UNIVERSIDADE
FEDERAL RURAL
DE PERNAMBUCO

DEINFO
Departamento de Estatística e Informática

Prova de Corretude de Algoritmos

Disciplina: Projeto e Análise de Algoritmos
Alunos: José Bartolomeu Alheiros Dias Neto
Lucas Wagner

Tipos de Algoritmos

Tipos de Algoritmos

Um algoritmo pode ser:

Tipos de Algoritmos

Um algoritmo pode ser:

- Iterativo(com laços for, while, etc);

Tipos de Algoritmos

Um algoritmo pode ser:

- Iterativo(com laços for, while, etc);
- Recursivo(funções que chamam a si próprias).

Fatorial

Fatorial

Recursivo ou Iterativo???

Fatorial

Recursivo ou Iterativo???

- Recursivo;

Fatorial

Recursivo ou Iterativo???

- Recursivo;
- Iterativo.

Fatorial

Recursivo ou Iterativo???

- Recursivo;
- Iterativo.

Prova de Corretude

Se o algoritmo for iterativo, achamos as Invariantes para cada laço e provamos por indução, utilizando as invariantes.

Prova de Corretude

Se o algoritmo for iterativo, achamos as Invariantes para cada laço e provamos por indução, utilizando as invariantes.

Se for recursivo, utilizamos apenas a indução.

Fatorial Recursivo

Fatorial Recursivo

Fat(n)

if n == 0

    Fat(n) = 1

else

    Fat(n) = n · Fat(n − 1)

Fatorial Recursivo

O que queremos provar:

```
Fat(n)
if n == 0
    Fat(n) = 1
else
    Fat(n) = n · Fat(n − 1)
```

Fatorial Recursivo

O que queremos provar:

<u>Proposição de Corretude:</u>

Fat(n)

if n == 0

  Fat(n) = 1

else

  Fat(n) = n · Fat(n − 1)

Fatorial Recursivo

```
Fat(n)

if n == 0

    Fat(n) = 1

else

    Fat(n) = n · Fat(n − 1)
```

O que queremos provar:

<u>Proposição de Corretude:</u>

Fat(n) = n!, n>= 0.

Fatorial Recursivo

Fat(n)

if n == 0

    Fat(n) = 1

else

    Fat(n) = n · Fat(n − 1)

O que queremos provar:

<u>Proposição de Corretude:</u>

Fat(n) = n!, n>= 0.

Caso Base:

Fatorial Recursivo

Fat(n)

if n == 0

    Fat(n) = 1

else

    Fat(n) = n · Fat(n − 1)

O que queremos provar:

<u>Proposição de Corretude:</u>

Fat(n) = n!, n>= 0.

Caso Base:

Fatorial Recursivo

O que queremos provar:

<u>Proposição de Corretude:</u>

Fat(n) = n!, n>= 0.

Fat(n)

if n == 0

Fat(n) = 1

else

Fat(n) = n · Fat(n − 1)

Caso Base:

Para n = 0, nosso algoritmo retorna 1;

Fatorial Recursivo

Fat(n)

if n == 0

    Fat(n) = 1

else

    Fat(n) = n · Fat(n − 1)

O que queremos provar:

Proposição de Corretude:

Fat(n) = n!, n>= 0.

Caso Base:

Para n = 0, nosso algoritmo retorna 1;

Ou seja: Fat(0) = 1
Mas sabemos, pela definição de fatorial, que
0! = 1;

Fatorial Recursivo

Fat(0)

if n == 0

    Fat(n) = 1

else

    Fat(n) = n · Fat(n − 1)

O que queremos provar:

<u>Proposição de Corretude:</u>

Fat(n) = n!, n>= 0.

Caso Base:

Para n = 0, nosso algoritmo retorna 1;

Ou seja: Fat(0) = 1

Mas sabemos que, pela definição de fatorial, 0! = 1;

Então Fat(0) = 1 = 0!
E para o caso base nossa proposição está correta.

Fatorial Recursivo

Fat(0)

if n == 0

    Fat(n) = 1

else

    Fat(n) = n · Fat(n − 1)

O que queremos provar:

<u>Proposição de Corretude:</u>

Fat(n) = n!, n>= 0.

Caso Base: ✓

Para n = 0, nosso algoritmo retorna 1;

Ou seja: Fat(0) = 1

Mas sabemos que, pela definição de fatorial, 0! = 1;

Então Fat(0) = 1 = 0!
E para o caso base nossa proposição está correta.

Fatorial Recursivo

```
Fat(n)

if n == 0

    Fat(n) = 1

else

    Fat(n) = n · Fat(n − 1)
```

O que queremos provar:

<u>Proposição de Corretude:</u>

Fat(n) = n!, n>= 0.

Passo Indutivo:

Fatorial Recursivo

```
Fat(n)

if n == 0

    Fat(n) = 1

else

    Fat(n) = n · Fat(n − 1)
```

O que queremos provar:

Proposição de Corretude:

Fat(n) = n!, n>= 0.

Passo Indutivo:

H.I.(Hipótese de Indução): Supomos que para um valor k, k > 0, nossa proposição está correta:

Fatorial Recursivo

Fat(n)

if n == 0

    Fat(n) = 1

else

    Fat(n) = n · Fat(n − 1)

O que queremos provar:

<u>Proposição de Corretude:</u>

Fat(n) = n!, n>= 0.

Passo Indutivo:

H.I.(Hipótese de Indução): Supomos que para um valor k, k > 0, nossa proposição está correta:

*Fat(k) = k * Fat(k -1) = k!*

Fatorial Recursivo

Passo Indutivo(cont...):

Tese: Provamos que para um valor $n = k + 1$ nossa proposição está correta, utilizando a H.I.:

```
Fat(n)
if n == 0
    Fat(n) = 1
else
    Fat(n) = n · Fat(n − 1)
```

Fatorial Recursivo

Fat(n)

if n == 0

Fat(n) = 1

else

Fat(n) = n · Fat(n − 1)

Passo Indutivo(cont...):

Tese: Provamos que para um valor n = k + 1 nossa proposição está correta, utilizando a H.I.:

*Fat(k+1) = k + 1 * Fat(k+ 1 - 1)*

Fatorial Recursivo

Fat(n)

if n == 0

    Fat(n) = 1

else

    Fat(n) = n · Fat(n − 1)

Passo Indutivo(cont…):

Tese: Provamos que para um valor n = k + 1 nossa proposição está correta, utilizando a H.I.:

*Fat(k+1) = k + 1 * Fat(k+ 1 - 1)*
*(1) Fat(k+1) = k + 1 * Fat(k)*

Fatorial Recursivo

Fat(n)

if n == 0

Fat(n) = 1

else

$Fat(n) = n \cdot Fat(n - 1)$

Passo Indutivo(cont…):

Tese: Provamos que para um valor n = k + 1 nossa proposição está correta, utilizando a H.I.:

*Fat(k+1) = k + 1 * Fat(k+ 1 - 1)*
*(1) Fat(k+1) = k + 1 * Fat(k)*

Mas, pela H.I., sabemos que:

Fatorial Recursivo

Fat(n)

if n == 0

    Fat(n) = 1

else

    Fat(n) = n · Fat(n − 1)

Passo Indutivo(cont...):

Tese: Provamos que para um valor n = k + 1 nossa proposição está correta, utilizando a H.I.:

*Fat(k+1) = k + 1 * Fat(k+ 1 - 1)*
*(1) Fat(k+1) = k + 1 * Fat(k) → (2)*

Mas, pela H.I., sabemos que:

*(2) Fat(k) = k * Fat(k -1)*

Fatorial Recursivo

Fat(n)

if n == 0

    Fat(n) = 1

else

    Fat(n) = n · Fat(n − 1)

Passo Indutivo(cont…):

Tese: Provamos que para um valor n = k + 1 nossa proposição está correta, utilizando a H.I.:

*Fat(k+1) = k + 1 * Fat(k+ 1 - 1)*
*(1) Fat(k+1) = k + 1 * $\boxed{Fat(k)}$ → (2)*

Mas, pela H.I., sabemos que:

(2) Fat(k) = k!

*Fat(k+1) = (k + 1) * $\boxed{(k!)}$*

Fatorial Recursivo

```
Fat(n)

if n == 0

    Fat(n) = 1

else

    Fat(n) = n · Fat(n − 1)
```

Passo Indutivo(cont…):

Tese: Provamos que para um valor n = k + 1 nossa proposição está correta, utilizando a H.I.:

$Fat(k+1) = k + 1 * Fat(k+ 1 - 1)$

$(1)\ Fat(k+1) = k + 1 * \boxed{Fat(k)} \rightarrow (2)$

Mas, pela H.I., sabemos que:

$(2)\ Fat(k) = k!$

$Fat(k+1) = (k + 1) * \boxed{(k!)}$

$Fat(k+1) = (k + 1)!$

Fatorial Recursivo

```
Fat(n)
if n == 0
    Fat(n) = 1
else
    Fat(n) = n · Fat(n − 1)
```

Passo Indutivo(cont…):

Tese: Provamos que para um valor n = k + 1 nossa proposição está correta, utilizando a H.I.:

*Fat(k+1) = k + 1 * Fat(k+ 1 - 1)*
*(1) Fat(k+1) = k + 1 * $\boxed{\text{Fat(k)}}$* → *(2)*

Mas, pela H.I., sabemos que:

(2) Fat(k) = k!

*Fat(k+1) = (k + 1) * $\boxed{\text{(k!)}}$*

Fat(k+1) = (k + 1)! ✓

C.Q.D

Fatorial Recursivo

Fat(n)

if n == 0

Fat(n) = 1

else

Fat(n) = n · Fat(n − 1)

Conclusão:

Nosso algoritmo de Fatorial Recursivo é correto, pois conseguimos provar, usando indução matemática, que para qualquer inteiro k >= 0, o algoritmo é correspondente à função matemática de Fatorial:

Fatorial Recursivo

```
Fat(n)

if n == 0

    Fat(n) = 1

else

    Fat(n) = n · Fat(n − 1)
```

Conclusão:

Nosso algoritmo de Fatorial Recursivo é correto, pois conseguimos provar, usando indução matemática, que para qualquer inteiro k >= 0, o algoritmo é correspondente à função matemática de Fatorial:

n! = n*(n-1)*(n-2)...0!, para qualquer inteiro n>=0.

Fatorial Recursivo

Fat(n)

if n == 0

Fat(n) = 1

else

Fat(n) = n · Fat(n − 1)

Conclusão:

Nosso algoritmo de Fatorial Recursivo é correto, pois conseguimos provar, usando indução matemática, que para qualquer inteiro k >= 0, o algoritmo é correspondente à função matemática de Fatorial:

n! = n*(n-1)*(n-2)...0!, para qualquer inteiro n>=0.

Exercício: máximo de um vetor com n elementos

Maximo(A,n)

if n≤1 **then**

    return A[1]

else

    return max(Maximo(A,n – 1), A[n])