POLITÉCNICO DE LEIRIA
ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO

Instituto Politécnico de Leiria

Escola Superior de Tecnologia e Gestão

Matemática Discreta - Componente PL

EI (D)

Ano letivo 2023/2024 - 2. ^{o} Sem. Ficha prática 4

Funções matemáticas - Teoria de números

Existem várias funções Matemáticas, em particular na área de Teoria de Números, que são bastante relevantes na implementação de algoritmos em Teoria da Computação.

1. A **função floor** aplicada a um número real x, $\lfloor x \rfloor$, devolve 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, $\lceil x \rceil$, devolve 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, $int\,(x)$, devolve a parte inteira do número x.

   Exemplos:

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

4. A **função resto** na aritmética modular devolve o resto de uma divisão inteira:

   Seja k um número inteiro qualquer e seja m um inteiro positivo. Então $k \pmod m$ (que se lê "k módulo m") devolve o resto inteiro r da divisão inteira de k por m, ou seja,

   $$k \pmod m = r$$

   sendo

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

   Exemplos:

   - $25 \pmod 7 = 4$; $25 \pmod 5 = 0$; $-26 \pmod 7 = 2$.

O termo mod é também usado na relação de congruência. Os inteiros a e b dizem-se congruentes módulo m (onde m é um inteiro positivo) se e só se m divide $a - b$, ou seja,

$$a \equiv b \pmod{m} \text{ sse } m \text{ divide } a - b.$$

6. A **função fatorial** de n, $n!$, para n inteiro não negativo, devolve o inteiro

$$n! = n \times (n-1) \times (n-2) \times ... \times 4 \times 3 \times 2 \times 1.$$

Exemplos:

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

A função fatorial é uma função recursiva, 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, tal como podemos verificar ao longo da resolução dos exercícios desta ficha.

As funções matemáticas definidas previamente podem ser implementadas através das instruções dadas na tabela seguinte:

Funções Matemáticas - Teoria de Números	
`math.floor(x)`	Devolve o maior inteiro menor ou igual a x
`math.ceil(x)`	Devolve o menor inteiro maior ou igual a x
`int(x)`	Devolve a parte inteira do número x
`k%m`	Devolve o resto da divisão inteira de k por m, onde m é inteiro positivo, k um inteiro e o resto é um valor r tal que $0 \leq r < m$
`k//m`	Devolve o quociente da divisão inteira de `k` por `m` para k e m inteiros com $m > 0$
`math.factorial(n)`	Devolve o valor $n! = n \times (n-1) \times ... \times 3 \times 2 \times 1$

Exemplos:

```
In [1]: import math
        math.floor(2.5)
Out[1]: 2

In [2]: math.floor(-2.5)
Out[2]: -3

In [3]: math.ceil(2.5)
Out[3]: 3

In [4]: math.ceil(-2.5)
Out[4]: -2

In [5]: int('11')
Out[5]: 11

In [6]: int(12.1)
Out[6]: 12
```

```
In [7]: 15%4
Out[7]: 3

In [8]: 3%9
Out[8]: 3

In [9]: -3%9
Out[9]: 6

In [10]: 15//4
Out[10]: 3

In [11]: -3//9
Out[11]: -1

In [12]: math.factorial(5)
Out[12]: 120
```

```
In [13]: math.factorial(50)
Out[13]: 30414093201713378043612608166064768844377641568960512000000000000
```

Uma `string` é um tipo de dados constituído por uma sequência ordenada de caracteres entre aspas (simples, duplas ou triplas) que não podem ser substituídos (não é possível fazer algo como string[i]='k'). No entanto, é possível converter maiúsculas em minúsculas ou vice-versa. A próxima tabela ilustra algumas das operações com strings que são possíveis de efetuar:

Operações com strings	
`print(f"Texto {x} texto {y}.")`	Escreve na consola "texto" substituindo as variáveis pelos respetivos valores
`string[n]`	Devolve o caractere da `string` que se encontra com índice `n` (`string`[0] devolve o primeiro caractere da `string`)
`string[i:j]`	Devolve a string que resulta da extração dos caracteres indexados de `i` a `j-1`
`len(string)`	Devolve o número de caracteres na `string`

Exemplos:

```
In [14]: i=46
         x=math.pi
         print(f'Na iteração {i} o valor de x é {x}')
         Na iteração 46 o valor de x é 3.141592653589793

In [15]: print(f'O valor de Pi é {x:.30f}')
         O valor de Pi é 3.141592653589793115997963468544

In [16]: print("o Valor de Pi é", format(x, "0.5f"))
         o Valor de Pi é 3.14159
```

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). Crie um script em Python que:

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

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

2. Fazendo uso dos comandos `floor` e `ceil`, construa uma função chamada `inteiro` que devolva a parte inteira do número real x.

3. Um ano bissexto possui 366 dias e é sempre 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 o termo de ordem n 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: 1, 1, 2, 3, 5, 8, 13, ...
   Em termos matemáticos, a sucessão de Fibonacci F_n é definida recursivamente por:

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

   Esta sucessão surge nas mais diversas áreas (ciências da computação, biologia, economia, ...).

5. Construa uma função designada por `soma_Fib` que receba um inteiro `k` e devolva a soma dos primeiros `k` 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:

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

   Crie um script em Python que:

   (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 o valor

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

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

$$d \equiv_7 50 + A + ((SÉCULO - 1)//4) + (A//4) - 2 \times (SÉCULO - 1),$$

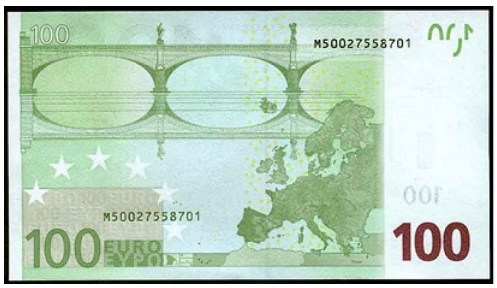
considerando que 0 representa domingo, 1 representa segunda, 2 representa terça, etc, onde:

- se o ano em causa é $XYZW$ então deverá considerar-se $A = ZW$
- $p//q$ representa o quociente da divisão inteira de p por q.

Crie um script em Python que permita determinar em que dia da semana se celebra o Natal no ano:

(a) 2017 (b) 2032 (c) 2050

10. 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

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

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). Crie um script em Python que:

(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.

11. Uma instituição de ensino superior atribui o n.º de estudante $d_1d_2d_3d_4d_5d_6d_7$ tendo em atenção que, lendo o número da esquerda para a direita:

- o primeiro dígito d_1 diz respeito ao campus que o estudante frequenta;
- os dois dígitos seguintes d_2d_3 dizem respeito ao ano em que o estudante ingressou no ensino superior;
- os quatro dígitos seguintes $d_4d_5d_6d_7$ revelam o dia da semana em que o processo do estudante foi criado, de acordo com o seguinte:
  (a) Se $d_4 + d_5 + d_6 + d_7 \equiv_5 D$, com $D \in \{0, 1, 2, 3, 4\}$, então D indica o dia da semana em que o processo foi criado, considerando que 0 representa segunda, 1 representa terça, 2 representa quarta, 3 representa quinta e 4 representa sexta.

(a) Construa uma função designada por $Identificador$ que, dado o n.º de um estudante, identifique o campus que este frequenta, o ano letivo em que entrou no ensino superior e o dia da semana em que o seu processo foi criado. Por exemplo, $Identificador(2175243)$ deverá devolver o output

"Campus 2 - Ano letivo 17/18 - Sexta-feira".

(b) Supondo que este algoritmo é usado no Politécnico de Leiria, identifique em que dia da semana foi criado o seu processo, aplicando a função $Identificador$ ao seu número de estudante.

12. As congruências são usadas todos os dias nos algoritmos de controlo que, através de um (ou mais) dígitos de controlo, detetam erros em sequências de números, tais como as usadas no cartão de cidadão, em produtos com código de barras ou em cartões de crédito, entre outros.

    O algoritmo de Luhn é um exemplo da utilização das congruências na validação destes números:

- Começando com o penúltimo dígito e seguindo para a esquerda, alternadamente, duplicar o valor de cada um dos dígitos. Se, ao fazer esta operação, se obtiver um resultado maior do que 9, adicionam-se os valores dos dígitos que compõem esse resultado (por exemplo, se o dobro for 16, o resultado final será $1 + 6 = 7$).
- Adicionam-se todos os resultados obtidos na etapa anterior com o valor de todos os dígitos que não foram alterados.

  Para que o algoritmo valide o número do cartão de crédito, o valor obtido na soma final tem de ser congruente com 0 módulo 10.

  Observe o exemplo seguinte:

7	9	9	2	7	3	9	8	7	1	3
	x2		x2		x2		x2		x2	
	18		4		6		16		2	
7	9	9	4	7	6	9	7	7	2	3

7 + 9 + 9 + 4 + 7 + 6 + 9 + 7 + 7 + 2 + 3 = 70

(a) Uma empresa de crédito tem cartões de crédito com 14 dígitos. Se os primeiros dígitos forem 5027811020103 qual deverá ser o último dígito de modo que o número do cartão seja válido.

(b) Crie um script em Python que permita implementar o algoritmo descrito e use-o para verificar quais dos seguintes números de cartão de crédito serão válidos:

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

(c) Verifique que uma alteração num dos dígitos de um número de cartão válido apresenta erro.

(d) Analise que tipo de erros não serão detetados por este algoritmo. Dê um exemplo de um desses tipos de erros.

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