

# Paradigmas de Programação

Prof. Maicon R. Zatelli

Python - Uma Linguagem Multiparadigma

Universidade Federal de Santa Catarina  
Florianópolis - Brasil

# Python

Desenvolvida por Guido van Rossum (1991)

Multiparadigma:

- Orientado a Objetos
- Imperativo
- Funcional
- Procedural (estruturado)
- Reflectivo (capacidade de um programa inspecionar e modificar a si próprio durante a execução )

Interpretada

Multiplataforma

Software livre e código aberto

<https://www.python.org/>

# Python - Variáveis

## Tipagem dinâmica

- Interpretador infere o tipo da variável

```
n, m = 2, 5  
print("Soma: ", n + m)
```

\* Tipagem estática: C, Java

## Tipagem forte

- Interpretador avalia as expressões e não faz conversões automáticas entre tipos não compatíveis

```
x, y = 1, "Floripa"  
print("Soma: ", x + y)
```

\* Tipagem fraca: PHP e JavaScript

Tente executar os códigos Python em:

[https://www.tutorialspoint.com/execute\\_python3\\_online.php](https://www.tutorialspoint.com/execute_python3_online.php)

# Python - Sintaxe

```
if x >= 100 or x > 15 and x < 25:  
    print("a")  
elif not (x < 10):  
    print("b")  
else:  
    print("c")
```

# Python - Sintaxe

```
i = 0
while i >= 10:
    print(i)
    i += 2
```

# Python - Sintaxe

```
for i in range(10):  
    print(i)
```

```
for i in range(1,11):  
    print(i)
```

# Python - Sintaxe

```
#cria um vetor preenchido com ''nada'' de n posições
vetor = [None] * n
for i in range(n):
    vetor[i] = [] #cria uma lista para cada posição do vetor

i = 0
while vetor[i] != []: #testa vetor[i] não é vazia
    ...

#para cada elemento k da lista da posição i do vetor
for k in vetor[i]:
    ...
```

# Compilando e Executando

Faça o download do Python e instale em seu computador:

- <https://www.python.org/downloads/>

Para compilar e executar use o seguinte comando:

- `python3 arquivo.py`

Para compilar e executar informando uma entrada:

- `python3 arquivo.py < entrada.txt`

Há várias versões do Python. A sintaxe apresentada aqui utiliza Python 3.

Compilador online Python3:

[https://www.tutorialspoint.com/execute\\_python3\\_online.php](https://www.tutorialspoint.com/execute_python3_online.php)



# Compilando e Executando

O cliente python.

```
[maicon@localhost Python]$ python3
Python 3.6.6 (default, Jul 19 2018, 14:25:17)
[GCC 8.1.1 20180712 (Red Hat 8.1.1-5)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> 5+3
8
>>> print ((lambda a, b: "sim" if a > b else "nao")(3, 2))
sim
>>> 5 + 3
8
>>>
```

# Entrada de Datos

---

```
10
20
*
A
```

---

```
10 * 34
```

---

```
10 20 30 40 50 0
```

---

```
a = int(input())
b = int(input())
c = input()
d = input()
```

---

```
a, b, c = input().split()
a = int(a)
c = int(c)
```

---

```
lista = input().split()
for i in range(len(lista)):
    lista[i] = int(lista[i])
```

# Exemplo 1

10 30 40 50 10 2 4

```
lista = input().split()
x = 0
for i in range(len(lista)):
    x += int(lista[i])
print(x)
```

## Exemplo 2

10 30 40 50 10 2 4

```
lista = input().split()
x = 0
for i in lista:
    x += int(i)
print(x)
```

## Exemplo 3

10,30,40,50,10,2,4

```
lista = input().split(",")  
x = 0  
for i in lista:  
    x += int(i)  
print(x)
```

## Exemplo 4

4 / 2

5 \* 2

```
a,b,c = input().split()
a = int(a)
c = int(c)
if b == "/":
    print(a / c)
else:
    print(a * c)
```

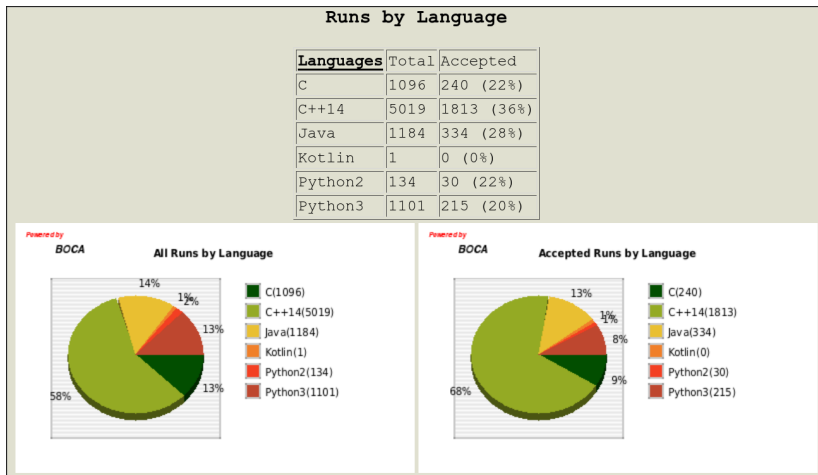
## Exemplo 5

$$4 / 2 = 2$$

$$50 * 2 = 8$$

```
a,b,c,d,e = input().split()
a = int(a)
c = int(c)
e = int(e)
if b == "/":
    resposta = a / c
else:
    resposta = a * c
if resposta == e:
    print("OK")
else:
    print("Erro")
```

# Python em Competições de Programação



<http://maratona.ime.usp.br/prim-fase18/stat.html>



# Juiz Online - URI Online Judge

The screenshot shows the URI Online Judge website. At the top, there is a navigation bar with links: HOME, PROFILE, NEWS, ACADEMIC, CONTESTS, FORUM, PROBLEMS, SUBMISSIONS, RANKS, and SIGN OUT. Below the navigation bar, the main content area is divided into several sections. On the left, there is a sidebar with the URI logo, a trophy icon labeled 'TOP 20', and a list of top performers. The main area features a 'CATEGORIES' section with a magnifying glass icon and a prompt to 'SELECT ONE OF THE 9 BIG CATEGORIES OF PROBLEMS TO BEGIN SOLVING.' Below this, there are nine category cards, each with a large number, a title, a description, and a count of problems. The categories are: 1. BEGINNER (297 PROBLEMS), 2. AD-HOC (693 PROBLEMS), 3. STRINGS (119 PROBLEMS), 4. DATA STRUCTURES AND LIBRARIES (133 PROBLEMS), 5. MATHEMATICS (263 PROBLEMS), 6. PARADIGMS (174 PROBLEMS), 7. GRAPH (284 PROBLEMS), 8. COMPUTATIONAL GEOMETRY (68 PROBLEMS), and 9. SQL (34 PROBLEMS). Additionally, there are two more cards: 'LIST ALL' (1821 PROBLEMS) and 'ORIGINS' (155 ORIGINS). The bottom of the page shows the date and time: 8/2/18, 6:03 PM.

URI ONLINE JUDGE PROBLEMS & CONTESTS

**TOP 20**

Maycon Alves  
Gabriel Duarte  
Gustavo Polcario  
gabriel Leonardo ...  
Erick Leonardo de ...  
Lutz Joaquim  
Sleeping  
Luis Fernando Ver ...  
Mteheh (UBAT)  
Thalyson Nepomuceno  
Vyllian Brico  
Diego Rangel  
Ricardo Oliveira  
Renan Tashiro  
Thaddeus Hieronymus  
Eduardo Theodoro ...  
Redsifo Riyosi Goya  
[Traveling Ballos...  
Marcello Marques  
The Install vizard

8/2/18, 6:03 PM

**CATEGORIES**

SELECT ONE OF THE 9 BIG CATEGORIES OF PROBLEMS TO BEGIN SOLVING.

**1 BEGINNER**  
Basic problems for anyone who has just started to program.  
297 PROBLEMS

**2 AD-HOC**  
Simulation Problems, Dates, Games and general Ad-Hoc...  
693 PROBLEMS

**3 STRINGS**  
Palindromes, Frequency, Ad-Hoc, LCS, String Manipulation...  
119 PROBLEMS

**4 DATA STRUCTURES AND LIBRARIES**  
Queue, Stack, Set, Map, Set...  
133 PROBLEMS

**5 MATHEMATICS**  
Number Theory, Prime Numbers, Combinatorics, BigInteger...  
263 PROBLEMS

**6 PARADIGMS**  
Dynamic Programming, Binary Search, Greedy, Backtracking...  
174 PROBLEMS

**7 GRAPH**  
Flood Fill, MST, SSSP, DAG, Maximum Flow, Tree...  
284 PROBLEMS

**8 COMPUTATIONAL GEOMETRY**  
Points and Lines, Polygon...  
68 PROBLEMS

**9 SQL**  
Query Languages: Select, Insert, Update, Create  
34 PROBLEMS

**LIST ALL**  
All the problems of the website in one place.  
1821 PROBLEMS

**AUTHORS**  
All available problems grouped by author.  
230 AUTHORS

**ORIGINS**  
All available problems grouped by contests or events.  
155 ORIGINS

<https://www.urionlinejudge.com.br/>

# Primeiro Problema - Extremamente Básico (1001)

Leia 2 valores inteiros e armazene-os nas variáveis **A** e **B**. Efetue a soma de **A** e **B** atribuindo o seu resultado na variável **X**. Imprima **X** conforme exemplo apresentado abaixo. Não apresente mensagem alguma além daquilo que está sendo especificado e não esqueça de imprimir o fim de linha após o resultado, caso contrário, você receberá *"Presentation Error"*.

## Entrada

A entrada contém 2 valores inteiros.

## Saída

Imprima a mensagem "X = " (letra X maiúscula) seguido pelo valor da variável **X** e pelo final de linha. Cuide para que tenha um espaço antes e depois do sinal de igualdade, conforme o exemplo abaixo.

Exemplos de Entrada	Exemplos de Saída
10 9	X = 19
-10 4	X = -6
15 -7	X = 8

# Primeiro Problema - Extremamente Básico (1001)

## Solução em Python

```
# -*- coding: utf-8 -*-  
a = int(input())  
b = int(input())  
print("X = %s" % (a + b))
```

## Pontos importantes

- A função **print()** já imprime com uma quebra de linha
- Note os espaços entre **X**, **=** e **%s**
- Lembre de incluir como primeira linha: **# -\*- coding: utf-8 -\*-**

# Primeiro Problema - Extremamente Básico (1001)

## Testando a Solução

### Execução - entrada por redirecionamento de arquivo

```
[maicon@inf58-140 testes]$ python3 teste.py < entrada.txt  
X = -6  
[maicon@inf58-140 testes]$
```

- Conteúdo do arquivo entrada.txt:

-10

4

# Primeiro Problema - Extremamente Básico (1001)

## Testando a Solução

### Execução - entrada por redirecionamento de arquivo

```
[maicon@inf58-140 testes]$ python3 teste.py < entrada.txt  
X = 19  
[maicon@inf58-140 testes]$
```

- Conteúdo do arquivo entrada.txt:

10

9

# Primeiro Problema - Extremamente Básico (1001)

## Testando a Solução

### Execução - entrada por redirecionamento de arquivo

```
[maicon@inf58-140 testes]$ python3 teste.py < entrada.txt  
X = 8  
[maicon@inf58-140 testes]$
```

- Conteúdo do arquivo entrada.txt:

15

-7

# Primeiro Problema - Extremamente Básico (1001)

**1001**




[Descrição](#)  
[Tela Cheia](#)  
[Enviar](#)  
[Blocos](#)  
[Ranking](#)  
[Fórum](#)  
[uDebug](#)


+ 6.1 POINTS

INICIANTE

# Primeiro Problema - Extremamente Básico (1001)



**URI**  
ONLINE JUDGE  
PROBLEMS & CONTESTS



**1001**


Descrição  
Tela Cheia  
Enviar  
Blocos  
Ranking  
Fórum  
uDebug

+ 6.1 POINTS

INICIANTE

**USING AN AD BLOCKER?**

Please, consider disabling your ad-blocker for URI Online Judge. We'll show a non-intrusive, dev-oriented ad in this area. Help us keep education free!



**ENVIAR**

ENVIE SUA SOLUÇÃO PARA UM DOS PROBLEMAS DISPONÍVEIS.

PROBLEMA  
1001

LINGUAGEM  
Python 3 (Python 3.4.3) [+1s]

SOURCE CODE


```
1  #-*- coding: utf-8 -*-
2
3  ...
4  Escreva a sua solução aqui
5  Code your solution here
6  Escriba su solución aquí
7  ...
```

ATENÇÃO: PARA SUBMISSÕES EM JAVA, O NOME DA CLASSE DEVE SER MAIN


ENVIAR



# Primeiro Problema - Extremamente Básico (1001)



**URI**  
ONLINE JUDGE  
PROBLEMS & CONTESTS



**1001**


- Descrição
- Tela Cheia
- Enviar
- Blocos
- Ranking
- Fórum
- uDebug

+ 6.1 POINTS

INICIANTE

**USING AN AD BLOCKER?**

Please, consider disabling your ad-blocker for URI Online Judge. We'll show a non-intrusive, dev-oriented ad in this area. Help us keep education free!



**ENVIAR**

ENVIE SUA SOLUÇÃO PARA UM DOS PROBLEMAS DISPONÍVEIS.

PROBLEMA

1001

LINGUAGEM

Python 3 (Python 3.4.3) [+1s]

SOURCE CODE

```
1 # -*- coding: utf-8 -*-
2 a = int(input())
3 b = int(input())
4 print("X = %s" % (a + b))
5
```

ATENÇÃO: PARA SUBMISSÕES EM JAVA, O NOME DA CLASSE DEVE SER MAIN

**ENVIAR**

# Primeiro Problema - Extremamente Básico (1001)

The screenshot shows the URI Online Judge website interface. At the top, there's a navigation bar with links: HOME, PERFIL, NEWS, ACADEMIC, CONTESTS, FÓRUM, PROBLEMAS, SUBMISSÕES, RANKS, and SAIR. A green notification banner reads: "Código recebido! Sua submissão está na fila para ser julgada...". Below this, there's a sidebar on the left with four sections: "AO VIVO" (O que os outros estão resolvendo), "LISTAR" (Liste todas as suas submissões), "RASCUNHOS" (Liste todos os seus rascunhos), and "URI ONLINE JUDGE PROBLEMS & CONTESTS". The main content area features a large "ENVIAR" button with a pencil icon and the text "ENVIE SUA SOLUÇÃO PARA UM DOS PROBLEMAS DISPONÍVEIS.". Below this, there's a "PROBLEMA" input field and a "LINGUAGEM" dropdown menu currently set to "C++ (g++ 4.8.5, -std=c++11 -O2 -lm) [+0s]". The "SOURCE CODE" section contains a C++ template for problem 1001, which includes a header for iostream, a namespace std, and a main function with a comment in Portuguese, English, and Spanish to write the solution.

Hi, Maicon Rafael Zetelli [Fur...]  
xsolyter@gmail.com

HOME PERFIL NEWS ACADEMIC CONTESTS FÓRUM PROBLEMAS SUBMISSÕES RANKS SAIR

Código recebido! Sua submissão está na fila para ser julgada...

**ENVIAR**  
ENVIE SUA SOLUÇÃO PARA UM DOS PROBLEMAS DISPONÍVEIS.

PROBLEMA

LINGUAGEM  
C++ (g++ 4.8.5, -std=c++11 -O2 -lm) [+0s]

**SOURCE CODE**

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6
7     /**
8      * Escreva a sua solução aqui
9      * Code your solution here
10     * Escriba su solución aquí
11     */
12
13     return 0;
14 }
```

**TODAS**  
ENVIAR  
AO VIVO  
RASCUNHOS  
FAQS  
RESPOSTAS

**URI**  
ONLINE JUDGE  
PROBLEMS & CONTESTS

**AO VIVO**  
O que os outros estão resolvendo.

**LISTAR**  
Liste todas as suas submissões.

**RASCUNHOS**  
Liste todos os seus rascunhos.

# Primeiro Problema - Extremamente Básico (1001)



**URI**  
ONLINE JUDGE  
PROBLEMS & CONTESTS



**AO VIVO**  
O que os outros  
estão resolvendo.



**LISTAR**  
Liste todas as  
suas submissões.



**TENTADO**  
Problemas ainda  
não resolvidos.



**SUBMISSÕES**

AQUI VOCÊ PODE ENCONTRAR TODAS AS SUAS SUBMISSÕES.

**BARRA DE PESQUISA**

#	PROBLEMA	RESPOSTA	LINGUAGEM	HORA	DATA
12650526	1001 Extremamente Básico	Wrong answer (100%)	Python 3	0.012	14/01/19 12:28:33
12650505	1001 Extremamente Básico	Accepted	Python 3	0.016	14/01/19 12:22:37
12650501	1001 Extremamente Básico	Accepted	Python 3	0.020	14/01/19 12:21:53
12650483	1001 Extremamente Básico	Accepted	Python 3	0.020	14/01/19 12:17:34
12625026	1002 Área do Circulo	Accepted	Python 3	0.024	08/01/19 09:36:25
12557906	2399 Campo Minado	Accepted	Python	0.016	21/12/18 11:06:49
12557901	1439 Bora Bora	Accepted	C++17	0.000	21/12/18 11:05:18

## Segundo Problema - Área do Círculo (1002)

A fórmula para calcular a área de uma circunferência é: **area** = **n** . **raio**<sup>2</sup>. Considerando para este problema que **n** = 3.14159:

- Efetue o cálculo da área, elevando o valor de **raio** ao quadrado e multiplicando por **n**.

### Entrada

A entrada contém um valor de ponto flutuante (dupla precisão), no caso, a variável **raio**.

### Saída

Apresentar a mensagem "A=" seguido pelo valor da variável **area**, conforme exemplo abaixo, com 4 casas após o ponto decimal. Utilize variáveis de dupla precisão (double). Como todos os problemas, não esqueça de imprimir o fim de linha após o resultado, caso contrário, você receberá "Presentation Error".

Exemplos de Entrada	Exemplos de Saída
2.00	A=12.5664
100.64	A=31819.3103
150.00	A=70685.7750

## Segundo Problema - Área do Círculo (1002)

### Solução em Python

```
# -*- coding: utf-8 -*-  
PI = 3.14159  
raio = float(input())  
area = PI * raio * raio  
print("A=%.4f" % (area))
```

### Pontos importantes

- A função **print()** já imprime com uma quebra de linha
- Note que não há espaços entre **A**, **=** e **%.4f**
- **%.4f** permite formatar um número com 4 casas decimais. Já são preenchidos os 0's para completar 4 casas.
- Lembre de incluir como primeira linha: **# -\*- coding: utf-8 -\*-**

## Segundo Problema - Área do Círculo (1002)

### Testando a Solução

#### Execução - entrada por redirecionamento de arquivo

```
[maicon@inf58-140 testes]$ python3 teste.py < entrada.txt  
A=70685.7750  
[maicon@inf58-140 testes]$
```

- Conteúdo do arquivo entrada.txt:

150.00

# Terceiro Problema - Cálculo Simples (1010)

Neste problema, deve-se ler o código de uma peça 1, o número de peças 1, o valor unitário de cada peça 1, o código de uma peça 2, o número de peças 2 e o valor unitário de cada peça 2. Após, calcule e mostre o valor a ser pago.

## Entrada

O arquivo de entrada contém duas linhas de dados. Em cada linha haverá 3 valores, respectivamente dois inteiros e um valor com 2 casas decimais.

## Saída

A saída deverá ser uma mensagem conforme o exemplo fornecido abaixo, lembrando de deixar um espaço após os dois pontos e um espaço após o "R\$". O valor deverá ser apresentado com 2 casas após o ponto.

Exemplos de Entrada	Exemplos de Saída
12 1 5.30 16 2 5.10	VALOR A PAGAR: R\$ 15.50
13 2 15.30 161 4 5.20	VALOR A PAGAR: R\$ 51.40
1 1 15.10 2 1 15.10	VALOR A PAGAR: R\$ 30.20

# Terceiro Problema - Cálculo Simples (1010)

## Solução em Python

```
# -*- coding: utf-8 -*-
c1, n1, v1 = input().split()
c1 = int(c1)
n1 = int(n1)
v1 = float(v1)

c2, n2, v2 = input().split()
c2 = int(c2)
n2 = int(n2)
v2 = float(v2)

total = n1 * v1 + n2 * v2
print("VALOR A PAGAR: R$ %.2f" % total)
```

## Pontos importantes

- A função **split()** permite quebrar a linha onde estiver o espaço. Assim, ao ler algo como **12 1 5.30**, o valor **12** irá para **c1**, o valor **1** irá para **n1** e o valor **5.30** irá para **v1**.



## Terceiro Problema - Cálculo Simples (1010)

### Testando a Solução

#### Execução - entrada por redirecionamento de arquivo

```
[maicon@inf58-140 testes]$ python3 teste.py < entrada.txt  
VALOR A PAGAR: R$ 15.50  
[maicon@inf58-140 testes]$
```

- Conteúdo do arquivo entrada.txt:

12 1 5.30

16 2 5.10

# Problema 1087 - Casos de Teste Terminam com 0

O grande mestre de xadrez Kary Gasparov inventou um novo tipo de problema de xadrez: dada a posição de uma dama em um tabuleiro de xadrez vazio (ou seja, um tabuleiro  $8 \times 8$ , com 64 casas), de quantos movimentos, no mínimo, ela precisa para chegar em outra casa do tabuleiro?

Kary achou a solução para alguns desses problemas, mas teve dificuldade com outros, e por isso pediu que você escrevesse um programa que resolve esse tipo de problema.

## Entrada

A entrada contém vários casos de teste. A primeira e única linha de cada caso de teste contém quatro inteiros  $X_1, Y_1, X_2 \in Y_2$  ( $1 \leq X_1, Y_1, X_2, Y_2 \leq 8$ ). A dama começa na casa de coordenadas  $(X_1, Y_1)$ , e a casa de destino é a casa de coordenadas  $(X_2, Y_2)$ . No tabuleiro, as colunas são numeradas da esquerda para a direita de 1 a 8 e as linhas de cima para baixo também de 1 a 8. As coordenadas de uma casa na linha X e coluna Y são  $(X, Y)$ .

O final da entrada é indicado por uma linha contendo quatro zeros.

## Saída

Para cada caso de teste da entrada seu programa deve imprimir uma única linha na saída, contendo um número inteiro, indicando o menor número de movimentos necessários para a dama chegar em sua casa de destino.

Exemplo de Entrada	Exemplo de Saída
4 4 6 2	1
3 5 3 5	0
5 5 4 3	2
0 0 0 0	

# Solução em Python

```
# -*- coding: utf-8 -*-  
  
while True:  
    x1,y1,x2,y2 = input().split()  
  
    x1 = int(x1)  
    y1 = int(y1)  
    x2 = int(x2)  
    y2 = int(y2)  
  
    if (x1 == 0 and y1 == 0 and x2 == 0 and y2 == 0): break  
  
    #{...} Faz alguma coisa  
    print(k)
```

- Criamos um loop infinito para ler as entradas e saímos do loop apenas quando todos os valores do caso de teste são 0.
- Para cada caso de teste lido, processo o mesmo e imprimo a saída necessária.

# Problema 1467 - Casos de Teste Terminam com EOF

Todos devem conhecer o jogo Zerinho ou Um (em algumas regiões também conhecido como Dois ou Um), utilizado para determinar um ganhador entre três ou mais jogadores. Para quem não conhece, o jogo funciona da seguinte maneira. Cada jogador escolhe um valor entre zero ou um; a um comando (geralmente um dos competidores anuncia em voz alta "Zerinho ou... Um!"), todos os participantes mostram o valor escolhido, utilizando uma das mãos: se o valor escolhido foi um, o competidor mostra o dedo indicador estendido; se o valor escolhido foi zero, mostra a mão com todos os dedos fechados. O ganhador é aquele que tiver escolhido um valor diferente de todos os outros; se não há um jogador com valor diferente de todos os outros (por exemplo todos os jogadores escolhem zero, ou um grupo de jogadores escolhe zero e outro grupo escolhe um), não há ganhador. Alice, Beto e Clara são grandes amigos e jogam Zerinho a toda hora: para determinar quem vai comprar a pipoca durante a sessão de cinema, quem vai entrar na piscina primeiro, etc. Jogam tanto que resolveram fazer um plugin no Facebook para jogar Zerinho. Como não sabem programar, dividiram as tarefas entre amigos que sabem, inclusive você. Dados os três valores escolhidos por Alice, Beto e Clara, cada valor zero ou um, escreva um programa que determina se há um ganhador, e nesse caso determina quem é o ganhador.

## Entrada

A entrada é composta por vários casos de teste. Cada caso de teste consiste de uma única linha, que contém três inteiros A, B e C (A,B,C só podem ser 0 ou 1), indicando respectivamente os valores escolhidos por Alice, Beto e Clara. O final da entrada é determinado por EOF (End of File).

## Saída

Para cada caso de teste, seu programa deve produzir uma única linha, contendo um único caractere. Se o vencedor é Alice o caractere deve ser 'A', se o vencedor é Beto o caractere deve ser 'B', se o vencedor é Clara o caractere deve ser 'C' e se não há vencedor o caractere deve ser '\*' (asterisco).

Exemplo de Entrada	Exemplo de Saída
1 1 0	C
0 0 0	*
1 0 0	A

# Solução em Python

```
# -*- coding: utf-8 -*-  
  
while True:  
    try:  
        a, b, c = input().split()  
        a = int(a)  
        b = int(b)  
        c = int(c)  
  
        #{...} Faz alguma coisa  
        print(k)  
    except EOFError:  
        break
```

- Criamos um loop infinito para ler as entradas e saímos do loop apenas quando ocorrer uma exceção do tipo **EOFError**, que significa que o arquivo de entrada terminou.

# Atividade

Disponível no Moodle