Fundamentos de Programação

Prof. Italo Mendes da Silva Ribeiro

Lista 4

Não apague as funções desenvolvidas do seu código-fonte, pois poderão ser utilizadas em questões posteriores.

Sempre que uma função recebe um vetor, também deve receber o tamanho do vetor.

1 – Inicialize um vetor com os números 71, 62, 16, 11, 25 e 31. Mostre os elementos com índice par. Em seguida, mostre os elementos do vetor com valor par.

```c
int main()
{
    int i;
    int numeros[6] = {71, 62, 16, 11, 25, 31};

    printf("Os elementos de indice par sao: \n");
    for(i = 0; i < 6; i += 2){

        printf("%i, ", numeros[i]);      // 71, 16, 25
    }

    printf("\nOs elementos com valores par sao: \n");
    for(i = 0; i < 6; i++){

        if(numeros[i] % 2 == 0)
            printf("%i, ", numeros[i]); // 62, 16
    }

    return 0;
}
```

2 – Inicialize um vetor com os números 54, 42, 33, 18, 75 e 72. Mostre os elementos com índice ímpar. Em seguida, mostre os elementos do vetor com valor par.

3 – Inicialize um vetor com os números 102, 450, 29, 559, 315, 94 e 120. Escreva os elementos com índice múltiplo de 3. Em seguida, mostre os elementos divisíveis por 5.

4 – Dado o vetor [35, 602, 100, 38, 82, 62, 510], mostre os elementos com índice par e que são múltiplos de 2 e 5.

---

5 – Implemente um algoritmo que leia 5 números e os armazene em um vetor. Em seguida, escreva os números ímpares armazenados nas posições de índice ímpar do vetor.

```c
int main()
{
    int i;
    int numeros[5];

    printf("Informe os 5 elementos do vetor: \n");
    for(i = 0; i < 5; i++){

        scanf("%i", &numeros[i]);  // 5, 9, 10, 12, 15
    }

    printf("Os elementos impares com indices impares sao: \n");
    for(i = 1; i < 5; i += 2){

        if(numeros[i] % 2 != 0)
            printf("%i, ", numeros[i]); // 9
    }

    return 0;
}
```

6 – Implemente um algoritmo que leia 5 números e os armazene em um vetor. Mostre o maior elemento do vetor.

7 – Implemente um algoritmo que leia 5 números e os armazene em um vetor. Mostre o menor elemento do vetor, dentre os elementos com índice ímpar.

8 – Escreva um programa que armazena 6 números em um vetor e informa o maior elemento do vetor que é divisível por 5.

---

9 – Dado o vetor de inteiros [3, 12, 25, 9], escreva uma função que receba um vetor de inteiros e retorne a soma dos elementos do vetor.

```c
#define TAMANHO 4

int somaVet(int vet[], int tamanho){

    int i, soma = 0;

    for(i = 0; i < tamanho; i++){
        soma += vet[i];
    }

    return soma;
}

int main()
{
    int numeros[TAMANHO] = {3, 12, 25, 9};

    printf("soma: %i \n", somaVet(numeros, TAMANHO));  // soma: 49
```

```
    return 0;
}
```

10 – Dado o vetor [4, 6, 15, 8, 4, 7, 5], informe a soma dos elementos do vetor. Utilize obrigatoriamente a função criada na questão anterior.

11 – Escreva uma função que receba um vetor de inteiros e retorne o produto dos elementos de um vetor.

12 – Implemente uma função que receba um vetor de inteiros e retorne o produto dos elementos ímpares do vetor.

13 – Desenvolva uma função que receba um vetor de inteiros e retorne a média dos elementos do vetor.

14 – Crie um vetor com 11 elementos. Faça uma função que receba um vetor e retorne o valor de S dado por:

$$S = \frac{0}{vet[0]} + \frac{1}{vet[1]} + \frac{2}{vet[2]} + \ldots + \frac{10}{vet[10]}$$

15 – Crie um vetor com 6 elementos. Implemente uma função que receba um vetor e retorne valor de T dado por:

$$T = \frac{vet[0]^2}{10} \times \frac{vet[1]^2}{11} \times \frac{vet[2]^2}{12} \times \ldots \times \frac{vet[5]^2}{15}$$

16 – Crie um vetor com 5 elementos. Desenvolva uma função que receba um vetor e retorne valor de U dado por:

$$U = 8 \times \frac{5}{vet[0]} \times \frac{7}{vet[1]} \times \frac{9}{vet[2]} \times \frac{11}{vet[3]} \times \frac{13}{vet[4]}$$

---

17 – Implemente uma função que insere um elemento no final do vetor.
Exemplo:
Vetor inicialmente –> [3, 7, 9]
adiciona o número 34
vetor com o novo elemento –> [3, 7, 9, 34]

```
int insereFinalVet(int vet[], int *tamanho, int valor){

    vet[*tamanho] = valor;

    (*tamanho)++;
}

int main()
{
    int i, x, tamanho = 0;
    int numeros[1000];

    while(tamanho < 1000){

        printf("Informe um numero: ");
        scanf("%i", &x);
```

```
        if(x < 0)
            break;

        insereFinalVet(numeros, &tamanho, x);
    }

    // mostra os elementos do vetor
    for(i = 0; i < tamanho; i++)
        printf("%i, ", numeros[i]);

    return 0;
}
```

18 – Desenvolva uma função que receba um vetor e escreva na tela todos os elementos do vetor.

19 – Crie uma função que atualize o valor de um elemento do vetor. A função recebe um vetor, o novo valor do elemento e o índice do elemento.

20 – Faça uma função que remove um elemento de um vetor. A função recebe um vetor e o índice do elemento que será removido.
Exemplo:
Vetor inicialmente –> [3, 7, 9, 10, 35, 79, 155]
remove o elemento de índice 2
vetor após remoção do elemento de índice 2 –> [3, 7, 10, 35, 79, 155]

---

21 – Escreva uma função que verifica se um valor está entre os elementos de um vetor. A função retorna o valor do índice do elemento onde o valor se encontra, e retorna -1 se o valor não está presente no vetor.

```
#define TAMANHO 6

int verificaElementoVet(int vet[], int tamanho, int valor){

    int i;

    for(i = 0; i < tamanho; i++){

        if(valor == vet[i]){

            return i;
        }
    }

    return -1;
}

int main()
{
    int i, x;
    int numeros[TAMANHO] = {1, 3, 6, 4, 8, 9};
```

```
    printf("Valor para busca: ");
    scanf("%i", &x);

    int indice = verificaElementoVet(numeros, TAMANHO, x);

    if (indice == -1)
        printf("%i não está no vetor \n", x);
    else
        printf("%i está na posição %i \n", x, indice);

    return 0;
}
```

22 – Desenvolva uma função que receba um vetor e um valor, e retorne quantas vezes o valor aparece no vetor.

23 – Faça um programa que mostre a quantidade de vezes que cada elemento do vetor ocorre no vetor.
vetor = [7, 3, 2, 5, 2, 7, 3, 7]
Saída:
7 ocorre 3 vezes
3 ocorre 2 vezes
2 ocorre 2 vezes
5 ocorre 1 vez

24 – Escreva um programa que receba um número de até 25 dígitos, onde cada dígito do número deve ser armazenado em uma posição de um vetor. Para encerrar a entrada de dígitos o usuário informará um valor negativo.

25 – Desenvolva uma função que imprima na tela o número informado pelo usuário e armazenado no vetor como na questão anterior, ou seja, imprima os elementos do vetor na ordem inversa. Os elementos vazios do vetor não devem ser impressos na tela.

---

26 – Desenvolva uma função que receba um vetor e retorne 1 se todos os elementos do vetor são ímpares ou retorne 0 em caso contrário.

```
#define TAMANHO 6

int todosImparesVet(int vet[], int tamanho){

    int i;

    for(i = 0; i < tamanho; i++){

        if(vet[i] % 2 == 0){

            return 0;    // 0 --> falso
        }
    }

    return 1; // 1 --> verdadeiro
}
```

```c
int main()
{
    int i;
    int numeros[TAMANHO] = {1, 3, 5, 7, 8, 9};

    if(todosImparesVet(numeros, TAMANHO) == 1){
        printf("Todos os elementos sao impares \n");
    }else{
        printf("Nem todos os elementos sao impares \n");
    }

    return 0;
}
```

27 – Desenvolva uma função que receba um vetor e retorne 1 se todos os elementos do vetor são pares ou retorne 0 em caso contrário.

28 – Escreva uma função que receba um vetor e retorne a quantidade de elementos pares do vetor.

29 – Implemente uma função que recebe um vetor e retorna 1 se os últimos 3 dígitos são maiores que 4, ou retorne 0 em caso contrário.

30 – Crie uma função que verifica se uma senha é válida. A função retorna 1 se a senha é valida e 0 se a senha é inválida. A senha é valida se:

- possui ao menos 4 dígitos;
- todos os elementos são ímpares.

É obrigatória a utilização das funções criadas nas questões anteriores.

31 – Escreva uma função que verifica se uma senha é válida. A função retorna 1 se a senha é valida e 0 se a senha é inválida. A senha é valida se:

- possui ao menos 5 dígitos;
- tem ao menos 2 dígitos pares;
- os 3 últimos dígitos são maiores que 4.

É obrigatória a utilização das funções criadas nas questões anteriores.

32 – Faça uma função que receba um vetor e retorne 1 se todos os elementos do vetor são diferentes ou retorne 0 em caso contrário.

33 – Implemente uma função que converte um número decimal inteiro e positivo em número binário. A função recebe um número inteiro e positivo e um vetor. O vetor recebido armazenará os bits do número binário.

34 – Escreva uma função que receba dois vetores, e retorne 1 caso os dois vetores sejam iguais, ou seja, se os elementos de um vetor, são iguais ao do outro vetor. Caso os vetores sejam diferentes a função deve retornar 0.

35 – Desenvolva uma função que receba três vetores de um mesmo tamanho, os elementos que são interseção dos dois primeiros vetores devem ser postos no terceiro vetor.

36 – Crie uma função que receba três vetores, onde os dois primeiros vetores tem tamanho cinco. Os dois primeiros vetores devem ser unidos no terceiro vetor de forma intercalada. O terceiro vetor tem o tamanho dez.
Exemplo:

vetorA = {1, 2, 3, 4, 5};
vetorB = {51, 52, 53, 54, 55};
vetorC = {1, 51, 2, 52, 3, 53, 4, 54, 5, 55};

---

37 – Faça um programa que simula a mega-sena. Escreva uma função que recebe um vetor com os 6 números do cartão de apostas, sorteia 6 números e retorna o número de acertos do cartão. No programa principal será informado o prêmio do usuário de acordo com a tabela.

Nº de acertos	**Prêmio**
0, 1, 2 ou 3	Sem premiação
4	Quadra
5	Quina
6	Sena

38 – Crie uma função gera uma cartela de bingo com 24 números, onde os números da cartela não se repetem. A função recebe um vetor, onde os números da cartela serão armazenados.

39 – Desenvolva uma função que sorteia N números até que o jogador "bater o bingo"com a cartela cheia. A função recebe a cartela em um vetor e retorna quantas pedras (números) foram chamados até o jogador "bater o bingo".

40 – Escreva uma função que receba três vetores que tem um mesmo tamanho, onde a soma dos dois primeiros vetores deve ser armazenada no terceiro vetor. A soma de dois vetores é a soma dos elementos dos dois vetores que estejam em uma mesma posição,
por exemplo
vetC[0] = vetA[0] + vetB[0]
vetC[1] = vetA[1] + vetB[1]; etc.

41 – Desenvolva uma função que recebe dois vetores, e deve armazenar no segundo vetor, o primeiro vetor invertido.