Computação e Programação

Textos de apoio às aulas práticas

Jaime Ramos, Francisco Miguel Dionísio

DMIST, Dezembro de 2010

Parte I

MATLAB

Capítulo 1

Exercícios preliminares

1. Defina a função `conta_divisores` que recebe como argumento um inteiro positivo e devolve o número de divisores desse número.

2. Defina a função `primo` que recebe como argumento um inteiro positivo e devolve verdadeiro se esse número é primo e falso caso contrário.

3. Defina a função `soma_primos` que recebe como argumento um inteiro positivo n e devolve a soma dos números primos até n.

4. Defina a função `div` que recebe como argumento dois números inteiros m e n e devolve o resultado da divisão inteira de m por n.

5. Defina a função `prim_alg` que recebe como argumento um inteiro positivo n e devolve o primeiro algarismo da representação decimal de n. Por exemplo, `prim_alg`$(54763) = 5$.

6. Um *número perfeito* é um número inteiro positivo tal que é a soma dos seus divisores próprios, isto é, a soma de todos os divisores excluindo o próprio número. Por exemplo, 6 é um número perfeito porque os seus divisores próprios são 1, 2 e 3, e $1 + 2 + 3 = 6$. Defina a função `num_perf` que recebe como argumento um número inteiro positivo e devolve verdadeiro se esse número for um número perfeito e falso caso contrário.

7. O *algoritmo de Euclides* é um método simples e eficiente para calcular o máximo divisor comum (mdc). Este algoritmo baseia-se no facto de o máximo divisor comum verificar as seguintes igualdades:

$$\begin{cases} mdc(a,b) = mdc(a-b,b) & \text{se } a > b \\ mdc(a,b) = mdc(a,b-a) & \text{se } a < b \\ mdc(a,b) = a & \text{se } a = b \end{cases}$$

   O algoritmo vai subtraindo sucessivamente o menor argumento ao maior até que sejam iguais. Nessas condições, o máximo divisor comum será esse valor. Defina a função `mdc` que recebe como argumento dois números naturais e que calcula o máximo divisor comum desses dois números, recorrendo o algoritmo de Euclides.

8. Define-se *raiz quadrada inteira* de um número inteiro positivo n como sendo o maior número inteiro positivo m menor ou igual do que a raiz quadrada de n. Por exemplo, a raiz quadrada inteira de 27 é 5 porque $5^2 = 25 \leq 27$ e $6^2 = 36 > 27$.

   Um algoritmo utilizado frequentemente para aproximar a raiz quadrada inteira de um número, que a seguir se descreve, baseia-se no método de Newton. Define-se uma sucessão $\{x_k\}$ da seguinte forma:

   - $x_0 = n$;

- $x_{k+1} = (x_k + \frac{n}{x_k})/2$.

O método consiste em calcular valores sucessivos de x_k até que $|x_{k+1} - x_k| < 1$. Nestas condições a solução será $\lfloor x_{k+1} \rfloor$.

Defina a função `intsqrt` que recebe como argumento um inteiro positivo n e devolve a aproximação encontrada para a raiz quadrada inteira de n, recorrendo ao método descrito anteriormente.
Nota: Por $\lfloor x \rfloor$ entende-se a característica do número x, ou seja, o menor número inteiro que não excede x.

Capítulo 2

Programação imperativa sobre vectores

1. Defina a função `pertence` que dado um número n e um vector v de números devolve verdadeiro se n pertence a v e falso caso contrário.
2. Defina a função `ocorrencias` que dado um número n e um vector v conta quantas vezes n ocorre em v.
3. Defina a função `pos_max` que dado um vector v devolve a soma dos índices das posições onde o máximo de v ocorre.
4. Defina a função `split` que dado um vector v e um número n devolve um triplo $[a, b, c]$ tal que:
   - a é o número de elementos de v maiores que n,
   - b é o número de elementos de v iguais a n,
   - c é o número de elementos de v menores que n.
5. Defina a função `f` que dado um vector v devolve um par $[x, y]$ tal que
   - x é 1 se os elementos do vector estiverem ordenados por ordem crescente (em sentido lato) e é 0 caso contrário;
   - y é o número de vezes que um elemento é sucedido de outro estritamente maior.
6. Defina a função `f` que dado um vector v devolve um par $[x, y]$ tal que
   - x é 1 se o vector for constante (isto é, se todos os elementos forem iguais) e é 0 caso contrário;
   - y é o número de vezes que um elemento é sucedido de outro diferente.
7. Defina a função `f` que dado um vector v devolve um par $[x, y]$ tal que
   - x é a diferença entre o máximo e o mínimo do vector;
   - y é o número de vezes que o máximo ocorre na lista.
8. Defina a função `f` que dado um vector v devolve um par $[x, y]$ tal que
   - x é a soma das posições onde ocorre o mínimo do vector;
   - y é a última posição em que o mínimo do vector ocorre.
9. Defina a função `f` que dado um vector v devolve um par $[x, y]$ tal que

- x é o menor número primo que ocorre no vector;
- y é o número de números primos que ocorrem no vector.

No caso de não ocorrerem números primos no vector, x deve ser 0.

10. Defina a função `f` que dado um vector v devolve um par $[x, y]$ tal que

    - x é 1 se todos os elementos do vector forem pares e 0 caso contrário;;
    - y é o menor número par.

    No caso de não ocorrerem números pares no vector, y deve ser 0.

11. Defina a função `f` que dado um vector v devolve um par $[x, y]$ tal que

    - x é 1 se no vector ocorrerem mais números pares do que ímpares e 0 caso contrário;
    - y é a diferença entre o número de números pares e o número de números ímpares.

12. Defina a função `car_primos` que dado um vector v devolve um vector de igual comprimento com 1 nas posições onde v contém um número primo e 0 nas restantes.

13. Defina a função `f` que dado um vector v devolve um vector de igual comprimento tal que cada elemento do argumento é substituído:

    - pelo índice da posição que ocupa, caso esse elemento seja um número par;
    - por 0 caso contrário.

14. Defina a função `f` que dado um vector v devolve um vector de igual comprimento tal que cada elemento do argumento é substituído por:

    - 2 se esse elemento for maior do que o elemento na posição anterior;
    - 1 se esse elemento for igual ao elemento na posição anterior;
    - 0 caso contrário.

    O elemento na primeira posição, que não tem elemento na posição anterior, deve ser substituído por 0.

15. Defina a função `f` que dado um vector v devolve um vector de igual comprimento que contém em cada posição o número de números ímpares encontrados até essa posição (inclusive) no vector dado

16. Defina a função `f` que dado um vector v devolve um vector de igual comprimento tal que cada elemento do argumento é substituído:

    - pela soma do elemento com a posição que ocupa, caso o elemento ocorra no vector entre dois números pares;
    - pela sua posição, caso contrário.

17. Defina a função `f` que dado um vector v devolve um vector de igual comprimento com as posições trocadas duas a duas, isto é, os elementos da primeira e da segunda posição trocados entre si, os elementos da terceira e quarta posições trocados entre si, e assim sucessivamente. No caso de o vector ter comprimento ímpar a última posição não se altera.

18. Defina a função `f` que dado um vector v devolve um vector de igual comprimento tal que cada elemento do argumento é substituído:

    - pelo elemento que o antecede, caso o elemento anterior seja inferior à posição que ocupa;
    - por 0 caso contrário, caso contrário.

19. Um vector u diz-se prefixo de um vector v se o comprimento de u for menor ou igual do que o comprimento de v e $u(i) = v(i)$, para todo o $i = 1, \ldots, length(u)$. Defina a função `prefixo` que recebe como argumento dois vectores u e v e devolve verdadeiro se u é prefixo de v e falso caso contrário.

20. Diz-se que um vector tem um *elemento maioritário* se mais de metade dos seus elementos forem iguais. Defina uma função `maioritario` que dado um vector de números inteiros (positivos) e devolve o elemento maioritário do vector, caso exista. No caso do vector não ter elemento maioritário, a função deverá devolver -1.

21. O *reconhecimento de padrões* consiste em verificar se uma determinada sequência padrão `p` ocorre numa outra sequência `s`. Considere que, quer a sequência padrão `p`, quer a sequência `s`, são vectores de inteiros. Pretende-se verificar se os elementos do vector `p` ocorrem no vector `s`, pela mesma ordem e consecutivamente. Apresentam-se em seguida alguns exemplos.

    - O padrão $[1, 2, 3]$ ocorre na sequência $[2, \mathbf{1}, \mathbf{2}, \mathbf{3}, 4, 5]$;
    - O padrão $[4, 3, 4]$ não ocorre na sequência $[2, 1, 4, 2, 3, 4, 5]$;
    - O padrão $[5, 2, 7]$ ocorre na sequência $[\mathbf{5}, \mathbf{2}, \mathbf{7}]$;
    - O padrão $[9, 4]$ ocorre na sequência $[3, 2, 4, 2, \mathbf{9}, \mathbf{4}]$;
    - O padrão $[7, 4, 3]$ ocorre na sequência $[\mathbf{7}, \mathbf{4}, \mathbf{3}, 7, 2, 7]$.

    Defina uma função `reconhece` que recebe como argumento dois vectores `p` e `s` de números inteiros e devolve 1 se o padrão `p` ocorre na sequência `s`, e devolve 0 caso contrário.

Capítulo 3

Programação imperativa sobre matrizes

1. Defina a função `media_matriz` que recebe como argumento uma matriz e devolve a média dos seus elementos.

2. Defina a função `matriz_igual` que recebe como argumento duas matrizes de igual dimensão e devolve verdadeiro se ambas as matrizes são iguais elemento a elemento e falso caso contrário.

3. Defina a função `linha_par` que recebe como argumento um vector de inteiros e devolve verdadeiro se a soma dos elementos é par e falso caso contrário.

   Generalize esta função de modo a receber como argumento uma matriz e a devolver um vector e comprimento igual ao número de linhas da matriz, que contém na i-ésima posição o valor 1 se a linha i satisfizer `linha_par` e 0 caso contrário.

4. Defina a função `l_primos` que recebe como argumento uma matriz de números inteiros e devolve o número de linhas em que ocorre, pelo menos, um número primo. Pode utilizar, no máximo, dois ciclos encaixados.

5. Defina a função `col_min` que recebe como argumento uma matriz de números inteiros e devolve o número de colunas em que ocorre o mínimo da matriz. Pode utilizar, no máximo, dois ciclos encaixados, e não pode utilizar a função `min`.

6. Defina a função `f` que recebe como argumento uma matriz e devolve um par $[x, y]$ tal que:

   - x é o número de linhas constituídas exclusivamente por números pares;
   - y é o índice da primeira dessas linhas.

   Pode utilizar, no máximo, dois ciclos encaixados. Caso não existam linhas nestas condições, x deve ser 1 e y deve ser 0.

7. Defina a função `f` que recebe como argumento uma matriz quadrada de números inteiros e devolve um par $[x, y]$ tal que:

   - x é a diferença entre a soma dos elementos da diagonal principal da matriz e a soma dos elementos da diagonal oposta;
   - y é a diferença entre a soma dos elementos da triangular superior e a soma dos elementos da triangular inferior.

Pode utilizar, no máximo, dois ciclos encaixados. Os elementos das triangulares não incluem os elementos da diagonal principal.

8. Defina a função `f` que recebe como argumento uma matriz quadrada de números inteiros e devolve um par $[x, y]$ tal que:

   - x é o número de números pares que se encontram acima das duas diagonais;
   - y é o número de números pares que se encontram abaixo das duas diagonais.

   Pode utilizar, no máximo, dois ciclos encaixados.

9. Defina a função `f` que recebe como argumento uma matriz quadrada de números inteiros e devolve 1 se o número de números pares que acima das diagonais for igual ao número de números impares que ocorre abaixo das diagonais. Pode utilizar, no máximo, dois ciclos encaixados.

10. Defina a função `f` que recebe como argumento uma matriz quadrada de números inteiros e devolve o maior elemento que ocorre entre as duas diagonais. Pode utilizar, no máximo, dois ciclos encaixados. No caso da matriz singular, a função deve devolver `-inf`.

11. Defina a função `f` que recebe como argumento uma matriz de números inteiros e devolve um vector de comprimento igual ao número de linhas que contém na posição k o valor:

    - 2 se na linha k existirem mais números pares do que ímpares;
    - 1 se na linha k existirem mais números ímpares do que pares;
    - 0 se na linha k existirem tantos números pares como ímpares.

12. Defina a função `f` que recebe como argumento uma matriz de números inteiros e devolve um vector de comprimento igual ao número de linhas que contém na posição k a diferença entre o maior número ímpar e o menor número ímpar que ocorrem na linha k. Se não ocorrerem números ímpares considera-se o valor 0.

13. Defina a função `f` que recebe como argumento uma matriz de números inteiros e devolve um vector de comprimento igual ao número de colunas que contém na posição k a soma dos elementos pares da coluna k que ocorrem em linhas ímpares.

14. Defina a função `f` que recebe como argumento uma matriz de números inteiros e devolve um vector de comprimento igual ao número de colunas que contém na posição k o maior elemento da coluna k.

15. Defina a função `f` que recebe como argumento uma matriz de números inteiros e devolve um vector de comprimento igual ao número de linhas que contém na posição k o índice a primeira coluna onde ocorre o maior elemento da linha k.

Capítulo 4

Programação recursiva

Os exercícios desta secção devem ser resolvidos recorrendo a *funções definidas por recursão.*

1. Defina a função `soma_natR` que recebe como argumento um número inteiro positivo n e devolve a soma de todos os naturais até n.

2. Defina a função `divR` que recebe como argumento dois números inteiros positivos m e n e devolve o resultado da divisão inteira de m por n.

3. Defina a função `prim_algR` que recebe como argumento um número inteiro positivo n e devolve o primeiro algarismo na representação de n.

4. Defina a função `num_perfR` que recebe como argumento um número inteiro positivo e que devolve verdadeiro se esse número for um número perfeito e falso caso contrário. Se necessário, recorde a noção de *número perfeito* no Exercício 6 do Capítulo 1.

5. Defina a função `mdcR` que recebe como argumento dois números naturais e devolve o seu máximo divisor comum, calculado através do *algoritmo de Euclides.*

6. Considere a função $f : \mathbb{N} \to \mathbb{N}$ tal que

$$f(x) = \begin{cases} x/2 & \text{se } x \text{ for um número par} \\ 3x+1 & \text{caso contrário} \end{cases}$$

   Defina a função `num_itR` que recebe como argumento um número natural n e devolve o número de vezes que f tem que se aplicada (recursivamente) a n até se atingir o número 1, i.e., devolve o número k tal que

$$\underbrace{f(f(\ldots f(n)))}_{k-\text{vezes}} = 1.$$

7. Defina a função `comb` que recebe como argumentos dois naturais `m` e `q`, com `m` $\geq$ `q`, tal que `comb(m,q)`$=$ $\begin{pmatrix} m \\ q \end{pmatrix}$, as combinações de `m`, `q` a `q`. Recorde que as combinações satisfazem a relação

$$\begin{pmatrix} m \\ q \end{pmatrix} = \begin{pmatrix} m-1 \\ q-1 \end{pmatrix} + \begin{pmatrix} m-1 \\ q \end{pmatrix}$$

   para $0 < q < m$.

8. Defina a função `pertenceR` que recebe como argumento um número n e um vector v e devolve verdadeiro se n pertence a v e falso caso contrário.

9. Defina a função `ocorrenciasR` que dado um número n e um vector v conta quantas vezes n ocorre v.

10. Defina a função `maxR` que recebe como argumento um vector v e devolve o máximo de v.

11. Defina a função `pos_maxR` que recebe como argumento um vector v e devolve a soma dos índices das posições onde o máximo de v ocorre.

12. Defina a função `splitR` que dado um vector v e um número n devolve um triplo $[a, b, c]$ tal que:
    - a é o número de elementos de v maiores que n,
    - b é o número de elementos de v iguais a n,
    - c é o número de elementos de v menores que n.

13. Um vector u diz-se prefixo de um vector v se o comprimento de u for menor ou igual do que o comprimento de v e $u(i) = v(i)$, para todo o $i = 1, \ldots, \mathit{length}(u)$. Defina a função `prefixoR` que recebe como argumento dois vectores u e v e devolve verdadeiro se u é prefixo de v e falso caso contrário.

14. Recorde o *algoritmo de pesquisa binária* para pesquisar um número num vector ordenado. Defina uma função `pesquisa_bin` que recebe como argumento um vector v de números ordenado (por ordem crescente) e um elemento x e que devolve verdadeiro se x pertence a v e falso caso contrário.

15. Defina a função `media_matrizR` que recebe como argumento uma matriz e devolve a média dos seus elementos.

16. Defina a função `max_matrizR` que recebe como argumento uma matriz a e devolve o máximo de a.