Programação de Computadores I (ICP131)

Prof. Ronald Souza – IC/UFRJ

Segunda Prova – 27/11/2023

Questão 1) (2.0 pontos) A função abaixo deve realizar uma busca binária de um inteiro (**num**) sobre um vetor de inteiros (**vet**) de tamanho **tam**, já ordenado em ordem **decrescente,** e deve retornar 1 caso o elemento buscado seja encontrado no vetor, e 0 caso contrário. No entanto, a função possui 7 erros, cada um em uma linha distinta. **Cuidado:** nem toda linha está errada!

→ Reescreva a função abaixo **preservando a formatação** e **corrigindo os seus 7 erros.** ***Não acrescente nem remova linhas em relação ao total de linhas do código abaixo.***

```
int busca(int num, int vet[], int tam) {
    int ini = 0, fim = tam, meio;
    while (meio < fim) {
       meio = ini+fim / 2;
       if (num == vet[meio])
             return 0;
       if (vet[meio] < num)
             ini = meio / 2;
       else
             fim = fim * 2;
    }
    return 1;
}
```

Questão 2 (2.0 pontos) Escreva uma **função** que receba 2 matrizes **quadradas**, A e B, e também a dimensão **N** comum a elas. Sua função deverá então retornar o inteiro

- 1, caso as matrizes sejam **simétricas** em relação à **diagonal principal**
- 0, caso contrário.

Por exemplo:

Entrada (A e B, respectivamente, de tamanho N = 3)**:**

3 4 1	**3** 2 4
2 **7** 2	4 **7** 3
4 3 **9**	1 2 **9**

Saída :

1 *//Pois A e B são simétricas!*

Questão 3) (2.0 pontos) O que o código abaixo imprimirá? No seu **caderno de respostas** (e não aqui!), **informe claramente** a alternativa que você escolheu dentre as alternativas a seguir, **e escreva o valor correspondente:**

A) 7124 C) 1742 E) 742
B) 724 D) 427

```
#include <stdio.h>
void sequencia(int x) {
    if (x == 1)
       return;
    if (x % 2 == 0) {
       sequencia(x / 10);
       printf("%d", x % 10);
    }
    else {
       printf("%d", x % 10);
       sequencia(x / 10);
    }
}
int main() {
    sequencia(1274);
    return 0;
}
```

Questão 4) (5.0 pontos) Crie o Tipo Abstrato de Dado (TAD) **Triangulo**. Para isso, siga os seguintes passos:

a) **(0.5 pontos)** Crie o **tipo** estruturado **Triangulo**. Sua estrutura conterá 4 variáveis: 3 **racionais**, **l1**, **l2** e **l3**, referentes à magnitude de cada **lado**, e também uma **string**, referente ao **nome** do triângulo.

b) **(1.5 pontos)** Considere a seguinte definição de **Desigualdade Triangular**: "*Nenhum lado de um triângulo possui magnitude equivalente ou superior à magnitude da soma de seus outros dois lados.*"

Agora, crie uma função que, dados 3 racionais de entrada, ***a***, ***b*** e ***c***, retorne um **Triangulo** de lados `l1` = **a**, `l2` = **b** e `l3` = **c** e **nome** estabelecido como segue:

Se ***a***, ***b*** e ***c*** **não** satisfazem a **desigualdade triangular** (definida acima), o triângulo retornado se chamará `"invalido"`; caso contrário, o nome dependerá da relação entre ***a***, ***b*** e ***c***: `"equilatero"`, caso o triângulo possua 3 lados iguais;

`"isosceles"`, caso **exatamente** 2 lados sejam iguais, ou `"escaleno"` caso seus 3 lados tenham magnitudes distintas.

c) **(1.0 ponto)** Crie uma **função** que, dado um triângulo de entrada, retorne o seu **perímetro** (i.e. a soma de seus lados) ou retorne 0, **caso o triângulo seja inválido**;

d) **(2.0 pontos)** Crie uma **função** que receba um vetor de triângulos e seu tamanho N, e então (i) ordena esse vetor em ordem **crescente** de **perímetro** pelo método **Bubble Sort** e (ii) imprime o **perímetro** e o **nome** de cada triângulo, **seguindo a ordenação crescente estabelecida no item (i)**. Sua função **não** deve retornar valores.
DICA: *Caso queira, é permitido chamar a função de cálculo de perímetro que você criou no item **c).***

Questão 5) (1.0 ponto)
Deseja-se implementar uma função **recursiva** que verifica se um **vetor de inteiros** é um palíndromo i.e. se gera a mesma sequência se lido "de trás pra frente" por ex.: {7,4,4,7}), retornando 1 se verdadeiro e 0 caso contrário.
Em seu caderno de respostas, **indique claramente** a alternativa correta para ser o corpo da função **palindromo()** abaixo:

```
int palindromo(int v[], int ini, int fim) {
    // O que retornar??
}
```

```
A) return (ini >= fim) ? 1 : (v[ini] == v[fim]) && palindromo(v, ini + 1, fim - 1);
B) return (ini == fim) ? 1 : (v[ini] == v[fim]) && palindromo(v, ini + 1, fim - 1);
C) return (ini <= fim) ? 1 : (v[ini] == v[fim]) || palindromo(v, ini + 1, fim - 1);
D) return (ini > fim)  ? 0 : (v[ini] == v[fim]) && palindromo(v, ini + 1, fim - 1);
E) return (ini != fim) ? 0 : (v[ini] == v[fim]) || palindromo(v, ini + 1, fim - 1);
```

```
//Exemplo de chamada:
#include <stdio.h>
#define TAM_v1 5
#define TAM_v2 6
int main() {
    int v1[TAM_v1] = { 5,4,1,4,5 };    //Tamanho ímpar
    int v2[TAM_v2] = { 3,5,4,4,5,3 };  //Tamanho par
    puts("v1 eh palindromo?");
    puts((palindromo(v1, 0, TAM_v1 - 1)) ? "sim" : "nao"); //Imprimirá "sim".
    puts("v2 eh palindromo?");
    puts((palindromo(v2, 0, TAM_v2 - 1)) ? "sim" : "nao"); //Imprimirá "sim".
    return 0;
}
```