = 0;
    int colunaMaiorValor = 0;

    for (int i = 0; i < linhas; i++) {
        for (int j = 0; j < colunas; j++) {
            if (matriz[i][j] > maiorValor) {
                maiorValor = matriz[i][j];
                linhaMaiorValor = i;
                colunaMaiorValor = j;
            }
        }
    }

    cout << "Localização do maior valor:" << endl;
    cout << "Linha: " << linhaMaiorValor << endl;
    cout << "Coluna: " << colunaMaiorValor << endl;

    return 0;
}
```

EXERCÍCIO 12:

```
#include <iostream>
#include <locale>

using namespace std;

int main() {
    setlocale(LC_ALL, "Portuguese");

    int NUM_PACIENTES = 4;
    int NUM_HORAS = 24;

    int pulso[NUM_HORAS][NUM_PACIENTES];

    // Leitura dos valores das pulsações
    for (int paciente = 0; paciente < NUM_PACIENTES; paciente++) {
        cout << "Paciente " << paciente + 1 << ":" << endl;
        for (int hora = 0; hora < NUM_HORAS; hora++) {
            cout << "Digite o valor da pulsação para a hora " << hora + 1 << ": ";
            cin >> pulso[hora][paciente];
        }
        cout << endl;
    }

    // Cálculo e apresentação da média das pulsações para cada paciente
    cout << "Média das pulsações para cada paciente:" << endl;
```

```
    for (int paciente = 0; paciente < NUM_PACIENTES; paciente++) {
        int soma = 0;
        for (int hora = 0; hora < NUM_HORAS; hora++) {
            soma += pulso[hora][paciente];
        }
        double media = static_cast<double>(soma) / NUM_HORAS;
        cout << "Paciente " << paciente + 1 << ": " << media << endl;
    }
    cout << endl;

    // Identificação da cama do paciente com maior valor médio das pulsações
    double maiorMedia = 0;
    int camaMaiorMedia = 0;

    for (int paciente = 0; paciente < NUM_PACIENTES; paciente++) {
        int soma = 0;
        for (int hora = 0; hora < NUM_HORAS; hora++) {
            soma += pulso[hora][paciente];
        }
        double media = soma / static_cast<double>(NUM_HORAS);

        if (media > maiorMedia) {
            maiorMedia = media;
            camaMaiorMedia = paciente + 1;
        }
    }

    cout << "Cama do paciente com maior valor médio das pulsações: Cama " <<
camaMaiorMedia << endl;

    return 0;
}
```

EXERCÍCIO 13:

```
#include <iostream>

using namespace std;

int main() {
    int matriz1[4][4];
    int matriz2[4][4];
    int matrizMaior[4][4];

    cout << "Digite os elementos da primeira matriz:" << endl;
    for (int i = 0; i < 4; i++) {
        for (int j = 0; j < 4; j++) {
            cin >> matriz1[i][j];
```

```
        }
    }

    cout << "Digite os elementos da segunda matriz:" << endl;
    for (int i = 0; i < 4; i++) {
        for (int j = 0; j < 4; j++) {
            cin >> matriz2[i][j];
        }
    }

    for (int i = 0; i < 4; i++) {
        for (int j = 0; j < 4; j++) {
            matrizMaior[i][j] = (matriz1[i][j] > matriz2[i][j]) ? matriz1[i][j] : matriz2[i][j];
        }
    }

    cout << "Matriz com os maiores elementos:" << endl;
    for (int i = 0; i < 4; i++) {
        for (int j = 0; j < 4; j++) {
            cout << matrizMaior[i][j] << " ";
        }
        cout << endl;
    }

    return 0;
}
```

EXERCÍCIO 14:

```
#include <iostream>
#include <cmath>

using namespace std;

double calcularVolumeEsfera(double raio) {
    double volume = (4.0/3.0) * M_PI * pow(raio, 3);
    return volume;
}

int main() {
    double raio;

    cout << "Digite o raio da esfera: ";
    cin >> raio;

    double volume = calcularVolumeEsfera(raio);

    cout << "O volume da esfera é: " << volume << endl;
```

```
    return 0;
}
```

EXERCÍCIO 15:

```
#include <iostream>
#include <cstdlib>
#include <ctime>

using namespace std;

void preencherMatrizAleatoria(int matriz[][3]) {
    srand(time(0));

    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            matriz[i][j] = rand() % 101 - 50;
        }
    }
}

int contarZeros(int matriz[][3]) {
    int contador = 0;

    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            if (matriz[i][j] == 0) {
                contador++;
            }
        }
    }

    return contador;
}

int main() {
    int matriz[3][3];

    preencherMatrizAleatoria(matriz);

    cout << "Matriz gerada:" << endl;

    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            cout << matriz[i][j] << " ";
        }
        cout << endl;
```

```
    }

    int quantidadeZeros = contarZeros(matriz);

    cout << "Quantidade de zeros na matriz: " << quantidadeZeros << endl;

    return 0;
}
```

EXERCÍCIO 16:

```
#include <iostream>

using namespace std;

bool primo(int numero) {
    if (numero <= 1) {
        return false;
    }

    for (int i = 2; i * i <= numero; i++) {
        if (numero % i == 0) {
            return false;
        }
    }

    return true;
}

int main() {
    int numero;

    cout << "Digite um número inteiro e positivo: ";
    cin >> numero;

    bool resultado = primo(numero);

    if (resultado) {
        cout << "O número é primo." << endl;
    } else {
        cout << "O número não é primo." << endl;
    }

    return 0;
}
```

EXERCÍCIO 17:

```cpp
#include <iostream>

using namespace std;

bool perfeito(int numero) {
    int somaDivisores = 0;

    for (int i = 1; i < numero; i++) {
        if (numero % i == 0) {
            somaDivisores += i;
        }
    }

    return (somaDivisores == numero);
}

int main() {
    int numero;

    cout << "Digite um número inteiro: ";
    cin >> numero;

    bool resultado = perfeito(numero);

    if (resultado) {
        cout << "O número é perfeito." << endl;
    } else {
        cout << "O número não é perfeito." << endl;
    }

    return 0;
}
```