

CI1055: Algoritmos e Estruturas de Dados I

Profs. Drs. Marcos Castilho, Bruno Müller Jr, Carmem Hara

Departamento de Informática/UFPR

27 de outubro de 2021

Resumo

Maior segmento crescente

- Apresentar uma interessante combinação de várias técnicas para resolver um problema nada trivial.

- Os problemas agora se tornam bem complexos
- Exigem combinações das técnicas elementares
- Exige bastante raciocínio
- Usar técnica de resolver subproblemas ajuda!

Maior segmento crescente

Problema: Dada uma sequência de n números naturais, imprimir o valor do comprimento do segmento crescente de tamanho máximo dentre os números lidos. A sequência é terminada em zero, que não deve ser processado.

Exemplo 1:

5, 10, 3, 2, 4, 7, 9, 8, 5, ...

Resposta: 4, para o segmento 2, 4, 7, 9.

Exemplo 2:

10, 8, 7, 5, 2, ...

Resposta: 1, todos os segmentos crescentes são unitários.

Quais são os subproblemas

Problema: Dada uma sequência de n números naturais, imprimir o valor do comprimento do segmento crescente de tamanho máximo dentre os números lidos. A sequência é terminada em zero, que não deve ser processado.

- ler os números da entrada;
- para cada número, descobrir se estamos em um segmento crescente ou não;
- memorizar qual é o tamanho do segmento crescente “da vez”;
- saber qual é o tamanho do segmento crescente de tamanho máximo.
- podem haver outros...

A sequência é terminada em zero, que não deve ser processado.

```
begin
  read (n);
  while n <> 0 do
    begin
      (* faz algo *)
      read (n);
    end;
  end.
end.
```

Saber se uma subsequência é crescente

É preciso lembrar do número anterior.

Rascunho1

```
begin
  read (n);
  while n <> 0 do
  begin
    (* faz algo *)
    read (n);
  end;
end.
```

Rascunho 2

```
begin
  read (n);
  while n <> 0 do
  begin
    n_anterior:= n;
    read (n);
    if n > n_anterior then
      (* seq crescente *)
    else
      (* acabou sequencia *)
    end;
  end;
end.
```

A sequência pode ter tamanho 1. O primeiro pode ser nulo.

Rascunho2

```
begin
  read (n);
  while n <> 0 do
    begin
      n_anterior:= n; (* <--- *)
      read (n);
      if n > n_anterior then
        (* seq crescente *)
      else
        (* acabou sequencia *)
      end;
    end.
end.
```

Rascunho 3

```
begin
  read (n);
  n_anterior:= n;
  if n <> 0 then
    begin
      read (n);
      while n <> 0 do
        begin
          if n > n_anterior then
            (* seq crescente *)
          else
            (* acabou sequencia *)
          n_anterior:= n;
          read (n);
        end;
      end;
    end.
end.
```

Memorizar o tamanho do segmento crescente atual

- Tem que introduzir um contador

Memorizar o tamanho do segmento crescente atual

Introduzindo um contador

Rascunho3

```
begin
  read (n);
  n_anterior:= n;
  if n <> 0 then
  begin
    read (n);
    while n <> 0 do
    begin
      if n > n_anterior then
        (* seq crescente *)
      else
        (* acabou sequencia *)
        n_anterior:= n;
        read (n);
      end;
    end;
  end.
```

Rascunho 4

```
begin
  read (n);
  n_anterior:= n;
  tam:= 0;
  if n <> 0 then
  begin
    read (n);
    tam:= tam + 1;
    while n <> 0 do
    begin
      if n > n_anterior then
        tam:= tam + 1
      else
        tam:= 1;
        n_anterior:= n;
        read (n);
      end;
    end;
  end.
```

Qual é o tamanho do maior segmento?

- Usar a técnica de definir a priori e depois corrigir!

Definir a priori e depois corrigir

Rascunho 5

Rascunho4

```
begin
  read (n);
  n_anterior:= n;
  tam:= 0;
  if n <> 0 then
    begin
      read (n);
      tam:= tam + 1;
      while n <> 0 do
        begin
          if n > n_anterior then
            tam:= tam + 1
          else
            tam:= 1;
            n_anterior:= n;
            read (n);
          end;
        end;
      end;
    end.
end.
```

```
begin
  read (n);
  n_anterior:= n;
  tam:= 0;
  maior_tam:= tam;
  if n <> 0 then
    begin
      read (n);
      tam:= tam + 1;
      while n <> 0 do
        begin
          if n > n_anterior then
            tam:= tam + 1
          else
            begin
              if tam > maior_tam then
                maior_tam:= tam;
              tam:= 1;
            end;
            n_anterior:= n;
            read (n);
          end;
        end;
      end;
    end;
  writeln (maior_tam);
end.
```

- Se o maior segmento crescente for o último
- Neste caso, o tamanho deste não foi comparado!

Versão final

```
begin
  read (n);
  n_anterior:= n;
  tam:= 0;
  maior_tam:= tam;
  if n <> 0 then
    begin
      read (n);
      tam:= tam + 1;
      while n <> 0 do
        begin
          if n > n_anterior then
            tam:= tam + 1
          else
            begin
              if tam > maior_tam then
                maior_tam:= tam;
              tam:= 1;
            end;
          n_anterior:= n;
          read (n);
        end;
      if tam > maior_tam then
        maior_tam:= tam;
      writeln (maior_tam);
    end;
  end.
```

- Este não é um problema trivial!
- Pudemos resolvê-lo pela divisão em subproblemas
- Cada subproblema pode ser resolvido com mais foco e corretude

- este material está no livro no capítulo 7, seção 7.9

- Slides feitos em \LaTeX usando beamer
- Licença

Creative Commons Atribuição-Uso Não-Comercial-Vedada a Criação de Obras Derivadas 2.5 Brasil License.<http://creativecommons.org/licenses/by-nc-nd/2.5/br/>

Creative Commons Atribuição-Uso Não-Comercial-Vedada a Criação de Obras Derivadas 2.5 Brasil License.<http://creativecommons.org/licenses/by-nc-nd/2.5/br/>