

Universidade Federal de Goiás  
Goiânia 12 de Maio de 2018  
Edson Júnior Frota Silva  
Engenharia de Computação  
Estrutura de Dados II

Matrícula: 201515412  
Profº(a): Renata Dutra

## Atividade Pesquisa de Dados parte 2

Questão 01) A função a seguir implementa um algoritmo de busca binária sobre um vetor de inteiros ordenado de modo ascendente.

```
int busca(int vet[], int elem, int ini, int fim) {  
    int m;  
  
    if(fim < ini)  
        return -1;  
  
    m=(ini + fim) / 2;  
  
    System.out.println(vet[m]);  
  
    if(vet[m] == elem)  
        return m;  
  
    if(vet[m] > elem)  
        return busca(vet, elem, ini, m-1);  
  
    return busca(vet, elem, m+1, fim);  
}
```

Essa função recebe como parâmetros um vetor (vet), o elemento que se deseja procurar no vetor (elem), o índice do primeiro elemento do vetor (ini) e o índice do último elemento do vetor (fim). O comando `System.out.println(vet[m])` exibe no console o valor do elemento de índice m do vetor vet. Seja o seguinte vetor (vt) de inteiros:

20	25	27	38	51	57	60	65	73	74	78	80	83	88	90
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

Qual será o 3º valor exibido no console?

**Resposta: c)57**

Questão 02) Considere que na Defensoria há uma lista ordenada com o nome de 1000 cidadãos amazonenses. Utilizando o método de pesquisa binária para localizar o nome de um destes cidadãos, serão necessárias, no máximo,

**Resposta: b) 10 comparações**

**$O(\log n) 2^{10} = 1024$**

Questão 03) O algoritmo de busca e de ordenação que encontra o menor elemento e o troca com a primeira posição, depois o segundo menor com a segunda posição, e assim sucessivamente (n-1 vezes), usa o método de

**Resposta: a) Seleção**

Questão 04) Considere que um algoritmo de pesquisa, em um arquivo previamente ordenado, é caracterizado por realizar comparação de chaves e sucessivas divisões no espaço de busca até encontrar o termo pesquisado ou até haver um único registro. Trata-se de um algoritmo de

**Resposta: b) Pesquisa Binária**

Questão 05) No Programa 1, a busca pelo valor armazenado na variável *elemproc*

```
import javax.swing.JOptionPane;
public class Busca {
    public static void main(String[] args) {
        int vet[] = {9, 21, 34, 43, 45, 60, 66, 88};
        int pri, ult, med, pos, elemproc;
        pos = 0;
        pri = 0;
        ult = 7;
        elemproc = Integer.parseInt(JOptionPane.showInputDialog("Valor?"));
        while (pri <= ult && pos == 0) {
            med = (pri + ult)/2;
            if (vet[med] == elemproc) {
                pos = med + 1;
                ult=-1;
            } else if (vet[med] > elemproc) {
                ult = med - 1;
            } else {
                pri = med + 1;
            }
        }
        if (pos==0){
            System.out.println("O valor procurado não foi encontrado");
        }else{
            System.out.println("O valor procurado é o " + (pos) + "º do vetor");
        }
    }
}
```

**Resposta: d) Usa o método de pesquisa binária, normalmente mais eficiente do que o método de pesquisa linear.**

Questão 06) No Programa 2, para que a busca seja realizada corretamente as lacunas I e II devem ser preenchidas, respectivamente, com

```
import javax.swing.JOptionPane;
public class Busca {
    public static void main(String[] args) {
        int vet[] = {9, 21, 34, 43, 45, 60, 66, 88};
        int pri, ult, med, pos, elemproc;
        pos = 0;
        pri = 0;
        ult = 7;
        elemproc = Integer.parseInt(JOptionPane.showInputDialog("Valor?"));
        pos = busca(elemproc, vet, pri, ult);
        if (pos == -1) {
            System.out.println("O valor procurado não foi encontrado");
        } else {
            System.out.println("O valor procurado é o " + (pos) + "º do vetor");
        }
    }
    public static int busca(int x, int v[], int e, int d) {
        int meio = (e + d) / 2;
        if (v[meio] == x) {
            return meio + 1;
        }
        if (e >= d) {
            return -1;
        } else if (v[meio] < x) {
            return ..I..;
        } else {
            return ..II..;
        }
    }
}
```

**Resposta: b) busca(v, x, meio + 1, d) e busca(v, x, meio - 1, e)**

Questão 07) Avalie se são verdadeiras (V) ou falsas (F) as afirmativas a seguir.

I - O método de busca “pesquisa binária” necessita de um ordenamento prévio do vetor.

II - O método “pesquisa binária” possui o tempo de busca maior que o método “busca sequencial”.

III - O método “busca sequencial” é mais indicado quando se sabe antecipadamente que a maior parte dos registros necessita ser pesquisada.

As afirmativas I, II e III são, respectivamente:

**Resposta: b) V, F e V.**

Questão 08) Os algoritmos de busca e ordenação são bem conhecidos no contexto da computação. Embora muita coisa tenha evoluído na área, alguns algoritmos são clássicos. Nesse sentido, de acordo com o método *algoritmoX* abaixo, é possível afirmar que ele é um algoritmo de:

```
public class Q3 {  
    int algoritmoX (int chave, int vetor[], int limInferior, int  
limSuperior) {  
        int indice = (limInferior + limSuperior)/2;  
        if (vetor[indice] == chave)  
            return indice;  
        if (limInferior >= limSuperior)  
            return -1;  
        else  
            if (vetor[indice] < chave)  
                return algoritmoX(chave, vetor, indice+1,  
limSuperior);  
            else  
                return algoritmoX(chave, vetor,  
limInferior, indice-1);  
    }  
  
    public static void main(String[] args){  
        //...  
    }  
}
```

**Resposta: c) Busca Binária**

Questão 09) Um problema de busca consiste em determinar se um dado objeto é elemento de um vetor. Sobre o algoritmo conhecido como Busca Binária, é CORRETO afirmar:

**Resposta: e) É executado sobre um conjunto de dados previamente ordenado. Realiza sucessivas divisões do espaço de busca comparando o elemento buscado (chave) com o elemento no meio do vetor. Se o elemento do meio do vetor for a chave, a busca termina com sucesso. Caso contrário, se o elemento do meio vier antes do elemento buscado, então a busca continua na metade posterior do vetor. E, finalmente, se o elemento do meio vier depois da chave, a busca continua na metade anterior do vetor.**

Questão 10) Escreva uma função recursiva que receba uma cadeia de caracteres e devolva o inverso da cadeia. Por exemplo, ao receber "algoritmo" a função deve devolver "omtirogla".

```

1  #include<stdio.h>
2  #include<string.h>
3
4  void Inverter_palavra (char *vetor)
5  {
6      if(*vetor)
7      {
8          Inverter_palavra(vetor+1);
9          putchar(*vetor);
10     }
11 }
12 int main()
13 {
14     char palavra[10];
15
16     printf(" Digite a palavra que deseja inverter: ");
17     scanf("%s", palavra);
18
19     printf("\n Palavra invertida: ");
20     Inverter_palavra(palavra);
21
22     printf(" \n ");
23
24     return (0);
25 }

```

Questão 11) Implemente os algoritmos de busca sequencial e binária (pesquisa e inserção) para listas simplesmente encadeadas.

```

1  #include<stdio.h>
2
3  int Pesquisa_Binaria (int numero, int vetor[], int
limite_inferior, int limite_superior)
4  {
5      int meio = (limite_inferior + limite_superior) / 2;
6
7      if (vetor[meio] == numero)
8      {
9          printf("Nenhum numero encontrado!\n");
10         return (0);
11     }
12
13     if (limite_inferior >= limite_superior)
14     {
15         return (-1);
16     }
17
18     else
19     {
20         if (vetor[meio] < numero)
21         {
22             return (Pesquisa_Binaria(numero, vetor, meio+1,
limite_superior));

```

```
23         }
24         else
25         {
26             return (Pesquisa_Binaria(numero, vetor,
limite_inferior, meio-1));
27         }
28     }
29 }
30
31 int main()
32 {
33
34     int vetor[17]=
{0,5,5,8,10,13,16,17,29,31,34,36,43,49,51,11,55,65,32};
35     int numero = 34;
36
37     Pesquisa_Binaria(numero, vetor, 0, 17);
38
39     return(0);
40 }
```