Avaliação III

Correção

20141204

1 Minimax (2)

Na Teoria de Sistemas, define-se como elemento minimax de uma matriz o menor elemento da linha em que se encontra o maior elemento da matriz.

Escreva um algoritmo que:

- lê dois inteiros m, n, com $0 < n, m \leq 100$ e uma matriz $A : m \times n$ de elementos de tipo inteiro,
- determina e escreve o elemento minimax de A.

```
1  algoritmo "Minimax"
2  var n, m, i, j: inteiro
3     // max da linha corrente e linha onde ocorre o minimax
4      linha_minimax: inteiro
5      max, minimax: inteiro // max da matriz e minimax da matrix
6      a : vetor [1..100,1..100] de inteiro
7  inicio
8    repita
9     leia(n)
10    leia(m)
11   ate (n > 0) e (m > 0) e (n <= 100) e (m <= 100)
12
13   para i de 1 ate n faca
14     para j de 1 ate m faca
15       leia(a[i, j])
16     fimpara
17   fimpara
18
19   max <- a[1, 1]
20   linha_minimax <- 1
21   para i de 1 ate n faca
22     para j de 1 ate m faca
23       se a[i, j] > max entao
24         max <- a[i, j]
25         linha_minimax <- i
26       fimse
27     fimpara
28   fimpara
29
30   minimax <- a[linha_minimax, 1]
31   para j de 2 ate m faca
32     se a[linha_minimax, j] < minimax entao
33       minimax <- a[linha_minimax, j]
34     fimse
35   fimpara
36
37  escreva(minimax)
38 fimalgoritmo
```

2 Sufixo (3)

Um número b é dito ser *sufixo* de um número a se o número formado pelos últimos dígitos de a são iguais a b. Por exemplo:

a	b		
567890	890	$\rightarrow$	sufixo
1234	1234	$\rightarrow$	sufixo
2457	245	$\rightarrow$	não é sufixo
457	2457	$\rightarrow$	não é sufixo

1. Escreva uma função *sufixo* que dados dois números inteiros a e b verifica se b é um sufixo de a.

1. Utilize a função escrita em 1 para escrever um algoritmo que

   (a) lê dois números inteiros positivos a e b, e

   (b) verifica se o menor deles é subsequência do outro. Exemplo:

a	b		
567890	678	$\rightarrow$	b é subsequência de a
1234	2212345	$\rightarrow$	a é subsequência de b
235	236	$\rightarrow$	Um não é subsequência do outro

```
algoritmo "Subsequence"
var a, b : inteiro

funcao sufixo(a, b: inteiro): logico
inicio
   se a < b entao retorne falso
   senao
    se b < 10 entao
     retorne (a % 10) = b
    senao
     retorne ((a % 10) = (b % 10)) e (sufixo(a \ 10, b \ 10))
    fimse
   fimse
fimfuncao

// b <= a
funcao is_subsequence(a, b: inteiro): logico
inicio
   se a < b entao retorne falso
   senao
    retorne sufixo(a, b) ou is_subsequence(a \ 10, b)
   fimse
fimfuncao

inicio
  // Testes sufixo
  escreval(sufixo(567890, 890)) // verdadeiro
  escreval(sufixo(1234, 1234))  // verdadeiro
  escreval(sufixo(2457, 245))   // falso
  escreval(sufixo(457, 2457))   // falso

  // Testes is_subsequence
  escreval(is_subsequence(567890, 678))
  escreval(is_subsequence(2212345, 1234))
  escreval(is_subsequence(235, 236))

  repita
```

```
38      leia(a)
39      leia(b)
40    ate (a > 0) e (b > 0)
41
42    se a > b entao
43      escreval(is_subsequence(a, b))
44    senao
45      escreval(is_subsequence(b, a))
46    fimse
47  fimalgoritmo
```

3 Binário (2)

Escreva uma função **recursiva** que

- tem um parâmetro formal inteiro positivo n
- retorna o número de dígitos necessários para a representá-lo em binário

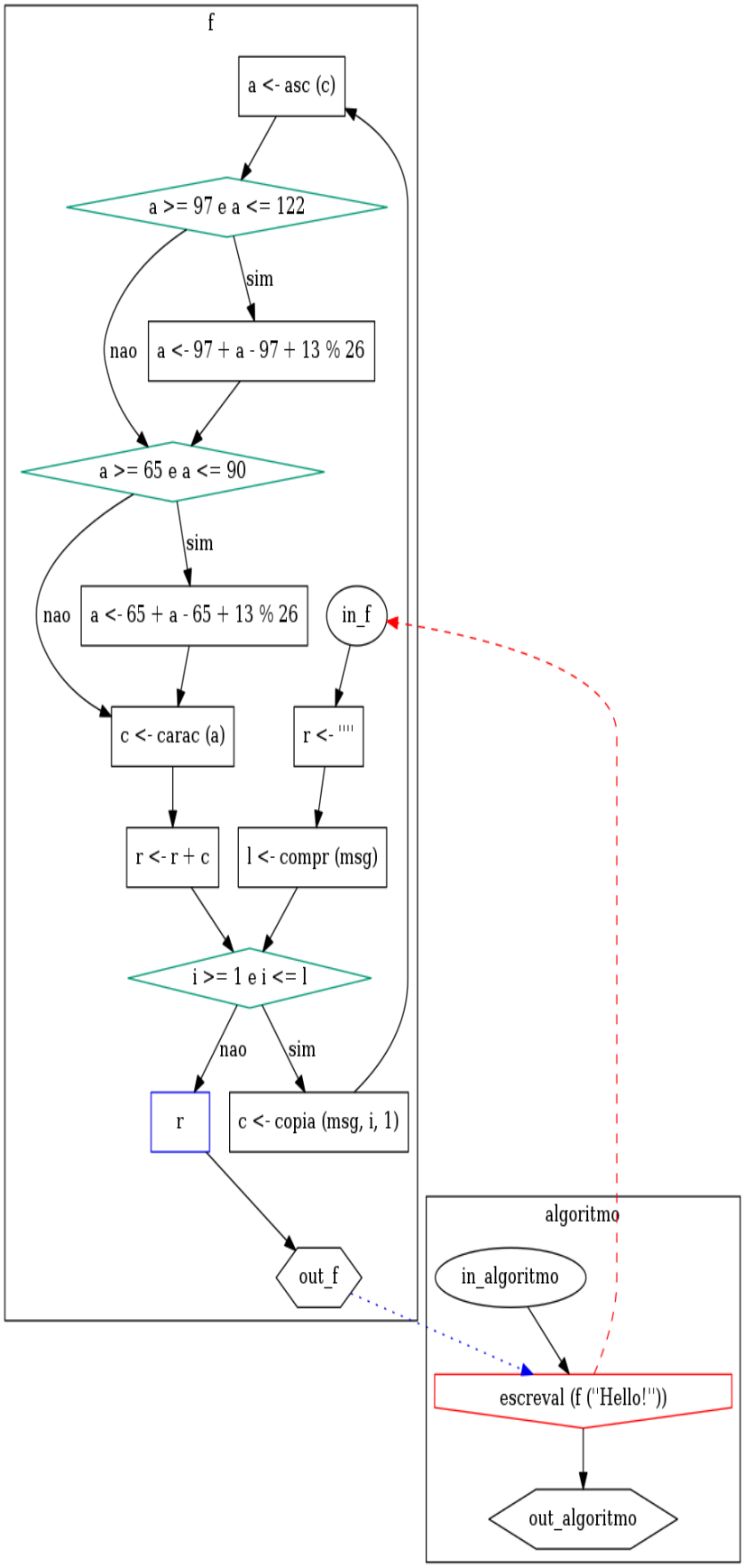
Natural	Binário	Retorno
1	1	1
2	10	2
7	111	3
14	1110	4

```
algoritmo "Binary"

// requires n >= 0
funcao nbinarydigits(n : inteiro): inteiro
inicio
  se n <= 1 entao retorne 1
  senao retorne 1 + nbinarydigits(n \ 2)
  fimse
fimfuncao

inicio
  // Testes
  escreval(nbinarydigits(1))
  escreval(nbinarydigits(2))
  escreval(nbinarydigits(7))
  escreval(nbinarydigits(14))
fimalgoritmo
```

4 Criptografia (3)

```
algoritmo "C"
funcao f(msg: caractere): caractere
var i, a, l : inteiro
    r, c : caractere
inicio
  r <- ""
  l <- compr(msg)
  para i de 1 ate l faca
    c <- copia(msg, i, 1)
    a <- asc(c)
    se ((a >= 97) e (a <= 122)) entao
      a <- 97 + (a - 97 + 13) % 26
    fimse
    se ((a >= 65) e (a <= 90)) entao
      a <- 65 + (a - 65 + 13) % 26
    fimse
    c <- carac(a)
    r <- r + c
  fimpara
  retorne r
fimfuncao

inicio
  escreval(f("Hello!"))
fimalgoritmo
```

1. Desenha o fluxograma da função `f`.



2. O que será exibido na tela na execução desse algoritmo?

    - `Uryyb!`

3. Descreva, com as suas palavras, o que faz esse algoritmo.

    - É o algoritmo ROT-13. Ele faz uma rotação de 13 posições das letras do alfabeto (e não dos outros caracteres) do texto. É uma cifra de César. Veja `http://pt.wikipedia.org/wiki/ROT13`

Elementos da biblioteca padrão

- **asc (s : caractere)** retorna um inteiro com o código ASCII do primeiro caractere da expressão.
- **carac (c : inteiro)** retorna o caractere cujo código ASCII corresponde à expressão.
- **compr (c : caractere)** retorna um inteiro contendo o comprimento (quantidade de caracteres) da expressão.
- **copia (c : caractere ; p, n : inteiro)** retorna um valor do tipo caractere contendo uma cópia parcial da expressão, a partir do caractere p, contendo n caracteres. Os caracteres são numerados da esquerda à direita, começando de 1.