Computação e Programação

(LEA, LEB, LEGI, LEMa, LEGM, LEQ, LQ)

Exame 2A

Departamento de Matemática, IST

31 de Janeiro de 2005, 9h00m

Duração: 2h30m

Nome: _____ Número: _____

Curso: _____ Nota: _____

Grupo I [1,5 + 1,5]

a) Desenvolva em MATLAB uma função `f` que recebe como argumento um vector de números inteiros, percorre o vector dado elemento a elemento recorrendo a um ciclo `while` e calcula o vector do mesmo comprimento que em cada posição contém o maior número par encontrado até essa posição no vector dado. Caso ainda não tenha sido encontrado nenhum número par, considera-se 0.

b) Desenvolva em MATLAB uma função `g` que recebe como argumento uma matriz de números inteiros, percorre a matriz dada elemento a elemento recorrendo a dois ciclos `while` encaixados e calcula o vector de comprimento igual ao número de colunas que contém em cada posição o maior número par que ocorre na coluna de índice correspondente. Caso não ocorra nenhum número par, considera-se 0.

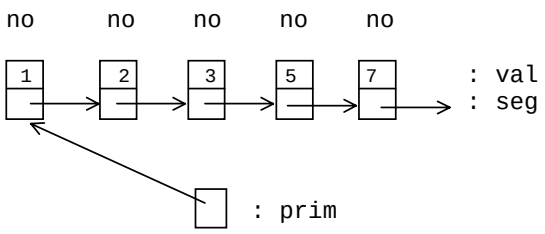
Grupo II [2,0 + 2,0]

a) Desenvolva em F um módulo que disponibilize as operações sobre *conjuntos de inteiros* a seguir descritas, escolhendo a implementação seguinte para o tipo `conj`:

```
type, private :: no
   integer :: val
   type(no), pointer :: seg
end type no

type, public :: conj
   private
   type(no), pointer :: prim
end type conj
```

Com esta implementação, pretende-se que o conjunto `{5,2,3,1,7}` seja representado por:



Note que a lista de nós encadeados deve ser mantida ordenada por ordem crescente. As operações disponibilizadas pelo módulo são as seguintes:

- `vaz()`: função que retorna o conjunto vazio;
- `acr(x,c)`: subrotina que recebe no parâmetro de entrada `x` um número inteiro e no parâmetro de entrada/saída `c` um conjunto e devolve em `c` o conjunto que resulta de acrescentar, caso seja possível, o elemento a esse conjunto. Recorde que num conjunto não existem repetições. Assim, acrescentar a um conjunto um elemento que já lá exista deverá resultar no mesmo conjunto;
- `apg(x,c)`: subrotina que recebe no parâmetro de entrada `x` um número inteiro e no parâmetro de entrada/saída `c` um conjunto e devolve em `c` o conjunto que resulta de apagar o elemento do conjunto. No caso de o elemento não ocorrer no conjunto este não sofre alterações;
- `prt(x,c,b)`: subrotina que recebe no parâmetro de entrada `x` um número inteiro, no parâmetro de entrada/saída `c` um conjunto e devolve no parâmetro de saída `b` o valor lógico `.true.` se o elemento pertencer ao conjunto e o valor lógico `.false.` caso contrário.
- `crd(c,n)`: subrotina que recebe no parâmetro de entrada/saída `c` um conjunto e devolve no parâmetro de saída `n` o cardinal desse conjunto;
- `min(c,m)`: subrotina que recebe no parâmetro de entrada `c` um conjunto não vazio e devolve no parâmetro de saída `m` o mínimo desse conjunto;
- `max(c,m)`: subrotina que recebe no parâmetro de entrada `c` um conjunto não vazio e devolve no parâmetro de saída `m` o máximo desse conjunto.

b) Desenvolva em F, recorrendo ao módulo acima, uma subrotina `interseccao(c1,c2,c)` que recebe nos parâmetros de entrada/saída `c1` e `c2` dois conjuntos e devolve no parâmetro de saída `c` o conjunto resultante da intersecção de `c1` e `c2`.

Grupo III [1,0 + 2,0]

a) Descreva as diversas formas de passagem de parâmetros numa subrotina. Explique a importância de cada uma dessas formas de passagem de parâmetros.

b) Considere a função `g` seguinte, definida em F. Descreva a funcionalidade de `g`. Defina uma versão recursiva de `g`, com a mesma funcionalidade.

```
function g(v1,v2) result(w)
integer, dimension(:), intent(in) :: v1,v2
integer, dimension(max(size(v1),size(v2))) :: w
integer :: i

do i=1,size(w)
   if (i>size(v1)) then
      w(i)=v2(i)
   elseif (i>size(v2)) then
      w(i)=v1(i)
   elseif (v1(i)>v2(i)) then
      w(i)=v1(i)
   else
      w(i)=v2(i)
   end if
end do
end function g
```