Computação e Programação

(LEAmb, LEBl, LEGI, LEGM, LEMat, LEQ, LQ)

Departamento de Matemática, IST

Exame 2

27 de Janeiro de 2006, 9h00

Duração: 2h30

não preencher
I –
II –
III – ___
T:

Curso: _____ Número: _____ Nome: _____

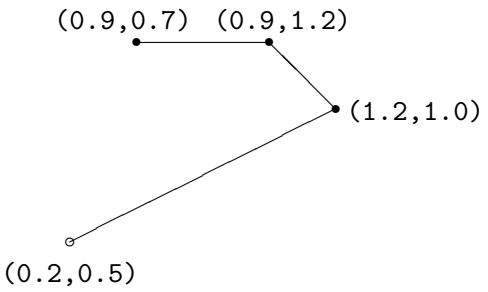
Grupo I [1,5+1,5]

a) Defina em Matlab uma função `f` que quando recebe como argumento um vector de números inteiros percorre as posições de índice par desse vector recorrendo a um ciclo `while` e devolve o número desses elementos que são números primos.

b) Defina em Matlab uma função `g` que quando recebe como argumento uma matriz de números inteiros percorre as linhas pares dessa matriz recorrendo a dois ciclos `while` encaixados e devolve o número de elementos dessas linhas que são números primos.

Grupo II [2,0+2,0]

Uma linha poligonal é uma sequência de pontos (vértices) em que cada ponto está unido ao seguinte por um segmento de recta (aresta). Ao primeiro ponto da linha poligonal chama-se origem. A figura seguinte apresenta um exemplo duma linha poligonal com origem em `(0.2,0.5)` e três arestas.

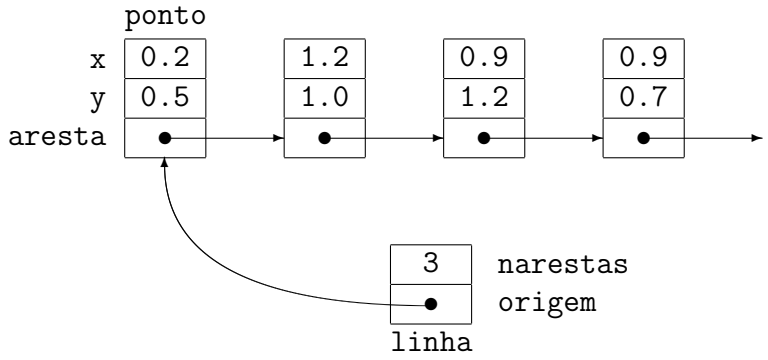


a) Desenvolva em F as funções de um módulo que disponibilize as operações a seguir descritas sobre linhas poligonais no plano real; o módulo deve recorrer à seguinte implementação do tipo de dados `linha` (onde uma linha poligonal é representada pela sequência dos seus vértices, começando pela origem).

```
type, public :: linha
   private
   integer :: narestas
   type(ponto), pointer :: origem
end type linha

type, private :: ponto
   real :: x,y
   type(ponto), pointer :: aresta
end type ponto
```

Com esta implementação, a linha poligonal do exemplo anterior seria implementada da seguinte forma.

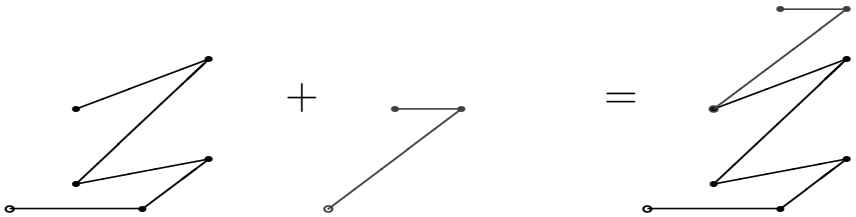


As operações disponibilizadas pelo módulo são as seguintes:

- `novo(x,y)`: função que recebe as coordenadas reais da origem da linha poligonal e devolve uma linha poligonal sem arestas;

- `acresc(z,x,y)`: subrotina que recebe no parâmetro de entrada/saída `z` uma linha poligonal e nos parâmetros de entrada `x` e `y` dois reais e devolve em `z` o resultado de acrescentar a esta linha poligonal uma aresta entre o seu último ponto e o ponto (`x`,`y`);
- `ntrocos(z)`: função que recebe no parâmetro de entrada `z` uma linha poligonal e devolve o número de arestas que a compõem;
- `comp(z,x)`: subrotina que recebe no parâmetro de entrada/saída `z` uma linha poligonal e devolve no parâmetro de saída `x` o comprimento total dessa linha (definido como a soma dos comprimentos dos segmentos de recta que a constituem);
- `ponto-n(z,n,x,y)`: subrotina que recebe no parâmetro de entrada/saída `z` uma linha poligonal e no parâmetro de entrada `n` um natural entre 1 e o número de vértices da linha e devolve nos parâmetros de saída `x` e `y` as coordenadas do `n`-ésimo ponto da linha poligonal;
- `equilatero(z,n)`: subrotina que recebe no parâmetro de entrada/saída `z` uma linha poligonal e devolve no parâmetro de saída `n` o número de arestas de `z`, caso `z` represente um polígono fechado equilátero, ou 0, caso contrário. **Nota:** um polígono fechado é uma linha poligonal cujo último ponto é a origem; um polígono equilátero é um polígono em que todos os lados têm o mesmo comprimento.

b) Desenvolva em F, recorrendo ao módulo acima, uma subrotina `compoe(z,w)` que recebe nos parâmetros de entrada/saída duas linhas poligonais `z` e `w` e devolve em `z` a linha poligonal resultante de juntar `w` a `z`.



Observação: a linha resultante de juntar `w` a `z` pode ser descrita da seguinte forma. Sejam n o número de vértices de `z`; os primeiros n pontos da nova linha são os mesmos de `z`. As coordenadas do ponto $n+i-1$ da nova linha são obtidas somando Δ_x e Δ_y às coordenadas do ponto i de `w`, para $i \geq 2$, sendo Δ_x e Δ_y escolhidos por forma a que a origem de `w` fique sobreposta com o último ponto de `z`.

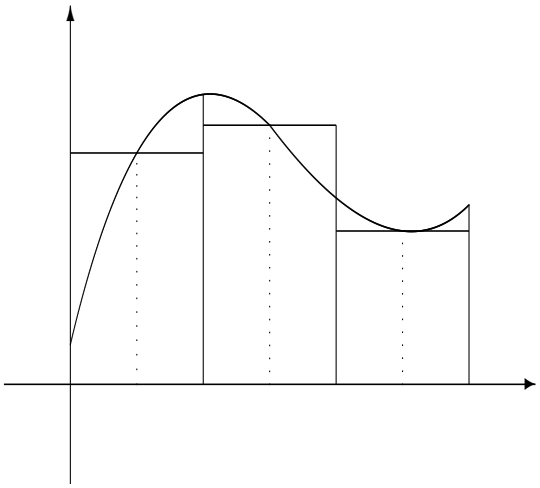
Grupo III [2,0+1,0]

a) Considere a seguinte função `g`, definida em F. Descreva a funcionalidade desta função. Defina recursivamente uma função `h` com a mesma funcionalidade de `g`

```
function g(v) result(y)
integer, dimension(:), intent(in) :: v
integer :: y
integer :: i,j,n

n=size(v)
i=1
y=0
do
   j=n-i+1
   if (i>j) then
      exit
   else
      if (v(i)==v(j)) then
         y=y+1
      end if
      i=i+1
   end if
end do
end function g
```

b) Seja f uma função contínua e positiva num intervalo $[a, b]$. A área entre o eixo das abcissas e o gráfico de f no intervalo $[a, b]$ (o integral de f em $[a, b]$) pode ser calculada com um erro tão pequeno quanto se queira através do seguinte método: divide-se o intervalo $[a, b]$ em n subintervalos de igual comprimento e aproxima-se o integral de f em cada subintervalo pela área do rectângulo com base no eixo das abcissas e altura igual ao valor da função no ponto médio do intervalo (veja a figura).



Implemente em F este método de cálculo de integrais através de uma função `integral` que receba como argumentos os reais `a` e `b` (extremos do intervalo), o inteiro positivo `n` (número de subintervalos) e a função real de variável real `f` e devolva a aprovimação encontrada para o integral de `f` em `[a,b]`.