Instituto Politécnico de Leiria

Escola Superior de Tecnologia e Gestão

Matemática Discreta - Componente PL

EI (D+PL)

Ano letivo 2018/2019 - 2.º Sem. Ficha prática 4

Funções

1. A **função floor** aplicada a um número real x, $\lfloor x \rfloor$, representa o maior inteiro menor ou igual a x.

   Exemplos:

   - $\lfloor 1,3 \rfloor = 1$; $\lfloor 2,8 \rfloor = 2$; $\lfloor -2,3 \rfloor = -3$ $\lfloor 2 \rfloor = 2$.

2. A **função ceiling** aplicada a um número real x, $\lceil x \rceil$, representa o menor inteiro maior ou igual a x.

   Exemplos:

   - $\lceil 1,3 \rceil = 2$; $\lceil 3,8 \rceil = 4$; $\lceil -1,4 \rceil = -1$; $\lceil 2 \rceil = 2$.

3. A **função valor inteiro** de x, $int\,(x)$, representa a parte inteira do número x.

   Exemplos:

   - $int(2,4) = 2$; $int\,(-2,4) = -2$.

4. A **função valor absoluto** de x, $|x|$, representa o valor de x se $x \geq 0$ ou devolve o valor de $-x$ se $x < 0$.

   Exemplos:

   - $|-3| = 3$; $|5| = 5$.

5. **Função resto e aritmética modular**

   Seja k um número inteiro qualquer e seja m um inteiro positivo. Então $k \,(\mathrm{mod}\, m)$ (lê-se "k módulo m") representa o resto inteiro da divisão de k por m, ou seja, sendo

   $$k = mq + r\ , \qquad \text{com } 0 \leq r < m \text{ e } q \text{ inteiro},$$

   temos

   $$k \,(\mathrm{mod}\, m) = r.$$

   Exemplos:

   - $25 \,(\mathrm{mod}\, 7) = 4$; $25 \,(\mathrm{mod}\, 5) = 0$; $-26 \,(\mathrm{mod}\, 7) = 2$.

O termo mod é também usado na relação de congruência, que é definida e denotada da seguinte forma:

$$a \equiv b \,(\mathrm{mod}\, M) \text{ sse } M \text{ divide } a - b.$$

A notação $a \equiv b \,(\mathrm{mod}\, M)$ lê-se "a é congruente a b módulo M".

6. A **função fatorial** de n, $n!$, é definida por $n! = n \times (n-1) \times (n-2) \times ... \times 4 \times 3 \times 2 \times 1$.

   Exemplos:

   - $4! = 4 \times 3 \times 2 \times 1 = 24$; $\quad 7! = 7 \times 6 \times ... \times 1 = 5040$.

A função fatorial é um exemplo de uma função que pode ser definida recursivamente, dado que é possível defini-la à custa de si própria:

$$n! = n \times (n-1)!$$

Existem muitas outras funções matemáticas que podem ser definidas por recursividade, por exemplo as funções abordadas em alguns dos exercícios desta ficha.

Para implementar no Scilab as funções definidas previamente, usaremos os comandos:

`floor(x)`	Devolve o maior inteiro menor ou igual a x;
`ceil(x)`	Devolve o menor inteiro maior ou igual a x;
`int(x)`	Devolve a parte inteira do número x;
`abs(x)`	Devolve o valor absoluto do número x;
`pmodulo(k,m)`	Devolve o resto da divisão de k por m, onde o resto é um valor r tal que $0 \leq r < m$.
`factorial(n)`	Devolve o valor $n! = n \times (n-1) \times ... \times 3 \times 2 \times 1$.

Exemplos:

- `-->floor(2.5)`
  ans =
  2.

- `-->floor(-2.5)`
  ans =
  - 3.

- `-->ceil(2.5)`
  ans =
  3.

- `-->ceil(-2.5)`
  ans =
  -2.

- `-->pmodulo(15, 4)`
  `ans =`
  `3.`

- `-->pmodulo(3,9)`
  `ans =`
  `3.`

- ```
  -->pmodulo(-3, 9)
  ans =
  6.
  ```

- ```
  -->factorial(5)
  ans =
  120.
  ```

Existem ainda outros comandos que poderão ser úteis na construção de algoritmos:

`format(n)`	Quando `n` é inteiro, define o formato de saída de um número (número máximo de caracteres);
`part(palavra,n)`	Devolve a string que resulta da extração do caracter que se encontra na posição `n`
	da string `palavra`;
`part(palavra,i:j)`	Devolve a string que resulta da extração dos caracteres que se encontram entre a
	posição `i` e a posição `j` da string `palavra`;
`strsplit(palavra)`	Devolve um vetor de strings composto pelos carateres da string `palavra`;
`find(A==x)`	Devolve o índice da posição que x ocupa no vetor A;

Exemplos:

- ```
  -->x=%pi
  x =
  3.1415927

  -->format(20)
  -->x=%pi
  x =
  3.14159265358979312
  ```

- ```
  -->part("matemática",4)
  ans =
  e
  ```

- ```
  -->part("matemática",3:8)
  ans =
  temáti
  ```

- ```
  -->strsplit("mate")'
  ans =
  !m a t e !
  ```

- ```
  -->V=strsplit("mate")'; find(V=="t")
  ans =
  3.
  ```

- ```
  -->V=strsplit("mate")'; V($)
  ans =
  e.
  ```

Exercícios propostos

1. O armazenamento de dados em memória ou a transmissão de dados através de uma rede são representados através de uma "string"de bytes (1 byte = 8 bits). Usando comandos do Scilab:

   (a) determine quantos bytes são necessários para codificar 100 bits de dados.

   (b) construa uma função, designada por `conversor`, que permita determinar quantos bytes são necessários para codificar x bits de dados.

2. Fazendo uso das funções `floor` e `ceiling,` construa uma função chamada `inteiro` que determine a parte inteira do número real x.

3. Um ano bissexto possui 366 dias e sempre é múltiplo de 4. Porém, há casos especiais de anos que, apesar de múltiplos de 4, não são bissextos: são aqueles que também são múltiplos de 100 e não são múltiplos de 400. Construa uma função com o nome `bissexto` que permita determinar se um dado ano X é um ano bissexto.

4. Construa uma função com o nome `F` que permita obter os termos da sucessão de Fibonacci, para $n \geq 1$. Recorde que os números de Fibonacci F_n são os números que compõe a seguinte sucessão de números inteiros:

   $$0, 1, 1, 2, 3, 5, 8, 13, ...$$

   Em termos matemáticos, para $n \geq 1$, a sucessão de Fibonacci F_n é definida recursivamente por:

   $$F_n = \begin{cases} 1 & se \quad n = 1 \\ 1 & se \quad n = 2 \\ F_{n-1} + F_{n-2} & se \quad n \geq 3 \end{cases} .$$

   É uma sucessão que surge nas mais diversas áreas (ciências da computação, biologia, economia, entre outras).

5. Construa uma função designada por `soma_Fib` que receba um inteiro `n` e devolva a soma dos primeiros `n` termos da sucessão de Fibonacci.

6. Construa uma função designada por `termos_Fib` que receba um inteiro `s` e devolva um vetor linha `v` com o maior número de termos da sucessão de Fibonacci tais que a sua soma seja menor ou igual a `s`.
   (ou seja, `v`$= [F_1, F_2, ..., F_n]$ tal que $\sum_{k=1}^{n} F_k \leq s$ e $\sum_{k=1}^{n+1} F_k > s$)

7. O número de ouro, geralmente denotado por ϕ, é dado por

$$\phi = \frac{1+\sqrt{5}}{2}.$$

O número de ouro é encarado como um número misterioso por surgir em várias relações entre números. Exemplo disso é o que acontece com os números da sucessão de Fibonacci: o limite da razão $\frac{F_n}{F_{n-1}}$ é igual ao número de ouro, ou seja:

$$\begin{aligned}
&\frac{F_2}{F_1} = 1\\
&\frac{F_3}{F_2} = 2\\
&\frac{F_4}{F_3} = 1.5\\
&\frac{F_5}{F_4} = 1.6666...\\
&...\\
&\lim_{n\to\infty} \frac{F_n}{F_{n-1}} = \phi
\end{aligned}$$

Usando comandos do Scilab:

(a) determine uma aproximação do número de ouro ϕ com um erro inferior ou igual a 10^{-10}, através da sucessão de Fibonacci.

(b) apresente um gráfico que ilustre esta aproximação.

8. Construa uma função com o nome `Ackermann` que, para valores x e y inteiros não negativos, devolva a imagem

$$Ackermann(x,y) = \begin{cases} y+1 & se \quad x = 0 \\ Ackermann(x-1,1) & se \quad x > 0 \text{ e } y = 0 \\ Ackermann(x-1, Ackermann(x, y-1)) & se \quad x > 0 \text{ e } y > 0 \end{cases}.$$

9. Prova-se por indução matemática que

$$\sum_{k=1}^{n} (k!k) = (n+1)! - 1.$$

Utilize o Scilab para verificar a igualdade quando $n = 10, n = 100$ e $n = 500$. O que se pode observar?

10. O dia da semana em que se celebra o Natal num dado ano entre 2001 e 2099 é o número natural $d \in \{0,1,2,3,4,5,6\}$ tal que

$$d \equiv_7 50 + ANO + ((SÉCULO - 1) \div 4) + (ANO \div 4) - 2 \times (SÉCULO - 1),$$

considerando que 0 representa domingo, 1 representa segunda, 2 representa terça, etc., onde se o ano em causa é $XYZW$ então $ANO = ZW$, e $A \div B$ é o quociente da divisão inteira de A por B. Calcule o dia da semana em que se celebra o Natal no ano

(a) 2017 (b) 2023 (c) 2050

11. O número de série de uma nota de euro verdadeira é constituído por uma letra seguida de onze algarismos.

A letra representa o país de emissão da nota e tem um valor numérico associado, que designaremos por β. Na tabela seguinte encontram-se listados os valores numéricos associados a alguns desses países:

PAÍS	**LETRA**	**VALOR**
Luxemburgo	R	1
Itália	S	2
Irlanda	T	3
França	U	4
Portugal	M	5
Áustria	N	6

Os dez algarismos que se seguem à letra identificam a nota e o último algarismo é o digito de controlo d, que é determinado de modo que

$$\beta + x_1 + x_2 + ... + x_9 + x_{10} + d \equiv 0 \bmod 9$$

onde x_j é o algarismo que se encontra na posição j do número que se segue à letra (por exemplo, x_1 é o primeiro número a seguir à letra, lido da esquerda para a direita). Usando comandos do Scilab:

(a) implemente um algoritmo que permita verificar se o número de série de uma nota de euro proveniente de um dos países listados na tabela é válido (sugestão: comece por solicitar a introdução do número de série em formato string).

(b) verifique se o número de série da nota da figura, "M50027558701", é válido.

12. As congruências são usadas todos os dias nos algoritmos de controlo que, através de um (ou mais) dígito(s) de controlo, detectam erros em sequências de números, tais como as usadas no cartão de cidadão, num cartão de crédito ou num produto com código de barras.

A maioria dos números de cartões de crédito é validada através do algoritmo de Luhn:

- retira-se o último dígito e inverte-se a posição dos restantes dígitos;
- substitui-se cada dígito em posição ímpar pelo seu dobro, exceto quando se obtém um número maior que 9, caso em que primeiro se subtrai 9 e depois se substitui;
- adicionam-se todos os dígitos, incluindo o que se removeu no início;

O valor x assim obtido tem de ser congruente com 0 módulo 10.

Usando comandos do Scilab, implemente o algoritmo descrito e use-o para verificar quais dos seguintes números de cartão de crédito é válido:

(A) 4012888888881882 (B) 378282246310004 (C) 371449635398431.

13. Descodifique a mensagem MLZ, usando a função $D(p) = (3p + 3) \bmod 23$ e identificando as 23 letras do alfabeto pelos inteiros $0, 1, 2, \ldots, 22$ da seguinte forma:

A	B	C	D	E	F	G	H	I	J	L	M	N	O	P	Q	R	S	T	U	V	X	Z
↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22