

PRIMEIROS PASSOS NO PYTHON



AMBIENTE DE DESENVOLVIMENTO PYTHON IDLE 3.7.3

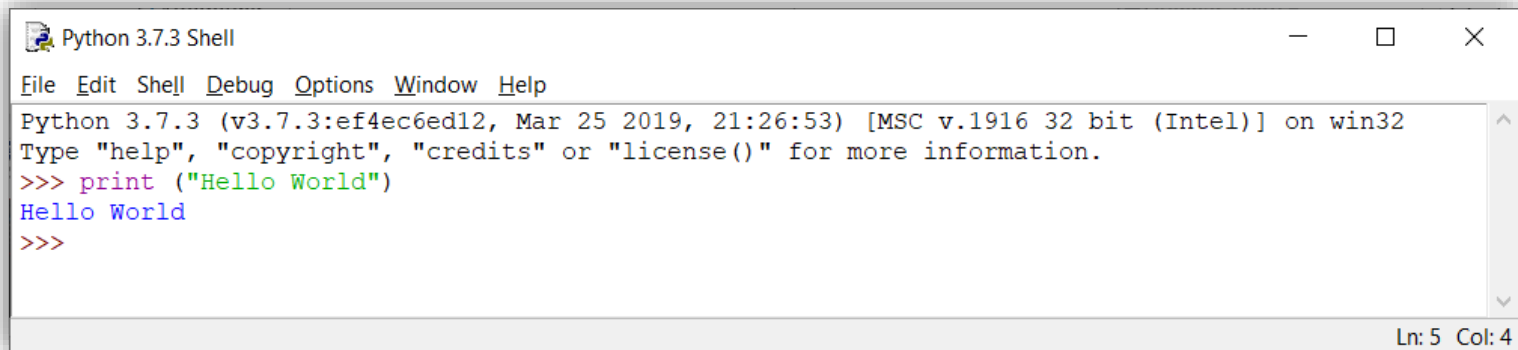
AMBIENTE DE DESENVOLVIMENTO PYTHON IDLE

INTERPRETADOR PYTHON IDLE

- Para acessar o **PYTHON IDLE**, vá em **Iniciar -> Programas -> Python 3.7 -> IDLE**
- No interpretador **PYTHON IDLE**, o programa pode ser executado direto no comando. Como exemplo, digitar o comando:

```
print ("Hello World")
```

- Ao pressionar <Enter> o comando é automaticamente executado no terminal:



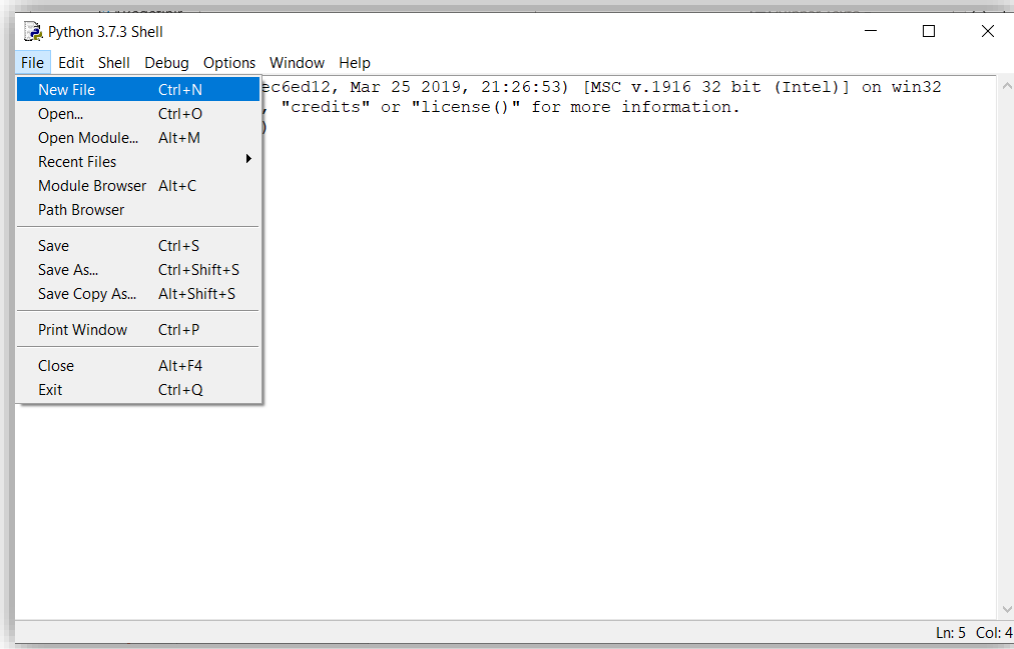
```
Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 21:26:53) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print ("Hello World")
Hello World
>>>
```

Ln: 5 Col: 4

AMBIENTE DE DESENVOLVIMENTO PYTHON IDLE

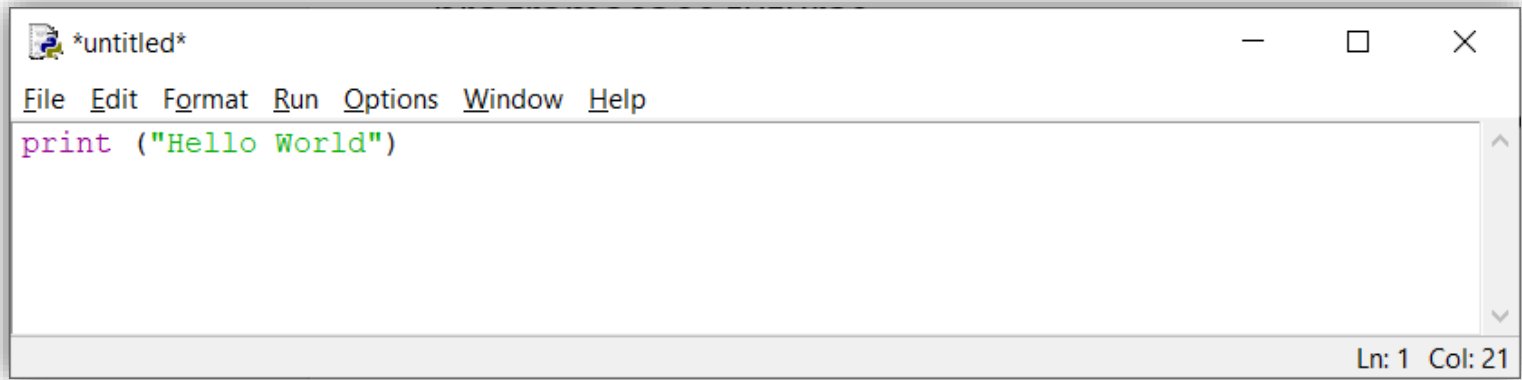
EDITANDO ARQUIVOS:

- Um programa nada mais é que um arquivo-texto, escrito em um formato especial (linguagem).
- Existem inúmeros editores de arquivos. Inclusive um editor de texto como “Bloco de Notas”.
- O IDLE contem um editor de texto que pode ser acessado pelo caminho: **File - > New File**.



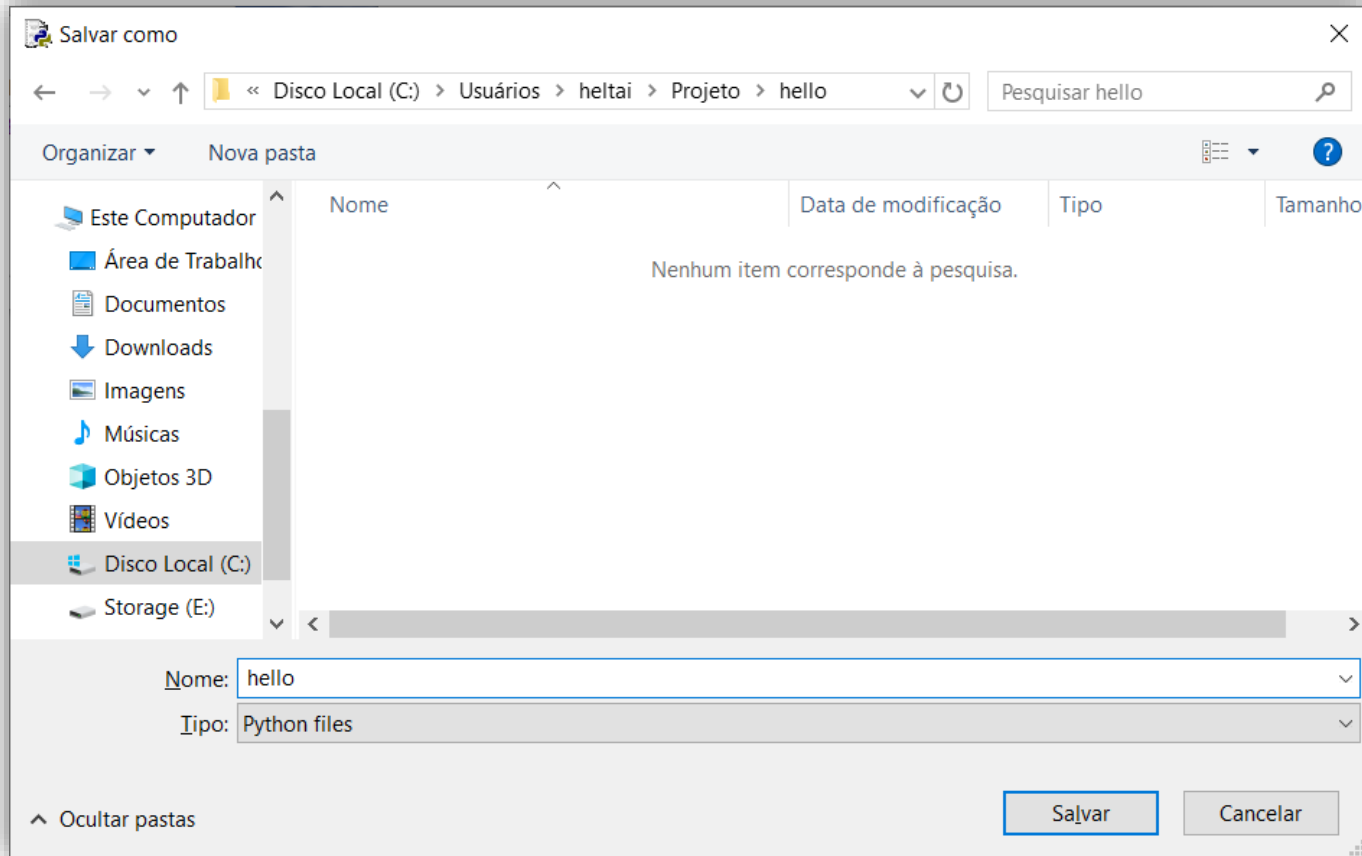
AMBIENTE DE DESENVOLVIMENTO PYTHON IDLE

- A janela de editor pode ser desenvolvido sua programação Python e salvo para eventuais retorno ou programações futuras.
- Escreva o mesmo comando **print** (“Hello World”) no editor.



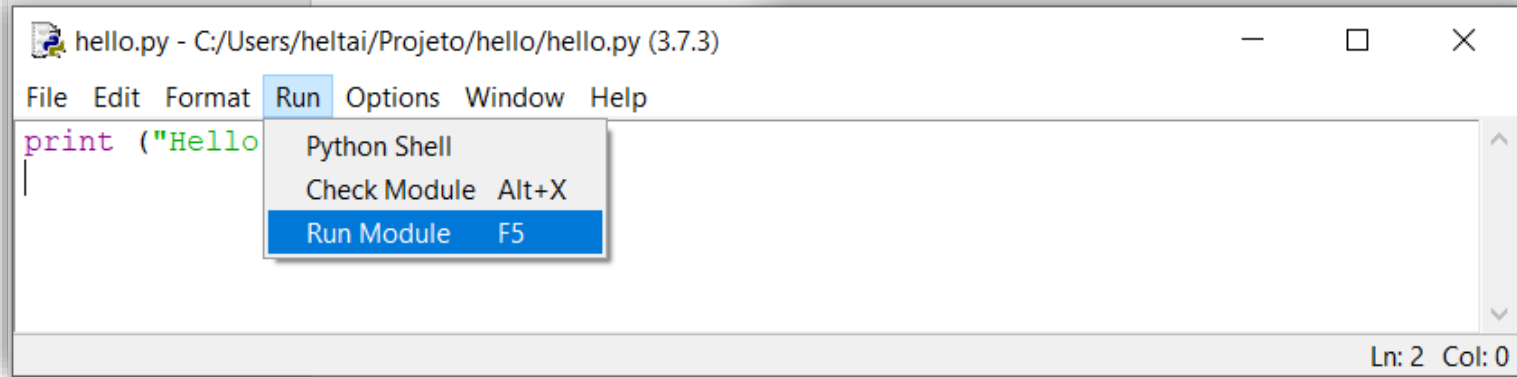
AMBIENTE DE DESENVOLVIMENTO PYTHON IDLE

- Salve em um diretório de preferencia com extensão **hello.py**

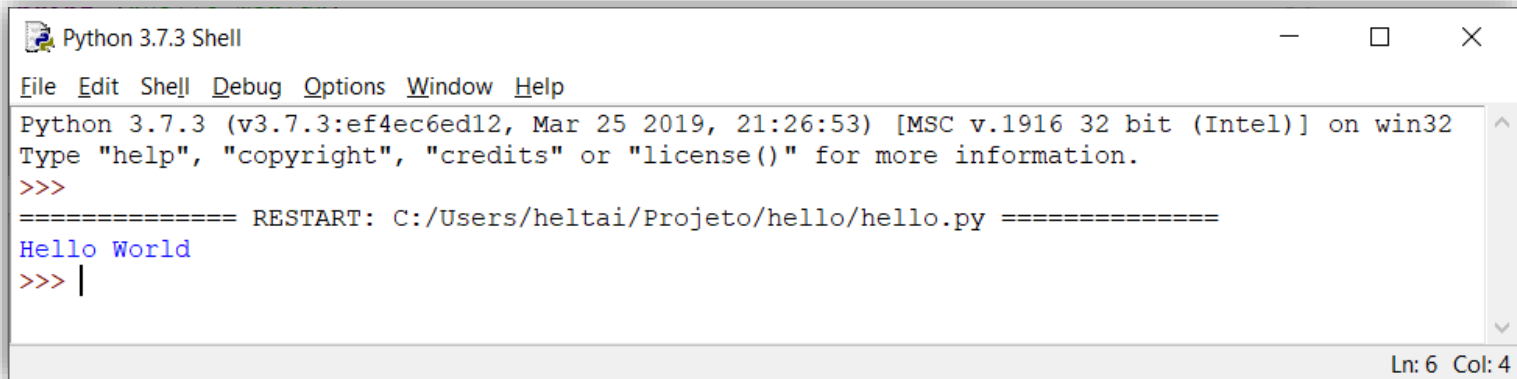


AMBIENTE DE DESENVOLVIMENTO PYTHON IDLE

- Execute o programa **hello.py** com **Run -> Run Module** (Atalho F5)



- O Python Shell executará o **hello.py** dando como resposta a mensagem **"Hello World"**.





CUIDADOS AO PROGRAMAR EM PYTHON

CUIDADOS AO PROGRAMAR EM PYTHON

ALGUNS CUIDADOS QUE OS PROGRAMADORES DEVEM TOMAR CUIDADO AO PROGRAMAR:

- **Letras** maiúsculas e minúsculas são diferentes. Exemplo: **print** e **Print** são completamente diferentes.
- **Aspas** são muito importantes e não devem ser esquecidas. Toda vez que abrir aspas, não se esqueça de fecha-las. Observe que os IDEs muda a cor do texto entre aspas, facilitando essa verificação.
- **Parênteses** não são opcionais em Python. Não remova os parentes dos programas e preste a mesma atenção dada em Aspas. Todo parêntese aberto deve ser fechado.
- **Espaços** são muitos importantes. Python se baseia na quantidade de espaço em branco antes do início de cada linha para realizar diversas operações. Nunca junte duas linhas em uma só até sentir-se seguro sobre como escrever corretamente. Tente usar apenas espaços para alinhar seu programa, evite usar TABs.

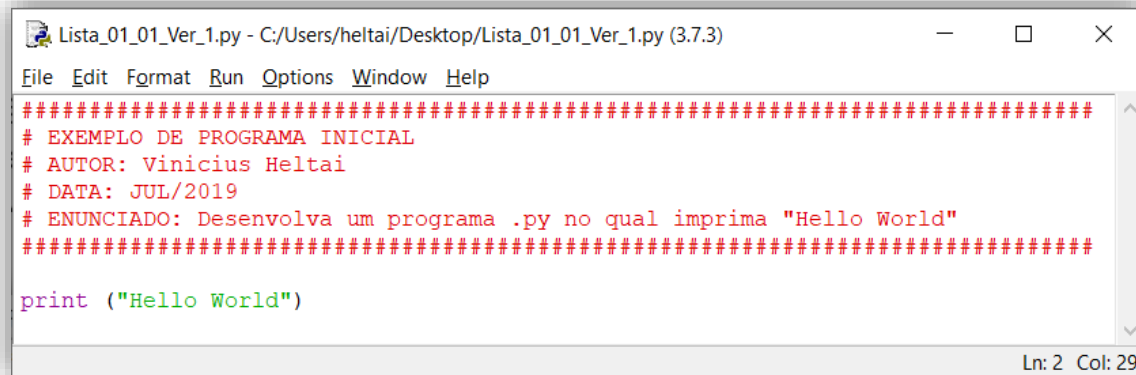


COMENTARIOS

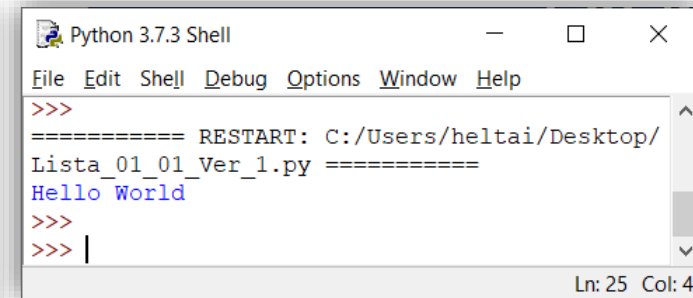
COMENTARIOS

COMENTARIOS (ÚNICA LINHA)

- Comentários são pequenos textos, em geral de algumas poucas linhas, que explicam alguma coisa no código, em geral para ajudar um possível leitor, um tempo depois, a entender o que está acontecendo.
- O comentário é ativado com o símbolo **#** no **início de cada linha**. Do ponto de inserção até o final da linha o interpretador ignora, fica apenas no código fonte. Exemplo:



```
#####  
# EXEMPLO DE PROGRAMA INICIAL  
# AUTOR: Vinicius Heltai  
# DATA: JUL/2019  
# ENUNCIADO: Desenvolva um programa .py no qual imprima "Hello World"  
#####  
  
print ("Hello World")
```

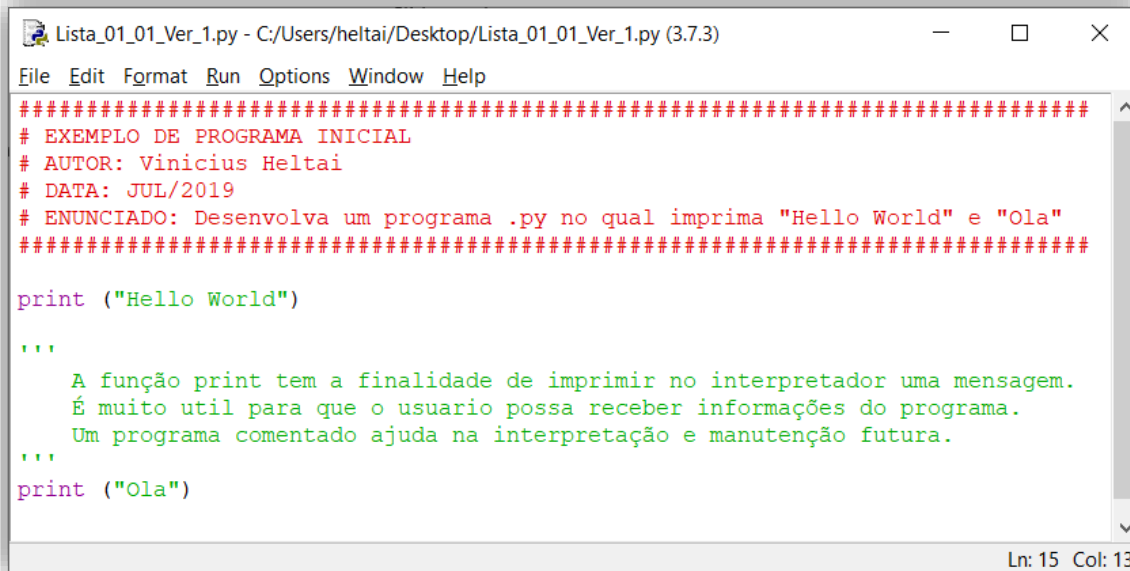


```
>>>  
===== RESTART: C:/Users/heltai/Desktop/  
Lista_01_01_Ver_1.py =====  
Hello World  
>>>  
>>> |
```

COMENTARIOS

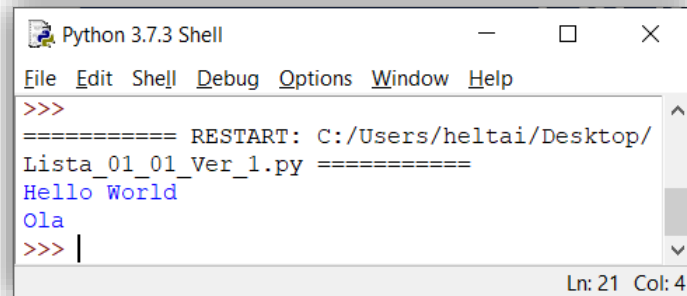
DOCSTRINGS (COMENTARIO DE VARIAS LINHAS)

- Comentário é um pequeno texto (uma única linha). Já o DOCSTRINGS são comentários de varias linhas. É ativado e desativado com a tríplice aspas **"""** na linha inicial que deseja ativar o docstrings e novamente **"""** na linha final que deseja encerrar o docstrings.
- Utilizada normalmente para documentar uma função, uma classe, um módulo, etc.



```
#####  
# EXEMPLO DE PROGRAMA INICIAL  
# AUTOR: Vinicius Heltai  
# DATA: JUL/2019  
# ENUNCIADO: Desenvolva um programa .py no qual imprima "Hello World" e "Ola"  
#####  
  
print ("Hello World")  
  
'''  
    A função print tem a finalidade de imprimir no interpretador uma mensagem.  
    É muito util para que o usuario possa receber informações do programa.  
    Um programa comentado ajuda na interpretação e manutenção futura.  
'''  
  
print ("Ola")
```

Ln: 15 Col: 13



```
Python 3.7.3 Shell  
File Edit Shell Debug Options Window Help  
>>>  
===== RESTART: C:/Users/heltai/Desktop/  
Lista_01_01_Ver_1.py =====  
Hello World  
Ola  
>>> |
```

Ln: 21 Col: 4

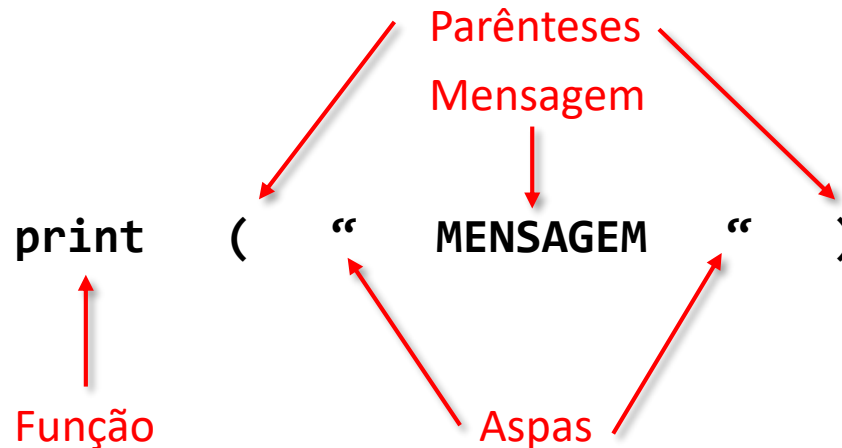


FUNÇÃO PRINT

FUNÇÃO PRINT

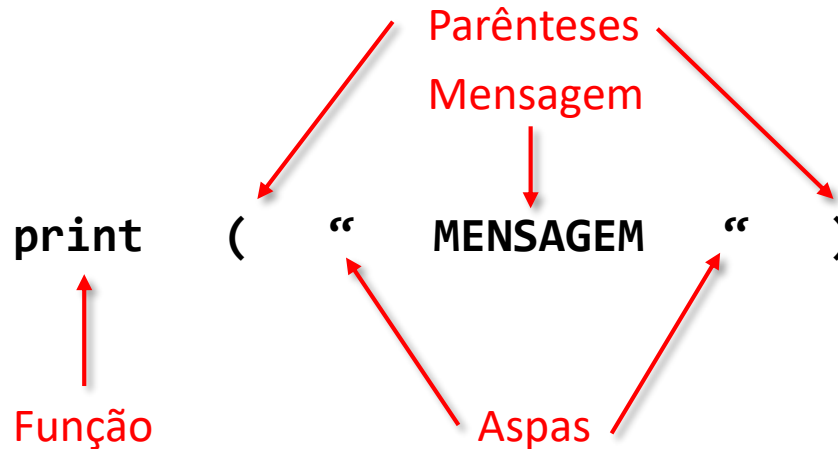
FUNÇÃO PRINT

- A função print informa que vamos exibir algo na tela. A função exibe uma mensagem na tela do computador.
- É utilizada sempre que quiser mostrar algo para o usuário do computador, como uma mensagem, uma pergunta ou um resultado de uma operação de cálculo.
- Sua sintaxe é mostrado seguir:



FUNÇÃO PRINT

- **ASPAS** – São utilizadas para separar textos destinados ao usuário do computador do resto do programa. Utiliza-se aspas para indicar o inicio e o fim do texto da mensagem.
- **PARÊNTESES** – São utilizados para separar os parâmetro de uma função, no caso, so de print.
- **PARAMETRO** – É o valor passado para uma função, no caso a mensagem.





CONCEITO DE VARIÁVEL

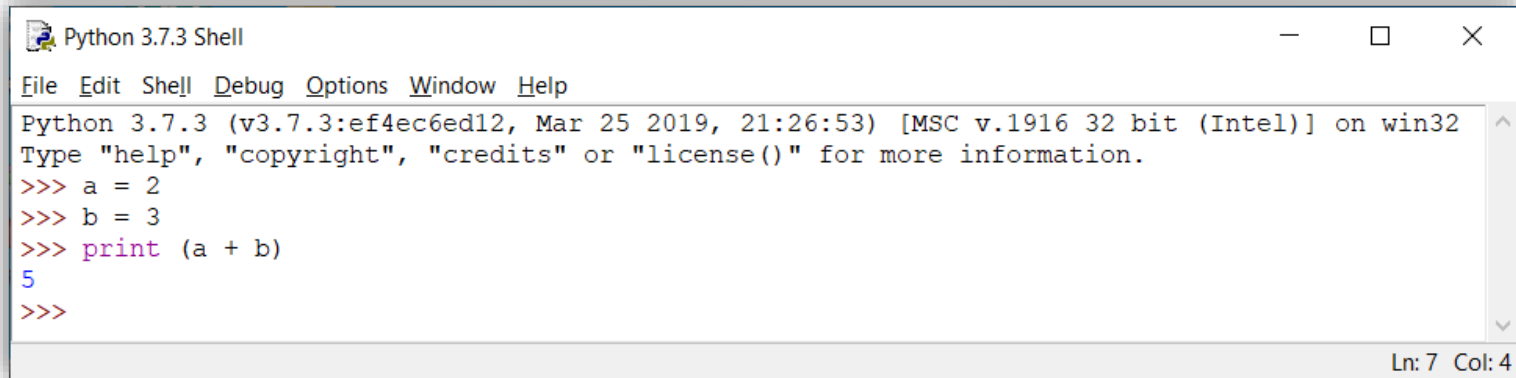
CONCEITO DE VARIÁVEL – REVISÃO

O QUE É UMA VARIÁVEL?

- Em programação todas as informações são armazenadas em estruturas denominadas variáveis.
- Uma variável em possui:
 - ✓ Um tipo que indica o tamanho
 - ✓ Um nome para referenciar o conteúdo
 - ✓ Um espaço reservado na memória para armazenar seu valor
- Variável é um espaço de memória contém um valor o qual pode ser alterado ao longo do tempo.
- Desta forma, podemos concluir que: Variável são utilizadas para armazenar valores e para dar nome a uma área de memória do computador onde armazenamos dados.
- Para armazenar algo numa variável se utiliza o símbolo “=” no qual chamamos de atribuição (uma vez que algo é atribuído a uma variável).
- Quando se lê um programa, as operações de atribuição serão chamadas de “recebe”, ou seja, uma variável recebe um valor.

CONCEITO DE VARIÁVEL – REVISÃO

- No exemplo abaixo, Lemos:
 - A variável a recebe o valor 2
 - A variável b recebe o valor 3



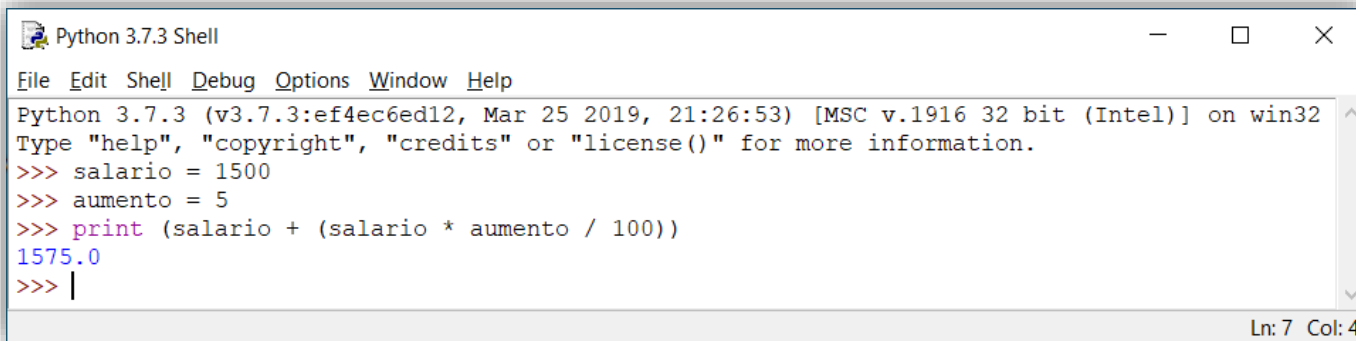
```
Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 21:26:53) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> a = 2
>>> b = 3
>>> print (a + b)
5
>>>
```

Ln: 7 Col: 4

- Como as variáveis a e b valem 2 e 3, a soma de a + b, equivale a 2 + 3 que é igual a 5.

CONCEITO DE VARIÁVEL – REVISÃO

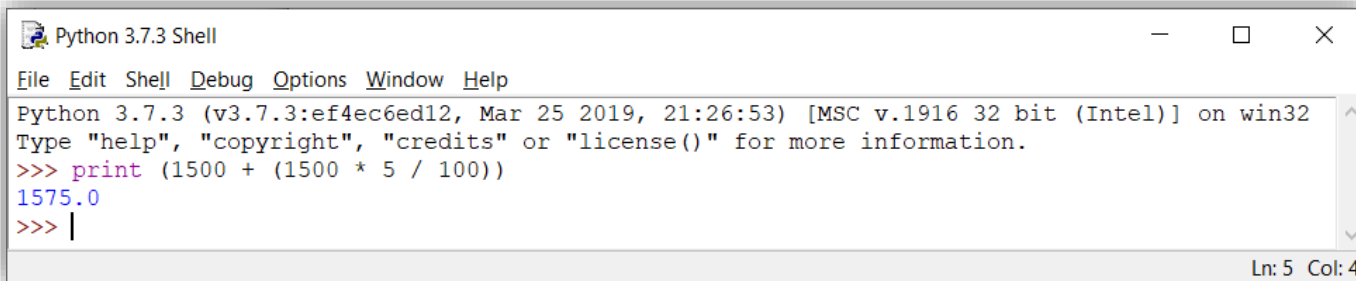
EXEMPLO:



```
Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 21:26:53) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> salario = 1500
>>> aumento = 5
>>> print (salario + (salario * aumento / 100))
1575.0
>>> |
```

Ln: 7 Col: 4

- Uma variável chamada salario, recebe o valor de 1500. Em outra variável chamada aumento, recebe o valor 5. Descrevendo a formula que calcula o valor do novo salario depois de receber um aumento.
- O mesmo programa poderia ser escrito como:



```
Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 21:26:53) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print (1500 + (1500 * 5 / 100))
1575.0
>>> |
```

Ln: 5 Col: 4

- Ao utilizar variáveis, podemos referenciar o mesmo valor varias vezes, sem esquecer o significado dos valores e podem ser extensamente utilizados para cálculos de outros aumentos de salários.

CONCEITO DE VARIÁVEL

NOME DE VARIÁVEIS

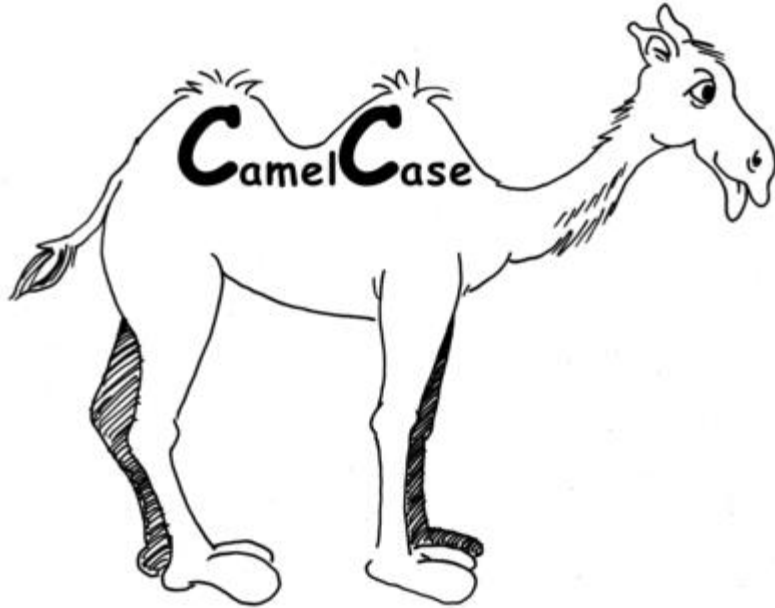
- Nomes de variáveis devem iniciar obrigatoriamente com uma letra, ou sublinhado (_).
- Pode conter números no nome da variável.
- Em Python 3 é permitido a utilização de acentos em nomes de variáveis pois utiliza UTF-8. Porém não recomendado essa prática para evitar erros.
- Exemplos de nomes válidos e inválidos em Python:

NOME	VÁLIDO	COMENTARIOS
a1	Sim	Embora contenha um número, o nome a1 inicia com letra.
velocidade	Sim	Nome formado por letras.
velocidade90	Sim	Nome formado por letras e números, mas iniciando por letra.
Salário médio	Não	Nomes de variáveis não podem conter espaços em branco.
_b	Sim	O sublinha () é aceito em nomes de variáveis, mesmo no início.
1a	Não	Nomes de variáveis não podem começar com números.

CONCEITO DE VARIÁVEL

NOME DE VARIÁVEIS – BOAS PRATICAS - CAMELCASE:

- As boas praticas recomendam:
 - Variáveis sejam escritas apenas em minúsculas.
 - Caso seja composto, utilizar-se uma letra maiúscula para cada nova palavra (contaPosicao e contaValorTotal) padrão conhecido como camelCase.





ENTRADA DE DADOS

ENTRADA DE DADOS

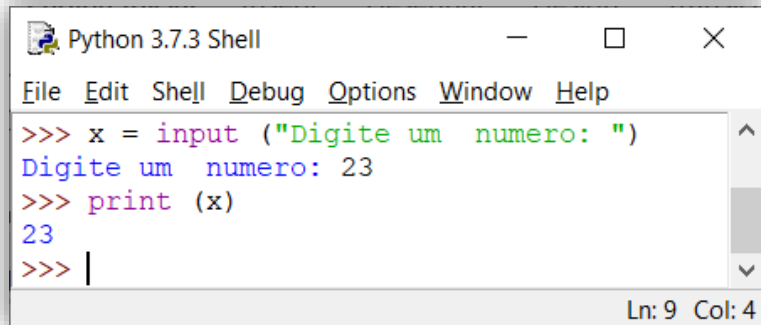
- A função input é utilizada para solicitar dados do usuário. O comando recebe um parâmetro, que é a mensagem a ser exibida, e retorna o valor digitado pelo usuário.
- A função sempre retorna valores do tipo string.

SINTAXE:

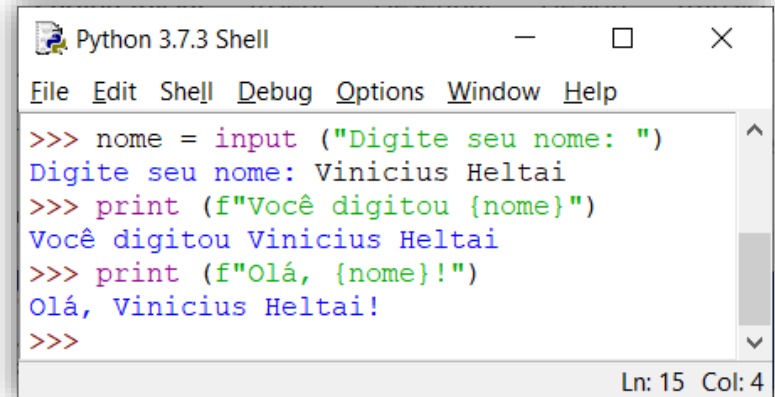
```
VARIAVEL = input ( " MENSAGEM AO USUÁRIO " )
```

- Atenção, pois o programa não observa o valor digitado pelo usuário, a verificação deve ser feita pelo programa.

EXEMPLO:



```
Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
>>> x = input ("Digite um numero: ")
Digite um numero: 23
>>> print (x)
23
>>> |
Ln: 9 Col: 4
```



```
Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
>>> nome = input ("Digite seu nome: ")
Digite seu nome: Vinicius Heltai
>>> print (f"Você digitou {nome}")
Você digitou Vinicius Heltai
>>> print (f"Olá, {nome}!")
Olá, Vinicius Heltai!
>>>
Ln: 15 Col: 4
```



CALCULOS SIMPLES EM PYTHON

CALCULOS SIMPLES EM PYTHON

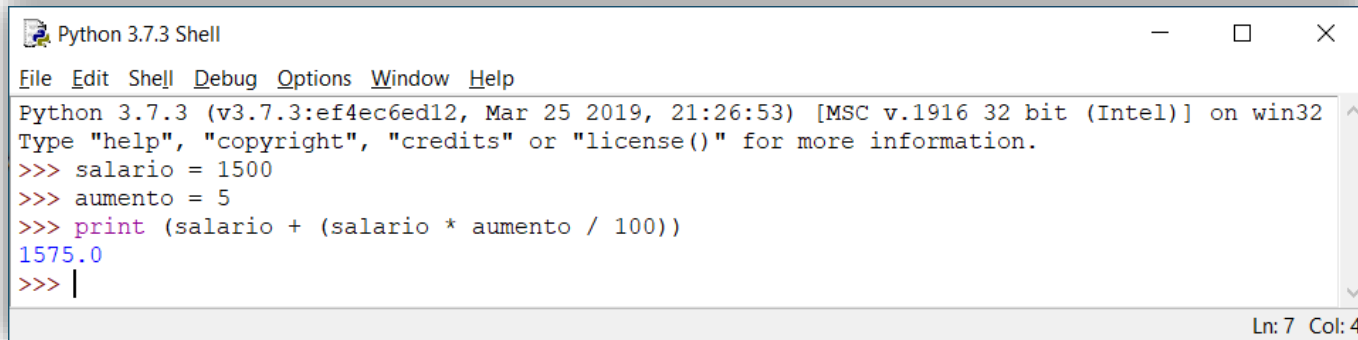
INTERPRETADOR COMO CALCULADORA

- Os parênteses são utilizados em Python da mesma forma que em expressões matemáticas, ou seja, para alternar a ordem de execução de uma operação. A ordem de precedência das operações, temos as seguintes prioridades:
 - Exponenciação ou potenciação (**)
 - Multiplicação (*), divisão (/ e //) e módulo (%)
 - Adição (+) e subtração (-)
- As operações são realizadas da esquerda para a direita.
- Tabela dos operadores e operações matemáticas.

OPERADOR	OPERAÇÃO
+	Adição
-	Subtração
*	Multiplicação
/	Divisão (com resultado fracionado)
//	Divisão (com resultados inteiros)
%	Modulo ou resto
**	Exponenciação ou Potenciação

CONCEITO DE VARIÁVEL

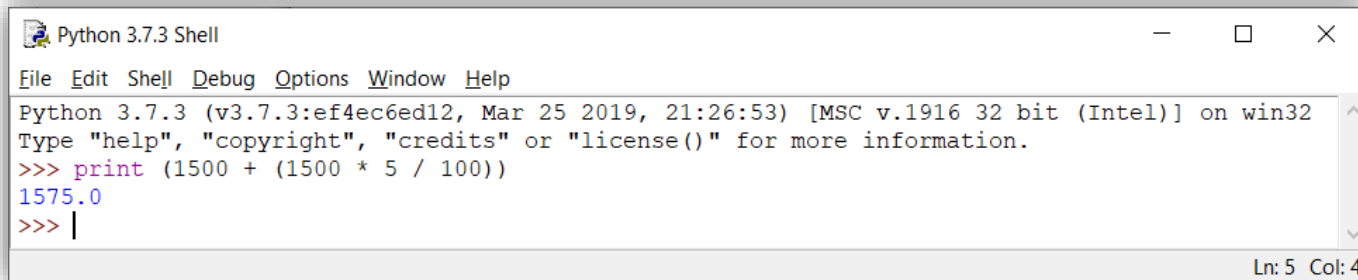
EXEMPLO:



```
Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 21:26:53) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> salario = 1500
>>> aumento = 5
>>> print (salario + (salario * aumento / 100))
1575.0
>>> |
```

Ln: 7 Col: 4

- Uma variável chamada salario, recebe o valor de 1500. Em outra variável chamada aumento, recebe o valor 5. Descrevendo a formula que calcula o valor do novo salario depois de receber um aumento.
- O mesmo programa poderia ser escrito como:



```
Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 21:26:53) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print (1500 + (1500 * 5 / 100))
1575.0
>>> |
```

Ln: 5 Col: 4

- Ao utilizar variáveis, podemos referenciar o mesmo valor varias vezes, sem esquecer o significado dos valores e podem ser extensamente utilizados para cálculos de outros aumentos de salários.



CONVERSÃO DA ENTRADA DE DADOS

CONVERSÃO DA ENTRADA DE DADOS

CONVERSÃO DA ENTRADA DE DADOS

- A função input sempre retorna valor do tipo string (não importa qual tipo de dados seja inserido, será entendido sempre como string).
- Para converter o tipo de dado, utiliza-se a seguinte sintaxe:

SINTAXE:

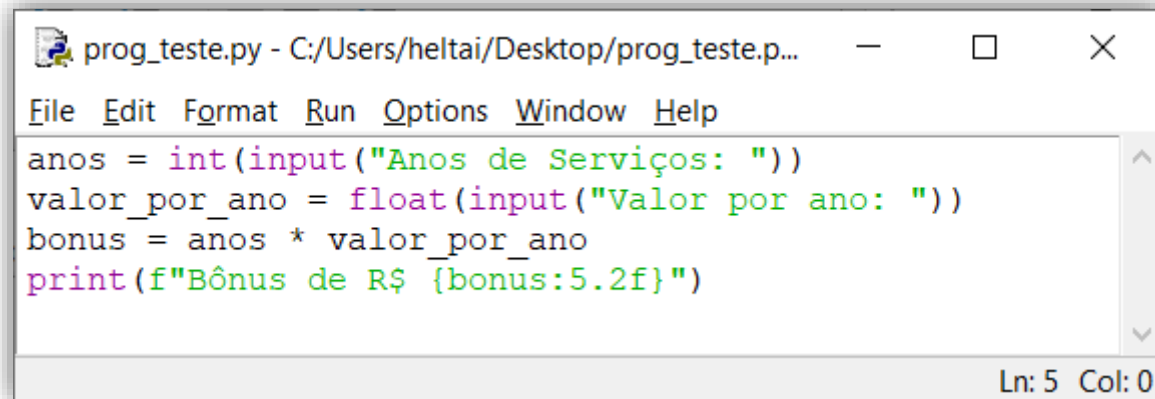
```
VARIAVEL = int ( input ( “ MENSAGEM AO USUÁRIO ” ))      #1
```

```
VARIAVEL = float ( input ( “ MENSAGEM AO USUÁRIO ” ))      #2
```

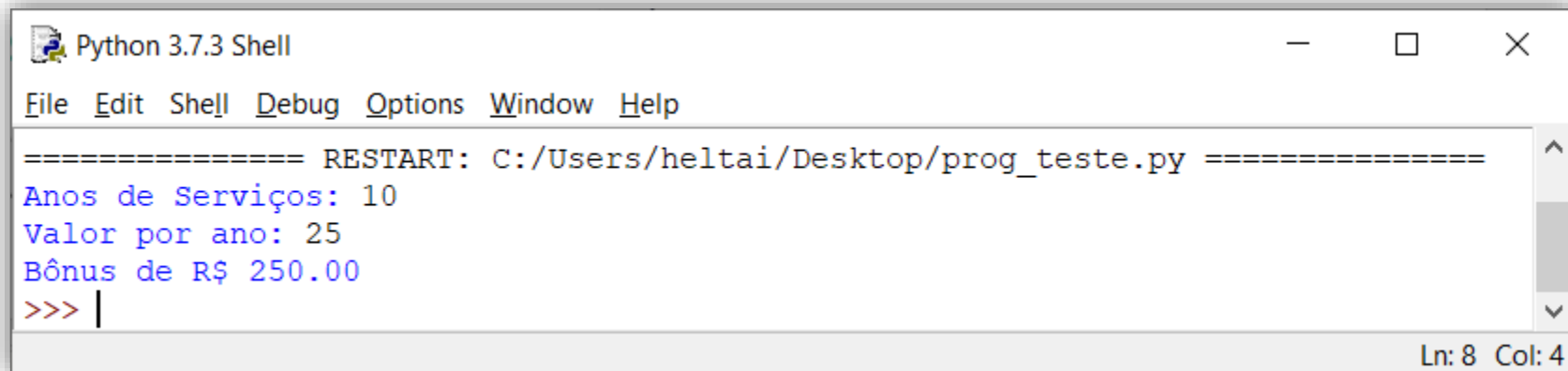
- #1 – Converte a string de entrada para int (numero inteiro)
- #2 – Converte a string de entrada para float (numero decimal)

CONVERSÃO DA ENTRADA DE DADOS

EXEMPLO:



```
File Edit Format Run Options Window Help
anos = int(input("Anos de Serviços: "))
valor_por_ano = float(input("Valor por ano: "))
bonus = anos * valor_por_ano
print(f"Bônus de R$ {bonus:5.2f}")
Ln: 5 Col: 0
```



```
Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
===== RESTART: C:/Users/heltai/Desktop/prog_teste.py =====
Anos de Serviços: 10
Valor por ano: 25
Bônus de R$ 250.00
>>> |
Ln: 8 Col: 4
```



PROF. VINICIUS HELTAI

vinicius.pacheco@docente.unip.br

www.heltai.com.br

(11) 98200-3932



/vheltai



@vheltai



/vheltai



@Vinicius Heltai



@vheltai