CI1055: Algoritmos e Estruturas de Dados I

Profs. Drs. Marcos Castilho e Bruno Muller Jr

Departamento de Informática/UFPR

22 de julho de 2020

Técnicas elementares de programação

- Até o momento, os problemas apresentados geraram programas simples e pequenos;
- Programas maiores requerem técnicas que:
  - minimizam erros de lógica
  - agilizem o desenvolvimento
- Técnicas elementares de programação:
  - Lógica de programação
  - Teste de mesa
  - Técnica dos acumuladores
  - Árvore de Decisão

Lógica de programação

Problema: Contar a quantidade de números digitados pelo usuário no teclado.

- questões para implementar a solução:
  - quando sabe que terminou de digitar?
  - quando for digitado zero: 5 8 3 9 0
  - números seguidos de <enter>

Lógica de programação

Problema: Contar a quantidade de números digitados pelo usuário no teclado.

- questões para implementar a solução:
  - quando sabe que terminou de digitar?
  - quando for digitado zero: 5 8 3 9 0
  - números seguidos de <enter>

Lógica de programação

Problema: Contar a quantidade de números digitados pelo usuário no teclado.

- questões para implementar a solução:
  - quando sabe que terminou de digitar?
  - quando for digitado zero: 5 8 3 9 0
  - números seguidos de <enter>

Lógica de programação

Problema: Contar a quantidade de números digitados pelo usuário no teclado.

- questões para implementar a solução:
  - quando sabe que terminou de digitar?
  - quando for digitado zero: 5 8 3 9 0
  - números seguidos de <enter>

Lógica de programação

- Como construir a lógica (algoritmo) que pode ser formalizada em um programa?
- Pensando como o computador faria:
  1. aguarda digitação de um número;
  2. "trata" número digitado;
  3. aguarda digitação do próximo número;
- Mapear a lógica acima para os comandos conhecidos.
- Quais, em que ordem (quebra-cabeça).
- O que colocar em "trata" número digitado?

Lógica de programação

- Como construir a lógica (algoritmo) que pode ser formalizada em um programa?
- Pensando como o computador faria:
  1. aguarda digitação de um número;
  2. "trata" número digitado;
  3. aguarda digitação do próximo número;
- Mapear a lógica acima para os comandos conhecidos.
- Quais, em que ordem (quebra-cabeça).
- O que colocar em "trata" número digitado?

Lógica de programação

- Como construir a lógica (algoritmo) que pode ser formalizada em um programa?
- Pensando como o computador faria:
  1. aguarda digitação de um número;
  2. "trata" número digitado;
  3. aguarda digitação do próximo número;
- Mapear a lógica acima para os comandos conhecidos.
- Quais, em que ordem (quebra-cabeça).
- O que colocar em "trata" número digitado?

Lógica de programação

- Como construir a lógica (algoritmo) que pode ser formalizada em um programa?
- Pensando como o computador faria:
  1. aguarda digitação de um número;
  2. "trata" número digitado;
  3. aguarda digitação do próximo número;
- Mapear a lógica acima para os comandos conhecidos.
- Quais, em que ordem (quebra-cabeça).
- O que colocar em "trata" número digitado?

Lógica de programação

- Como construir a lógica (algoritmo) que pode ser formalizada em um programa?
- Pensando como o computador faria:
  1. aguarda digitação de um número;
  2. "trata" número digitado;
  3. aguarda digitação do próximo número;
- Mapear a lógica acima para os comandos conhecidos.
- Quais, em que ordem (quebra-cabeça).
- O que colocar em "trata" número digitado?

Lógica de programação

- "trata" número digitado:
  1. se digitou zero, imprime a quantidade de números digitados;
  2. como contar? Utiliza uma variável (iniciar, somar, imprimir);
  3. Solução:

```
program contar_numeros_v1 ;
var cont , n: integer ;
begin
    cont:= 1;
    read (n) ;
    while n <> 0 do
    begin
        read (n) ;
        cont:= cont + 1;
    end;
    writeln (cont - 1) ;
end.
```

Lógica de programação

- "trata" número digitado:
  1. se digitou zero, imprime a quantidade de números digitados;
  2. como contar? Utiliza uma variável (iniciar, somar, imprimir);
  3. Solução:

```
program contar_numeros_v1 ;
var cont , n: integer ;
begin
    cont:= 1;
    read (n) ;
    while n <> 0 do
    begin
        read (n) ;
        cont:= cont + 1;
    end;
    writeln (cont - 1) ;
end.
```

Lógica de programação

- "trata" número digitado:
  1. se digitou zero, imprime a quantidade de números digitados;
  2. como contar? Utiliza uma variável (iniciar, somar, imprimir);
  3. Solução:

```
program contar_numeros_v1 ;
var cont , n: integer ;
begin
    cont:= 1;
    read (n) ;
    while n <> 0 do
    begin
        read (n) ;
        cont:= cont + 1;
    end;
    writeln (cont - 1) ;
end.
```

Lógica de programação

- "trata" número digitado:
  1. se digitou zero, imprime a quantidade de números digitados;
  2. como contar? Utiliza uma variável (iniciar, somar, imprimir);
  3. Solução:

```
program contar_numeros_v1 ;
var cont , n: integer ;
begin
    cont:= 1;
    read (n) ;
    while n <> 0 do
    begin
        read (n) ;
        cont:= cont + 1;
    end;
    writeln (cont - 1) ;
end.
```

Lógica de programação

- "trata" número digitado:
  1. se digitou zero, imprime a quantidade de números digitados;
  2. como contar? Utiliza uma variável (iniciar, somar, imprimir);
  3. Solução:

```
program contar_numeros_v1 ;
var cont , n: integer ;
begin
    cont:= 1;
    read (n) ;
    while n <> 0 do
    begin
        read (n) ;
        cont:= cont + 1;
    end;
    writeln (cont - 1) ;
end.
```

Lógica de programação

- Versão alternativa:

```
program contar_numeros_v2 ;
var cont, n: integer ;
begin
    cont:=0;
    read (n);
    while n <> 0 do
    begin
        cont:= cont + 1;
        read (n);
    end;
    writeln (cont);
end.
```

Teste de Mesa

- técnica para testar o funcionamento de um programa;
  - acompanhar o programa segundo o fluxo das instruções;
  - anotar em um papel, que supostamente está em uma mesa, os valores das variáveis;
  - durante o processo, permite observar se o programa contém algum erro de lógica;

Teste de Mesa

- técnica para testar o funcionamento de um programa;
  - acompanhar o programa segundo o fluxo das instruções;
  - anotar em um papel, que supostamente está em uma mesa, os valores das variáveis;
  - durante o processo, permite observar se o programa contém algum erro de lógica;

Teste de Mesa

- técnica para testar o funcionamento de um programa;
  - acompanhar o programa segundo o fluxo das instruções;
  - anotar em um papel, que supostamente está em uma mesa, os valores das variáveis;
  - durante o processo, permite observar se o programa contém algum erro de lógica;

Teste de Mesa

- técnica para testar o funcionamento de um programa;
  - acompanhar o programa segundo o fluxo das instruções;
  - anotar em um papel, que supostamente está em uma mesa, os valores das variáveis;
  - durante o processo, permite observar se o programa contém algum erro de lógica;

Aplicação do Teste de Mesa

```
program contar_numeros_v2 ;
var cont, n: integer ;
begin
   cont:=0;
   read (n);
   while n <> 0 do
   begin
      cont:= cont + 1;
      read (n);
   end;
   writeln (cont);
end.
```

Aplicação do Teste de Mesa [5 8 3 0]

```
program contar_numeros_v2;
var cont, n: integer;
begin
   cont:=0;
   read (n);
   while n <> 0 do
   begin
      cont:= cont + 1;
      read (n);
   end;
   writeln (cont);
end.
```

Aplicação do Teste de Mesa [5 8 3 0]

```
program contar_numeros_v2;
var cont, n: integer;
begin
   cont:=0;
   read (n);
   while n <> 0 do
   begin
      cont:= cont + 1;
      read (n);
   end;
   writeln (cont);
end.
```

cont	n
?	?

Aplicação do Teste de Mesa [5 8 3 0]

```
program contar_numeros_v2;
var cont, n: integer;
begin
   cont:=0;
   read (n);
   while n <> 0 do
   begin
      cont:= cont + 1;
      read (n);
   end;
   writeln (cont);
end.
```

cont	n
?	?

Aplicação do Teste de Mesa [5 8 3 0]

```
program contar_numeros_v2;
var cont, n: integer;
begin
   cont:=0;
   read (n);
   while n <> 0 do
   begin
      cont:= cont + 1;
      read (n);
   end;
   writeln (cont);
end.
```

cont	n
0	?

Aplicação do Teste de Mesa [5 8 3 0]

```
program contar_numeros_v2;
var cont, n: integer;
begin
   cont:=0;
   read (n);
   while n <> 0 do
   begin
      cont:= cont + 1;
      read (n);
   end;
   writeln (cont);
end.
```

cont	n
0	5

Aplicação do Teste de Mesa [5 8 3 0]

```
program contar_numeros_v2;
var cont, n: integer;
begin
   cont:=0;
   read (n);
   while n <> 0 do
   begin
      cont:= cont + 1;
      read (n);
   end;
   writeln (cont);
end.
```

cont	n
0	5

Aplicação do Teste de Mesa [5 8 3 0]

```
program contar_numeros_v2;
var cont, n: integer;
begin
   cont:=0;
   read (n);
   while n <> 0 do
   begin
      cont:= cont + 1;
      read (n);
   end;
   writeln (cont);
end.
```

cont	n
0	5

Aplicação do Teste de Mesa [5 8 3 0]

```
program contar_numeros_v2;
var cont, n: integer;
begin
   cont:=0;
   read (n);
   while n <> 0 do
   begin
      cont:= cont + 1;
      read (n);
   end;
   writeln (cont);
end.
```

cont	n
1	5

Aplicação do Teste de Mesa [5 8 3 0]

```
program contar_numeros_v2;
var cont, n: integer;
begin
   cont:=0;
   read (n);
   while n <> 0 do
   begin
      cont:= cont + 1;
      read (n);
   end;
   writeln (cont);
end.
```

cont	n
1	8

Aplicação do Teste de Mesa [5 8 3 0]

```
program contar_numeros_v2;
var cont, n: integer;
begin
   cont:=0;
   read (n);
   while n <> 0 do
   begin
      cont:= cont + 1;
      read (n);
   end;
   writeln (cont);
end.
```

cont	n
1	8

Aplicação do Teste de Mesa [5 8 3 0]

```
program contar_numeros_v2;
var cont, n: integer;
begin
   cont:=0;
   read (n);
   while n <> 0 do
   begin
      cont:= cont + 1;
      read (n);
   end;
   writeln (cont);
end.
```

cont	n
1	8

Aplicação do Teste de Mesa [5 8 3 0]

```
program contar_numeros_v2;
var cont, n: integer;
begin
   cont:=0;
   read (n);
   while n <> 0 do
   begin
      cont:= cont + 1;
      read (n);
   end;
   writeln (cont);
end.
```

cont	n
2	8

Aplicação do Teste de Mesa [5 8 3 0]

```
program contar_numeros_v2;
var cont, n: integer;
begin
   cont:=0;
   read (n);
   while n <> 0 do
   begin
      cont:= cont + 1;
      read (n);
   end;
   writeln (cont);
end.
```

cont	n
2	3

Aplicação do Teste de Mesa [5 8 3 0]

```
program contar_numeros_v2;
var cont, n: integer;
begin
   cont:=0;
   read (n);
   while n <> 0 do
   begin
      cont:= cont + 1;
      read (n);
   end;
   writeln (cont);
end.
```

cont	n
2	3

Aplicação do Teste de Mesa [5 8 3 0]

```
program contar_numeros_v2;
var cont, n: integer;
begin
   cont:=0;
   read (n);
   while n <> 0 do
   begin
      cont:= cont + 1;
      read (n);
   end;
   writeln (cont);
end.
```

cont	n
2	3

Aplicação do Teste de Mesa [5 8 3 0]

```
program contar_numeros_v2;
var cont, n: integer;
begin
   cont:=0;
   read (n);
   while n <> 0 do
   begin
      cont:= cont + 1;
      read (n);
   end;
   writeln (cont);
end.
```

cont	n
3	3

Aplicação do Teste de Mesa [5 8 3 0]

```
program contar_numeros_v2;
var cont, n: integer;
begin
   cont:=0;
   read (n);
   while n <> 0 do
   begin
      cont:= cont + 1;
      read (n);
   end;
   writeln (cont);
end.
```

cont	n
3	0

Aplicação do Teste de Mesa [5 8 3 0]

```
program contar_numeros_v2;
var cont, n: integer;
begin
   cont:=0;
   read (n);
   while n <> 0 do
   begin
      cont:= cont + 1;
      read (n);
   end;
   writeln (cont);
end.
```

cont	n
3	0

Aplicação do Teste de Mesa [5 8 3 0]

```
program contar_numeros_v2;
var cont, n: integer;
begin
   cont:=0;
   read (n);
   while n <> 0 do
   begin
      cont:= cont + 1;
      read (n);
   end;
   writeln (cont);
end.
```

```
cont          n
  3           0
  |
  +-----------------> 3
```

Técnica do Acumulador

- técnica desenvolvimento de um programa que utiliza modelos já desenvolvidos (preferencialmente já testados);
- separa variáveis da repetição em dois grupos (preferencialmente disjuntos):
  - variáveis de controle;
  - variáveis de execução, ou acumuladores;
- "copiar" ou "adaptar" variáveis de controle do while;
- modificar os acumuladores;

Técnica do Acumulador

- técnica desenvolvimento de um programa que utiliza modelos já desenvolvidos (preferencialmente já testados);
- separa variáveis da repetição em dois grupos (preferencialmente disjuntos):
  - variáveis de controle;
  - variáveis de execução, ou acumuladores;
- "copiar" ou "adaptar" variáveis de controle do while;
- modificar os acumuladores;

Técnica do Acumulador

- técnica desenvolvimento de um programa que utiliza modelos já desenvolvidos (preferencialmente já testados);
- separa variáveis da repetição em dois grupos (preferencialmente disjuntos):
  - variáveis de controle;
  - variáveis de execução, ou acumuladores;
- "copiar" ou "adaptar" variáveis de controle do while;
- modificar os acumuladores;

Técnica do Acumulador

- técnica desenvolvimento de um programa que utiliza modelos já desenvolvidos (preferencialmente já testados);
- separa variáveis da repetição em dois grupos (preferencialmente disjuntos):
  - variáveis de controle;
  - variáveis de execução, ou acumuladores;
- "copiar" ou "adaptar" variáveis de controle do while;
- modificar os acumuladores;

Técnica do Acumulador

- técnica desenvolvimento de um programa que utiliza modelos já desenvolvidos (preferencialmente já testados);
- separa variáveis da repetição em dois grupos (preferencialmente disjuntos):
  - variáveis de controle;
  - variáveis de execução, ou acumuladores;
- "copiar" ou "adaptar" variáveis de controle do while;
- modificar os acumuladores;

Técnica do Acumulador

- técnica desenvolvimento de um programa que utiliza modelos já desenvolvidos (preferencialmente já testados);
- separa variáveis da repetição em dois grupos (preferencialmente disjuntos):
  - variáveis de controle;
  - variáveis de execução, ou acumuladores;
- "copiar" ou "adaptar" variáveis de controle do while;
- modificar os acumuladores;

Exemplo

Problema: Ler uma sequência de números do teclado e imprimir a quantidade de números digitados. O programa deve terminar quando o número lido do teclado for zero

Problema: Ler uma sequência de números do teclado e imprimir a soma dos números positivos digitados. O programa deve terminar quando o número lido do teclado for zero

Exemplo

Problema: Ler uma sequência de números do teclado e imprimir a quantidade de números digitados. O programa deve terminar quando o número lido do teclado for zero

```
program contar_numeros_v2 ;
var cont, n: integer ;
begin
   cont:=0;
   read (n);
   while n <> 0 do
   begin
     cont:= cont + 1;
      read (n);
   end;
   writeln (cont);
end.
```

Problema: Ler uma sequência de números do teclado e imprimir a soma dos números positivos digitados. O programa deve terminar quando o número lido do teclado for zero

Exemplo

Problema: Ler uma sequência de números do teclado e imprimir a quantidade de números digitados. O programa deve terminar quando o número lido do teclado for zero

```
program contar_numeros_v2 ;
var cont, n: integer ;
begin
   cont:=0; (* acumulador *)
   read (n);
   while n <> 0 do
   begin
     cont:= cont + 1;(* acumulador *)
      read (n);
   end;
   writeln (cont);
end.
```

Problema: Ler uma sequência de números do teclado e imprimir a soma dos números positivos digitados. O programa deve terminar quando o número lido do teclado for zero

```
program somar_numeros ;
var soma, n: integer ;
begin

   read (n);
   while n <> 0 do
   begin

      read (n);
   end;
   writeln (soma);
end.
```

Exemplo

Problema: Ler uma sequência de números do teclado e imprimir a quantidade de números digitados. O programa deve terminar quando o número lido do teclado for zero

```
program contar_numeros_v2 ;
var cont, n: integer ;
begin
   cont:=0; (* acumulador *)
   read (n);
   while n <> 0 do
   begin
     cont:= cont + 1;(* acumulador *)
      read (n);
   end;
   writeln (cont);
end.
```

Problema: Ler uma sequência de números do teclado e imprimir a soma dos números positivos digitados. O programa deve terminar quando o número lido do teclado for zero

```
program somar_numeros ;
var soma, n: integer ;
begin
   soma:=0; (* acumulador *)
   read (n);
   while n <> 0 do
   begin
     soma:= soma + n;(* acumulador *)
      read (n);
   end;
   writeln (soma);
end.
```

Exemplo

Problema: Ler uma sequência de números do teclado e imprimir a quantidade de números digitados. O programa deve terminar quando o número lido do teclado for zero

```
program contar_numeros_v2 ;
var cont, n: integer ;
begin
   cont:=0; (* acumulador *)
   read (n);
   while n <> 0 do
   begin
     cont:= cont + 1;(* acumulador *)
      read (n);
   end;
   writeln (cont);
end.
```

Problema: Ler uma sequência de números do teclado e imprimir a soma dos números positivos digitados. O programa deve terminar quando o número lido do teclado for zero

```
program somar_numeros ;
var soma, n: integer ;
begin
   soma:=0; (* acumulador *)
   read (n);
   while n <> 0 do
   begin
      if (n>0) then
         soma:= soma + n;(* acumulador *)
      read (n);
   end;
   writeln (soma);
end.
```

Exercício

Problema: Ler um número N $>$ 0 do teclado e em seguida ler N números inteiros quaisquer. Ao final imprimir a soma deles.

Exercício

Problema: Ler um número N $>$ 0 do teclado e em seguida ler N números inteiros quaisquer. Ao final imprimir a soma deles.

```
program soma_valores ;
var n, numero, i, soma : integer;
begin
    read (n) ;
    soma:= 0; (* primeiro acumulador, para somar os numeros lidos *)
    i:= 1;    (* segundo acumulador, para contar os numeros lidos *)
    while i <= n do
    begin
        read (numero) ;
        soma:= soma + numero; (* atualizacao do primeiro acumulador *)
        i:= i + 1;            (* atualizacao do segundo acumulador  *)
    end;
    writeln (soma) ;
end.
```

Árvore de Decisão

Problema: Após ler três números no teclado, imprimir o menor deles.

Árvore de Decisão

Problema: Após ler três números no teclado, imprimir o menor deles.

```
program imprime menor;
var a, b, c : integer ;
begin
    write('entre com tres numeros inteiros: ') ;
    read(a , b, c) ;
    if a < b then
        if a < c then
            writeln ('o menor dos tres eh ' ,a)
        else
        writeln ('o menor dos tres eh ' , c)
    else (* entra neste else quando a >= b *)
        if b < c then
            writeln ('o menor dos tres eh ' ,b)
        else
            writeln ('o menor dos tres eh ' , c) ;
end.
```

Árvore de Decisão

Problema: Após ler três números no teclado, imprimir o menor deles.

```
program imprime menor;
var a, b, c : integer ;
begin
    write('entre com tres numeros inteiros: ') ;
    read(a , b, c) ;
    if a < b then
        if a < c then
            writeln ('o menor dos tres eh ' ,a)
        else
        writeln ('o menor dos tres eh ' , c)
    else (* entra neste else quando a >= b *)
        if b < c then
            writeln ('o menor dos tres eh ' ,b)
        else
            writeln ('o menor dos tres eh ' , c) ;
end.
```

- "Aninhamento de ifs"
- Árvore, que Árvore???

Árvore de Decisão

Problema: Após ler três números no teclado, imprimir o menor deles.

```
program imprime menor;
var a, b, c : integer ;
begin
    write('entre com tres numeros inteiros: ') ;
    read(a , b, c) ;
    if a < b then
        if a < c then
            writeln ('o menor dos tres eh ' ,a)
        else
        writeln ('o menor dos tres eh ' , c)
    else (* entra neste else quando a >= b *)
        if b < c then
            writeln ('o menor dos tres eh ' ,b)
        else
            writeln ('o menor dos tres eh ' , c) ;
end.
```

- "Aninhamento de ifs"
- Árvore, que Árvore???

Árvore de Decisão

Problema: Após ler três números no teclado, imprimir o menor deles.

```
program imprime menor;
var a, b, c : integer ;
begin
    write('entre com tres numeros inteiros: ') ;
    read(a , b, c) ;
    if a < b then
        if a < c then
            writeln ('o menor dos tres eh ' ,a)
        else
        writeln ('o menor dos tres eh ' , c)
    else (* entra neste else quando a >= b *)
        if b < c then
            writeln ('o menor dos tres eh ' ,b)
        else
            writeln ('o menor dos tres eh ' , c) ;
end.
```

- "Aninhamento de ifs"
- Árvore, que Árvore???

Fim

- este material está no livro no capítulo 6, seções de 6.1 até 6.4