

LISTA DE EXERCÍCIOS

Página 1 de 2

Curso <i>Bacharelado em Ciência da Computação</i>		Unidade <i>Instituto de Ciências Exatas e Tecnológicas</i>
Disciplina <i>AP1 – Algoritmos e Programação 2 – ICE 0607</i>		
Turma <i>2º Período Turma A</i>	Data <i>25/08/2025</i>	Professor(a) <i>Ana Paula Freitas Vilela Boaventura</i>

ORIENTAÇÕES PARA A RESOLUÇÃO - O conteúdo exigido para resolução desta lista de exercícios compreende os seguintes capítulos no *Plano de Ensino* da disciplina: **Funções em C – Variáveis globais e locais.**

1 – Em linguagem C, a organização de programas em módulos e o uso de funções permitem melhor estruturação e reutilização do código. Nesse contexto:

- Explique a diferença entre variáveis globais e variáveis locais, destacando o escopo e o tempo de vida de cada uma.
- Por que é considerado uma boa prática limitar o uso de variáveis globais em programas em C?

Para os exercícios 2, 3, 4 e 5 declare uma variável global `g_NUM` do tipo inteiro.

2 – Crie uma função `int verifica_par_impar()`, que leia o valor global e verifique se é par ou ímpar. Ao final, retorne o valor da verificação para a função `main()`, para imprimir o resultado.

3 – Crie uma função `int verifica_positivo_negativo()`, que leia o valor global e verifique se é positivo (retorna 1) ou não (retorna 0). Ao final, retorne o valor da verificação para a função `main()`, para imprimir o resultado.

4 – Crie uma função `int calcula_soma()`, que declare uma variável local `int x`. A função deverá ler o valor `x` e calcular a soma entre `x` e a variável global `num`. Ao final, retorne o valor da soma para a função `main()`, para imprimir o resultado.

5 – Crie uma função `int verifica_ano_bissexto()`, que leia a variável global e verifique se é um ano bissexto (retorna 1) ou não (retorna 0). Ao final, retorne o valor da verificação para a função `main()`, para imprimir o resultado.

Para os exercícios 6 e 7, declare uma variável global `g_N` do tipo inteiro.

6 – Crie uma função `int calcula_fibonacci()`. A função principal deverá ler o valor `g_N`, chamar a função para calcular o Fibonacci de `g_N`. Ao final, `calcula_fibonacci()` retorna o valor da soma para a função `main()`, para imprimir o resultado.

7 – Crie uma função `int calcula_fatorial()`. A função principal deverá ler o valor `g_N`, chamar a função para calcular o Fatorial de `g_N`. Ao final, `calcula_fatorial()` retorna o valor da soma para a função `main()`, para imprimir o resultado.

8 – Crie uma variável global float g_vetor[5]. Em seguida, crie a função float calcula_media_aritmetica(), que calcule a média aritmética dos elementos do vetor. A função main() deverá ler os elementos de g_vetor, chamar a função calcula_media_aritmetica e imprimir o resultado.

9 – A partir do trecho de código abaixo, complete o programa criando as funções que faltam, inclusive, a função main().

```
#include <stdio.h>
```

```
typedef struct{  
    int matricula;  
    float notas[3];  
}aluno;
```

```
aluno g_turma[5];
```

```
void le_aluno ( );  
void imprime_turma( );  
float calcula_media_aluno( );
```

```
int main()  
{  
    aluno g_turma[5];  
  
    for(i=0; i< 5; i++)  
    {  
        le_aluno( ); // função que lê de teclado  
                        // os dados do aluno  
    }  
    imprime_turma ( ); //Função que imprime os dados de todos os alunos
```

```
    for(i=0; i< 5; i++)  
    {  
        media = calcula_media_aluno( );// função que calcula a média das notas de um aluno  
  
        printf("Aluno %d - Media = %f\n"); // imprime os dados do aluno  
    }  
    return 0;  
}
```