Algoritmos de pesquisa em arrays 1D

Pesquisa

Definição

- A operação de **pesquisa** permite
  - **encontrar** um dado elemento numa dada sequência de elementos, ou
  - concluir que um dado elemento **não existe** numa dada sequência
- A pesquisa de um elemento pode ser feita
  - numa sequência ordenada, ou
  - numa sequência não ordenada

Algoritmos de Pesquisa

Algoritmos

- Quando a sequência não está ordenada, existe apenas um método
  - o método exaustivo
- Quando a sequência está ordenada, existem vários métodos, como
  - pesquisa sequencial
  - pesquisa binária.

Implementação

- Na implementação dos algoritmos que serão estudados, os elementos da sequência estão guardados num array 1D (1 dimensão)

Nota importante

- Daqui para a frente, sempre que omitido, considera-se que
  - a sequência (array 1D) é composta apenas por **elementos** do tipo **inteiro**
  - a sequência (array 1D) está ordenada por **ordem crescente**

Pesquisa Exaustiva

Ideia

- Dado um array 1D **A** com N elementos e o elemento a procurar, o método consiste em percorrer sequencialmente todo o array **A**, desde o primeiro elemento até
  - se encontrar o elemento procurado (concluindo-se que o elemento existe)
  - se analisar todos os elementos do array A (concluindo-se que o elemento não existe)
- O array A pode não estar ordenado

Subprograma em C

Operação: pesquisar o elemento **E** no array 1D **A** com N elementos

```
int pesquisaExaustiva (int E, int A[], int N)
{
  int  k, pos;
  pos = -1;
  k = 0;
  while (k < N && pos == -1) {
     if (A[k] == E)
        pos = k;
     else
        k = k + 1;
  }
  // pos = -1  => E não existe no array A
  // pos >= 0 => E está na posição pos do array A
  return pos;
}
```

Pesquisa Sequencial

Ideia

- Dado um array 1D e o elemento a procurar, o método consiste em percorrer o array desde o primeiro elemento até
  - encontrar o elemento procurado (concluindo-se que existe)
  - encontrar um elemento maior (quando ordenado crescentemente) que o elemento procurado (concluindo-se que não existe)
  - analisar todos os elementos do array (concluindo-se que não existe)

Subprograma em C

Operação: pesquisar o elemento **E** no array 1D **A** com **N** elementos

```
int pesquisaSequencial (int E, int A[], int N)
{
  int  k = 0, pos = -1;    //  pos = posição onde E se encontra em A (-1 = não existe)
  while (k < N && pos == -1) {
     if (A[k] == E)
        pos = k;
     else
        if (A[k] < E)
           k = k + 1;
        else
           pos = -2;
  }
  // pos < 0 => E não existe em A
  // pos ≥ 0 => E existe na posição pos de A
  return pos;
}
```

Pesquisa Binária

Ideia

- Comparar o elemento a pesquisar com o elemento que está ao **meio** do array 1D **A** com **N** elementos e analisar 3 situações diferentes
  - se aquele elemento é **igual** ao do **meio**
  - se aquele elemento é **menor** ao do **meio** (está antes do meio)
  - se aquele elemento é **maior** ao do **meio** (está depois do meio)
- Acontece a 1ª situação => encontrado o elemento na posição **meio** do array A
- Acontece a 2ª situação => basta **pesquisar** o elemento no sub-array até ao meio
- Acontece a 3ª situação => basta **pesquisar** o elemento no sub-array desde o meio

Versão iterativa - subprograma em C

Operação: pesquisar o elemento **E** no array 1D **A** com **N** elementos

```
int pesquisaBinaria (int E, int A[], int N)
{
  int  inicio = 0, fim = N - 1, meio, pos;
  pos = -1;
  while (inicio <= fim && pos == -1) {
     meio = (inicio + fim) / 2;
     if (A[meio] == E)
        pos = meio;              // E está na posição meio do array
     else
        if (A[meio] < E)
           inicio = meio + 1;  // pesquisar E a partir do meio
        else
           fim = meio - 1;     // pesquisar E até ao meio
  }
  return pos;
}
```

Versão iterativa - exemplos

- Exemplo 1

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	**7**	**20**	**25**	**26**	**28**	**30**	**34**	**42**	**44**	**50**	**60**	**68**	**72**	**78**	**81**	**85**	**89**	**92**	**99**

  - Valor a pesquisar: **34**

Versão iterativa - exemplos

- Exemplo 1 (cont)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	**7**	**20**	**25**	**26**	**28**	**30**	**34**	**42**	**44**	**50**	**60**	**68**	**72**	**78**	**81**	**85**	**89**	**92**	**9**

- Valor a pesquisar: **34**

- 1ª tentativa:
  - meio ← (inicio + fim) / 2 = (0 + 19) / 2 = 9

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	**7**	**20**	**25**	**26**	**28**	**30**	**34**	**42**	**44**	**50**	**60**	**68**	**72**	**78**	**81**	**85**	**89**	**92**	**99**
↑									↑										↑
inicio									meio										fim

  - fim ← meio − 1 = 8

Versão iterativa - exemplos

- Exemplo 1 (cont)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	**7**	**20**	**25**	**26**	**28**	**30**	**34**	**42**	**44**	**50**	**60**	**68**	**72**	**78**	**81**	**85**	**89**	**92**	**99**

- Valor a pesquisar: **34**

- 2ª tentativa:
  - meio ← (inicio + fim) / 2 = (0 + 8) / 2 = 4

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	**7**	**20**	**25**	**26**	**28**	**30**	**34**	**42**	**44**	**50**	**60**	**68**	**72**	**78**	**81**	**85**	**89**	**92**	**99**
↑				↑				↑											
inicio				meio				fim											

  - inicio ← meio + 1 = 5

Versão iterativa - exemplos

- Exemplo 1 (cont)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	**7**	**20**	**25**	**26**	**28**	**30**	**34**	**42**	**44**	**50**	**60**	**68**	**72**	**78**	**81**	**85**	**89**	**92**	**99**

- Valor a pesquisar: **34**

- 3ª tentativa:
  - meio ← (inicio + fim) / 2 = (5 + 8) / 2 = 6

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	**7**	**20**	**25**	**26**	**28**	**30**	**34**	**42**	**44**	**50**	**60**	**68**	**72**	**78**	**81**	**85**	**89**	**92**	**99**
					↑	↑		↑											
					inicio	meio		fim											

  - inicio ← meio + 1 = 7

Versão iterativa - exemplos

- Exemplo 1 (cont)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	**7**	**20**	**25**	**26**	**28**	**30**	**34**	**42**	**44**	**50**	**60**	**68**	**72**	**78**	**81**	**85**	**89**	**92**	**99**

- Valor a pesquisar: **34**

- 4ª tentativa:
  - meio ← (inicio + fim) / 2 = (7 + 8) / 2 = 7

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	**7**	**20**	**25**	**26**	**28**	**30**	**34**	**42**	**44**	**50**	**60**	**68**	**72**	**78**	**81**	**85**	**89**	**92**	**99**
							↑	↑											
							inicio	fim											
							meio												

Versão iterativa - exemplos

- Exemplo 1 (cont)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	**7**	**20**	**25**	**26**	**28**	**30**	**34**	**42**	**44**	**50**	**60**	**68**	**72**	**78**	**81**	**85**	**89**	**92**	**99**

- Valor a pesquisar: **34**

- 4ª tentativa:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	**7**	**20**	**25**	**26**	**28**	**30**	**34**	**42**	**44**	**50**	**60**	**68**	**72**	**78**	**81**	**85**	**89**	**92**	**99**
							↑	↑											
							inicio	fim											
							meio												

O elemento 34 foi encontrado após 4 tentativas

Versão iterativa - exemplos

- Exemplo 2

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	**7**	**20**	**25**	**26**	**28**	**30**	**34**	**42**	**44**	**50**	**60**	**68**	**72**	**78**	**81**	**85**	**89**	**92**	**99**

  - Valor a pesquisar: **40**

Versão iterativa - exemplos

- Exemplo 2 (cont)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	**7**	**20**	**25**	**26**	**28**	**30**	**34**	**42**	**44**	**50**	**60**	**68**	**72**	**78**	**81**	**85**	**89**	**92**	**99**

- Valor a pesquisar: **40**

- 1ª tentativa:
  - meio ← (inicio + fim) / 2 = (0 + 19) / 2 = 9

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	**7**	**20**	**25**	**26**	**28**	**30**	**34**	**42**	**44**	**50**	**60**	**68**	**72**	**78**	**81**	**85**	**89**	**92**	**99**
↑									↑										↑
inicio									meio										fim

  - inicio ← meio - 1 = 8

Versão iterativa - exemplos

- Exemplo 2 (cont)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	**7**	**20**	**25**	**26**	**28**	**30**	**34**	**42**	**44**	**50**	**60**	**68**	**72**	**78**	**81**	**85**	**89**	**92**	**99**

- Valor a pesquisar: **40**

- 2ª tentativa:
  - meio ← (inicio + fim) / 2 = (0 + 8) / 2 = 4

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	**7**	**20**	**25**	**26**	**28**	**30**	**34**	**42**	**44**	**50**	**60**	**68**	**72**	**78**	**81**	**85**	**89**	**92**	**99**
↑				↑				↑											
inicio				meio				fim											

  - inicio ← meio + 1 = 5

Versão iterativa - exemplos

- Exemplo 2 (cont)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	**7**	**20**	**25**	**26**	**28**	**30**	**34**	**42**	**44**	**50**	**60**	**68**	**72**	**78**	**81**	**85**	**89**	**92**	**99**

- Valor a pesquisar: **40**

- 3ª tentativa:
  - meio ← (inicio + fim) / 2 = (5 + 8) / 2 = 6

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	**7**	**20**	**25**	**26**	**28**	**30**	**34**	**42**	**44**	**50**	**60**	**68**	**72**	**78**	**81**	**85**	**89**	**92**	**99**
					↑	↑		↑											
					inicio	meio		fim											

  - inicio ← meio + 1 = 7

Versão iterativa - exemplos

- Exemplo 2 (cont)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	**7**	**20**	**25**	**26**	**28**	**30**	**34**	**42**	**44**	**50**	**60**	**68**	**72**	**78**	**81**	**85**	**89**	**92**	**99**

- Valor a pesquisar: **40**

- 4ª tentativa:
  - meio ← (inicio + fim) / 2 = (7 + 8) / 2 = 7

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	**7**	**20**	**25**	**26**	**28**	**30**	**34**	**42**	**44**	**50**	**60**	**68**	**72**	**78**	**81**	**85**	**89**	**92**	**99**
							↑	↑											
							inicio	fim											
							meio												

  - inicio ← meio + 1 = 8

Versão iterativa - exemplos

- Exemplo 2 (cont)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	**7**	**20**	**25**	**26**	**28**	**30**	**34**	**42**	**44**	**50**	**60**	**68**	**72**	**78**	**81**	**85**	**89**	**92**	**99**

- Valor a pesquisar: **40**

- 5ª tentativa:
  - meio ← (inicio + fim) / 2 = (8 + 8) / 2 = 8

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	**7**	**20**	**25**	**26**	**28**	**30**	**34**	**42**	**44**	**50**	**60**	**68**	**72**	**78**	**81**	**85**	**89**	**92**	**99**

↑

inicio / fim

meio

  - fim ← meio - 1 = 7

Versão iterativa - exemplos

- Exemplo 2 (cont)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	**7**	**20**	**25**	**26**	**28**	**30**	**34**	**42**	**44**	**50**	**60**	**68**	**72**	**78**	**81**	**85**	**89**	**92**	**99**

- Valor a pesquisar: **40**

- 6ª tentativa:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	**7**	**20**	**25**	**26**	**28**	**30**	**34**	**42**	**44**	**50**	**60**	**68**	**72**	**78**	**81**	**85**	**89**	**92**	**99**
							↑	↑											
							fim	inicio											

- o elemento **40 não está** no array 1D (inicio > fim)
- o que se concluiu após **6 tentativas**

Versão recursiva – subprograma em C

Operação: pesquisar o elemento **E** no array 1D **A** entre as posições inicio e fim

```
int pesquisaBinariaR (int E, int A[], int inicio, int fim)
{
  int  meio;
  // casos base/terminais
  if (inicio > fim)
     return -1;     //  E não está em A
  meio = (inicio + fim) / 2;
  if (A[meio] == E)
     return meio;   //  E está na posição meio
  // caso geral
  if (A[meio] < E)
     return pesquisaBinariaR(E, A, meio+1, fim);
  else
     return pesquisaBinariaR(E, A, inicio, meio-1);
}
```