[PT] modularização

A - Média de números com menos de X algarismos

a) Construa um módulo para determinar a quantidade de dígitos de um número dado como parâmetro.

b) Utilizando o módulo anterior faça um programa que leia uma sequência de números inteiros com menos de N algarismos e determine a sua média. O valor de N deve ser o primeiro número a ser lido. A sequência termina quando for introduzido um número com N ou mais algarismos.
A média deve ser visualizada com 2 casas decimais.

c) Adapte o programa para terminar no máximo ao fim de K números introduzidos.

OBS: Utilize uma constante para definir o número máximo de elementos da sequência (K=5).

[EN] modularization

A - Average of numbers with less than N digits

a) Implement a module to determine the number of digits of a given number as a parameter.

b)Write a program to read a sequence of integers with less than N digits and average these numbers.
The value of N must be the first number read.

The program must first read the maximum number of digits (N) the numbers can have.
Then read integers smaller than N.
The program ends when a number with more than N digits is entered or when more than K numbers are read.

The average should be displayed to 2 decimal places.

Use a constant to define the maximum number of elements that can be read (K = 5).

Example1:

Input	Output
4 123 5 41 1000	56.33

Example2:

Input	Output
5 1234 23 12345678	628.50

[PT] modularização

B - Gráfico de classificações

Faça um programa que represente sob a forma de gráficos de barras, o número de positivas e negativas dos alunos de uma turma a um conjunto de disciplinas.
O programa deverá começar por ler o nº de alunos da turma e o nº de disciplinas e, para cada disciplina, ler o nome da disciplina e o nº de alunos aprovados.
Deve implementar um módulo para imprimir a informação acerca de uma disciplina.
O output produzido deverá ter o seguinte aspeto, em que cada aluno será representado por um asterisco (*):

```
Subject: Portuguese
- Approved: ******
- Failed: ****
Subject: Math
- Approved: *******
- Failed: ***
```

[EN] modularization

B - Ranking chart

Write a program to display, in the form of bar graph, the number of students in a class that scored "Approved" and "Failed" on a set of subjects.

The program should start by asking the number of students in the class and the number of subjects and, for each subject will ask the name of the subject and the number of students with approval.

Implement a module to print information about a subject.

The output should look like this:

```
Subject: Portuguese
- Approved: ******
- Failed: ****
Subject: Math
- Approved: *******
- Failed: ***
```

Example1:

Input	Output
10 2 Portuguese 6 Math 7	Subject: Portuguese - Approved: ****** - Failed: **** Subject: Math - Approved: ******* - Failed: ***

Example2:

Input	Output
5 5 PT 1 MT 0 CI 4 IN 5 FR 3	Subject: PT - Approved: * - Failed: **** Subject: MT - Approved: - Failed: ***** Subject: CI - Approved: **** - Failed: * Subject: IN - Approved: ***** - Failed: Subject: FR - Approved: *** - Failed: **

Example3:

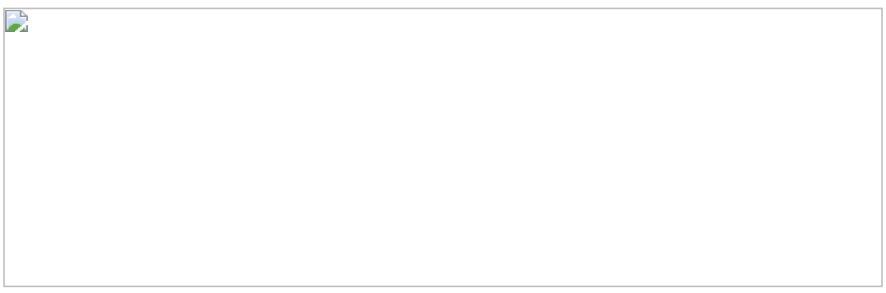
Input	Output
10 0	

[PT] modularização

C - Calcular os ângulos de um triângulo

a) Implemente um método que calcule um ângulo interno de um triângulo, sendo dadas as medidas dos três lados desse triângulo. O método deve receber por parâmetro as medidas dos três lados do triângulo e retornar o ângulo calculado em graus.

b) Faça um programa que peça as medidas de três lados, verifique se elas são válidas e se é possível formar um triângulo. Em caso afirmativo calcule todos os ângulos internos desse triângulo. Para isso invoque três vezes o método desenvolvido na alínea anterior.



Os resultados deverão ser apresentados em linhas separadas e os valores dos ângulos em graus, com duas casas decimais. Caso não seja possível formar um triângulo, a mensagem deverá ser: "impossible".
O resultado deverá apresentar o seguinte formato:

```
a=2.00
b=3.00
```

```
c=4.00
ang(a,b)=104.48
ang(a,c)=46.57
ang(b,c)=28.96
```

[EN] modularization

C - Calculate the angles of a triangle

a) Implement a method that calculates an inner angle of a triangle. The method must take as parameter the measurements on the three sides of the triangle and returns the calculated angle in degrees.

b) Write a program that asks for the three-sided measurements of a triangle, check if they are valid and if it is possible to form a triangle. If so, calculate all 3 internal angles of this triangle. To do this, invoke the method developed in the previous paragraph three times.

Results should be displayed in separate lines and angle values in degrees with two decimal places.
If it is not possible to form a triangle, the message "impossible" should be displayed.

The result should look like this:

```
a=2.00
b=3.00
c=4.00
ang (a,b)=104.48
ang (a,c)=46.57
ang (b,c)=28.96
```

Example1:

Input	Output
2 3 4	a=2.00 b=3.00 c=4.00 ang(a,b)=104.48 ang(a,c)=46.57 ang(b,c)=28.96

Example2:

Input	Output
3 4 5	a=3.00 b=4.00 c=5.00 ang(a,b)=90.00 ang(a,c)=53.13 ang(b,c)=36.87

Example3:

Input	Output
5 5 7	a=5.00 b=5.00 c=7.00 ang(a,b)=88.85 ang(a,c)=45.57 ang(b,c)=45.57

Example4:

Input	Output
5 5 10	impossible

Example5:

Input	Output
3 4 -2	impossible

Example6:

Input	Output
6 6 6	a=6.00 b=6.00 c=6.00 ang(a,b)=60.00 ang(a,c)=60.00 ang(b,c)=60.00

Example7:

Input	Output
2.5 2.5 5.0	impossible

Example8:

Input	Output
0 1 2	impossible

Example9:

Input	Output
0 0 0	impossible

[PT] modularização

D - Combinações e permutações

Faça um programa que calcule a quantidade de combinações e permutações possíveis de um conjunto de elementos.
Deve introduzir a quantidade total de elementos (m) e a quantidade a agrupar (n).
As fórmulas são as seguintes:

- Combinações de m elementos, n a n (m>=n).

$$C(m,n) = \frac{m!}{n!\ (m-n)!}$$

-Permutações de m elementos, n a n (m>=n).

$$P(m,n) = \frac{m!}{(m-n)!}$$

Cada resultado deverá aparecer em linhas separadas e no seguinte formato:
C(5,2)=10
P(5,2)=20

OBS: Implemente os seguintes métodos:
factorial()
combinations()
permutations()

[EN] modularization

D - Combinations and permutations

Write a program that calculates the number of possible combinations and permutations of a set of elements. You must enter the total quantity of elements (m) and the subset size to group (n).
The formulas are as follows:

- Combinations of m elements, in subsets of size n (m> = n).

$$C(m, n) = \frac{m!}{n!\,(m-n)!}$$

Permutations of m elements, in subsets of size n (m> = n).

$$P(m, n) = \frac{m!}{(m-n)!}$$

Each result should appear on separate line and in the following format:
C(5,2)=10
P(5,2)=20

NOTE: Implement the following methods:
factorial ()
combinations ()
permutations ()

Example1:

Input	Output
5 2	C(5,2)=10 P(5,2)=20

Example2:

Input	Output
7 5	C(7,5)=21 P(7,5)=2520

Example3:

Input	Output
5 5	C(5,5)=1 P(5,5)=120

Example4:

Input	Output
2 3	

[PT] modularização

E - Algarismos em posições comuns

a) Implemente um módulo que dados dois números inteiros positivos retorne a quantidade de dígitos comuns nas mesmas posições.

b) Elabore um programa que leia N pares de valores inteiros positivos, sendo N introduzido pelo utilizador e validado. Após a leitura dos N pares de valores o programa deve apresentar o par que tiver mais dígitos comuns nas mesmas posições. No caso de haver mais do que um par com a mesma quantidade de dígitos em comum, deve ser apresentado o último par encontrado.
O resultado deve conter apenas o par em causa no formato "numero1/numero2".
No caso de não haver nenhum par de números que tenha algarismos em comum deve ser apresentada a mensagem: "no results".

[EN] modularization

E - Digits at common positions

a) Implement a module that receives two positive integers and returns the number of common digits in the same positions.

b) Write a program to read N pairs of positive integers. The value of N is entered by the user.
After reading the N value pairs, the program must display the pair that has the most common digits at the same positions. If there is more than one pair with the same number of digits in common, the last pair found must be displayed.

The result should contain only the pair in question in the following format:
"number1/number2".

If there are no pair of numbers that have digits in common, the message "no results" should be displayed.

Example1:

Input	Output
4 12345 345 13579 12529 123 456789 123 456	13579/12529

Example1:

Input	Output
3 123 456 100 999 12345 21453	no results

[PT] modularização

F - Volume de sólidos de revolução

Faça um programa que permita determinar volumes de sólidos de revolução (ccylinder, cone e sphere).
Para cada sólido será introduzido o tipo de sólido e as respetivas dimensões (raio e altura, se necessário).
Cada resultado deverá ser apresentado em linhas separadas e com duas casas decimais. O programa termina quando o tipo de sólido for a palavra "end".
Implemente um método distinto para calcular o volume de cada sólido.

[EN] modularization

F - Volume of revolution solids

Write a program to determine volumes of revolution solids. Consider only these types of solids: cylinder, cone and sphere.
For each solid, the type of solid and its dimensions (radius and height if required) will be entered.

The program must be able to function repeatedly until the word "end" is entered as a solid type.

Each result should be presented on separate lines and with two decimal places.

Implement a separate method to calculate the volume of each solid.

NOTE:
Volume sphere = $4/3\ \pi\ R^3$
Volume cylinder = $\pi\ R^2$ Height
Volume cone = $1/3\ \pi\ R^2$ Height

Example1:

Input	Output
cone 2.5 3 sphere 3.1 end	19.63 124.79

Example2:

Input	Output
cone 2.5 3 sphere 3.1 cylinder 5.2	19.63 124.79 832.50 873.65

9.8 cylinder 5.3 9.9 end	

Example3:

Input	Output
end	

Example4:

Input	Output
cone 3.75 5.6 sphere 9.0 cylinder 5.2 9.8 cylinder 5.3 9.9 sphere 1.5 sphere 11.1 cone 11.11 10.01 end	82.47 3053.63 832.50 873.65 14.14 5728.72 1293.87

[PT] modularização

G - Números de Armstrong

Um número de Armstrong possui a seguinte característica: a soma dos algarismos elevados à quantidade de algarismos é igual ao próprio número.
Por exemplo:
2 = 2^1 (1 algarismo => somar todos os algarismos elevados a 1)
407 =4^3 + 0^3 + 7^3 (3 algarismos => somar todos os algarismos elevados a 3)
1634 = 1^4 + 6^4 + 3^4 + 4^4 (4 algarismos => somar todos os algarismos elevados a 4)

a) Construa um método para verificar se um número inteiro é um número de Armstrong.

b) Faça um programa que imprima todos os números de Armstrong até um valor N inserido pelo utilizador.

[EN] modularization

G - Armstrong Numbers

An Armstrong number has the following characteristic: the sum of the digits raised to the number of digits equals the number itself.
For example:
2 = 2 ^ 1 (1 digit => sum all digits raised to 1)
407 = 4 ^ 3 + 0 ^ 3 + 7 ^ 3 (3 digits => add all digits raised to 3)
1634 = 1 ^ 4 + 6 ^ 4 + 3 ^ 4 + 4 ^ 4 (4 digits => add up to 4 digits)

a) Implement a method to verify that an integer is an Armstrong number.

b) Write a program to display all Armstrong numbers up to a user-entered value N.

Example1:

Input	Output
200	0 1 2 3 4 5 6 7 8

	9 153

Example2:

Input	Output
100000	0 1 2 3 4 5 6 7 8 9 153 370 371 407 1634 8208 9474 54748 92727 93084

[PT] modularização

H - Palíndromo

Um palíndromo é uma sequência que é lida da mesma forma, da frente para trás e de trás para a frente.

a) Construa um método que verifique se um número inteiro é ou não um palíndromo.

b) Faca um programa que leia uma sequência de números inteiros e termine quando for introduzido um número palíndromo ou quando tiver analisado um máximo de tentativas (5) sem o encontrar.

Deve ser visualizada a mensagem "palindrome" ou "attempts exceeded"

[EN] modularization

H - Palindrome

A palindrome is a sequence that reads the same backward as forward.
a) Implement a method that checks whether or not an integer is a palindrome.

b) Write a program to find a palindrome in a sequence of user-entered integers. The program should read a number, verify if it is a palindrome and, if so, the message "palindrome" should be displayed, otherwise another number should be read. The program ends when it finds a palindrome or the size of the sequence exceeds a threshold value (5). The threshold value is the maximum number of attempts (5) without finding a palindrome. The message "attempts exceeded" should be displayed if the threshold value is exceeded.

Example1:

Input	Output
213 33331 42124	palindrome

Example2:

Input	Output
213 33331 421241 1112 10002	attempts exceeded

[PT] modularização

I - Elemento de Fibonacci

Elabore um programa para determinar se um número inteiro introduzido pelo utilizador é um elemento da sucessão de Fibonacci.
Na sucessão de Fibonacci, o primeiro termo é zero (0), o segundo é um (1) e qualquer um dos outros termos é a soma dos dois anteriores (0,1,1,2,3,5,8,13,21,34,55,89,144,...).

Deve ser visualizada a mensagem "is a Fibonacci number" ou "is not a Fibonacci number" caso o número inserido pertença à sucessão de Fibonacci ou não, respetivamente.

[EN] modularization

I - Fibonacci Number

Write a program to determine if a user-entered integer is a number of the Fibonacci sequence.
In Fibonacci's succession, the first term is zero (0), the second is one (1), and any of the other terms is the sum of the previous two (0,1,1,2,3,5,8,13, 21,34,55,89,144, ...).

The message "is a Fibonacci number" or "is not a Fibonacci number" should be displayed if the number entered belongs to the sequence of Fibonacci or not, respectively.

Example1:

Input	Output
13	is a Fibonacci number

Example2:

Input	Output
317811	is a Fibonacci number

Example3:

Input	Output
317812	is not a Fibonacci number

Example4:

Input	Output
20	is not a Fibonacci number