

Programando de verdade

SSC0301

Prof. Márcio Delamaro

Resolução Bhaskara

```
Terminal
Arquivo Editar Ver Pesquisar Terminal Ajuda
delamaro@dela-ubuntu:~$ python3
Python 3.6.5 (default, Apr 1 2018, 05:46:30)
[GCC 7.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import math
>>> a = 308
>>> b = 113
>>> c = -1033
>>> delta = b ** 2 - 4 * a * c
>>> delta
1285425
>>> x1 = (-b + math.sqrt(delta)) / (2 * a)
>>> x2 = (-b - math.sqrt(delta)) / (2 * a)
>>> x1
1.6570874169509975
>>> x2
-2.0239705338341145
>>> 
```

Criando o programa

- Arquivo de texto
- *bhaskara.py*
- Executar no prompt do Windows ou Linux
- `> python bhaskara.py`

bhaskara.py

```
import math
```

```
a = 308
```

```
b = 113
```

```
c = -1033
```

```
delta = b ** 2 - 4 * a * c
```

```
x1 = (-b + math.sqrt(delta)) / (2 * a)
```

```
x2 = (-b - math.sqrt(delta)) / (2 * a)
```

```
x1
```

```
x2
```

bhaskara.py

```
import math
```

```
a = 308
```

```
b = 113
```

```
c = -1033
```

```
delta = b ** 2 - 4 * a * c
```

```
x1 = (-b + math.sqrt(delta)) / (2 * a)
```

```
x2 = (-b - math.sqrt(delta)) / (2 * a)
```

x1

x2

Comando de saída

- Comando print
- Exibe um resultado, ou vários...
- `print(x1)`
 - Exibe o valor que estiver em x1
- `print(x1, x2)`
 - Exibe o valor das duas variáveis
- `print('SSC0301')`

Bhaskara com print

```
import math
```

```
a = 308
```

```
b = 113
```

```
c = -1033
```

```
delta = b ** 2 - 4 * a * c
```

```
x1 = (-b + math.sqrt(delta)) / (2 * a)
```

```
x2 = (-b - math.sqrt(delta)) / (2 * a)
```

```
print(x1,x2)
```

Bhaskara com print

```
Terminal
Arquivo Editar Ver Pesquisar Terminal Ajuda
delamaro@dela-ubuntu:~/disciplinas/2018/ICCAmbiental/Aulas/Aula04SalaResumo$
delamaro@dela-ubuntu:~/disciplinas/2018/ICCAmbiental/Aulas/Aula04SalaResumo$ pyt
hon3 bhaskara.py
1.6570874169509975 -2.0239705338341145
delamaro@dela-ubuntu:~/disciplinas/2018/ICCAmbiental/Aulas/Aula04SalaResumo$ █
= b ** 2 - 4 * a * c
-b + math.sqrt(delta)) / (2 * a)
-b - math.sqrt(delta)) / (2 * a)
x1,x2)
```

Bhaskara com print

```
import math
```

```
a = 308
```

```
b = 113
```

```
c = -1033
```

```
delta = b ** 2 - 4 * a * c
```

```
x1 = (-b + math.sqrt(delta)) / (2 * a)
```

```
x2 = (-b - math.sqrt(delta)) / (2 * a)
```

```
print("O valor da 1a raiz é ", x1)
```

```
print("O valor da 2a raiz é ", x2)
```

Bhaskara com print

```
Terminal
Arquivo Editar Ver Pesquisar Terminal Ajuda
delamaro@dela-ubuntu:~/disciplinas/2018/ICCAmbiental/Aulas/Aula04SalaResumo$ python3 bhaskara.py
0 valor da 1a raiz é 1.6570874169509975
0 valor da 2a raiz é -2.0239705338341145
delamaro@dela-ubuntu:~/disciplinas/2018/ICCAmbiental/Aulas/Aula04SalaResumo$
```

Se algo estiver errado

```
import math
```

```
a = 308
```

```
b = 113
```

```
c = -1033
```

```
delta = b ** 2 - 4 * a * c
```

```
x1 = (-b + math.sqrt(delta)) / (2 * a)
```

```
x2 = (-b - math.sqrt(delta)) / (2 * a)
```

```
print("O valor da 1a raiz é ", x1)
```

```
print("O valor da 2a raiz é ", x2)
```

Se algo estiver errado

```
delamaro@delamaro-OptiPlex-990: ~/disciplinas/2018/ICCAmbiental/Aulas/Aula04SalaR
Arquivo Editar Ver Pesquisar Terminal Ajuda
delamaro@delamaro-OptiPlex-990:~/disciplinas/2018/ICCAmbiental/Aulas/Aula04SalaR
esumo$ python3 bhaskara.py
Traceback (most recent call last):
  File "bhaskara.py", line 9, in <module>
    x1 = (-b + math.sart(delta)) / (2 * a)
AttributeError: module 'math' has no attribute 'sart'
delamaro@delamaro-OptiPlex-990:~/disciplinas/2018/ICCAmbiental/Aulas/Aula04SalaR
esumo$
```

Se algo estiver errado

```
delamaro@delamaro-OptiPlex-990: ~/disciplinas/2018/ICCAmbiental/Aulas/Aula04SalaR
Arquivo Editar Ver Pesquisar Terminal Ajuda
delamaro@delamaro-OptiPlex-990:~/disciplinas/2018/ICCAmbiental/Aulas/Aula04SalaR
esumo$ python3 bhaskara.py
Traceback (most recent call last):
  File "bhaskara.py", line 9, in <module>
    x1 = (-b + math.sart(delta)) / (2 * a)
AttributeError: module 'math' has no attribute 'sart'
delamaro@delamaro-OptiPlex-990:~/disciplinas/2018/ICCAmbiental/Aulas/Aula04SalaR
esumo$
```

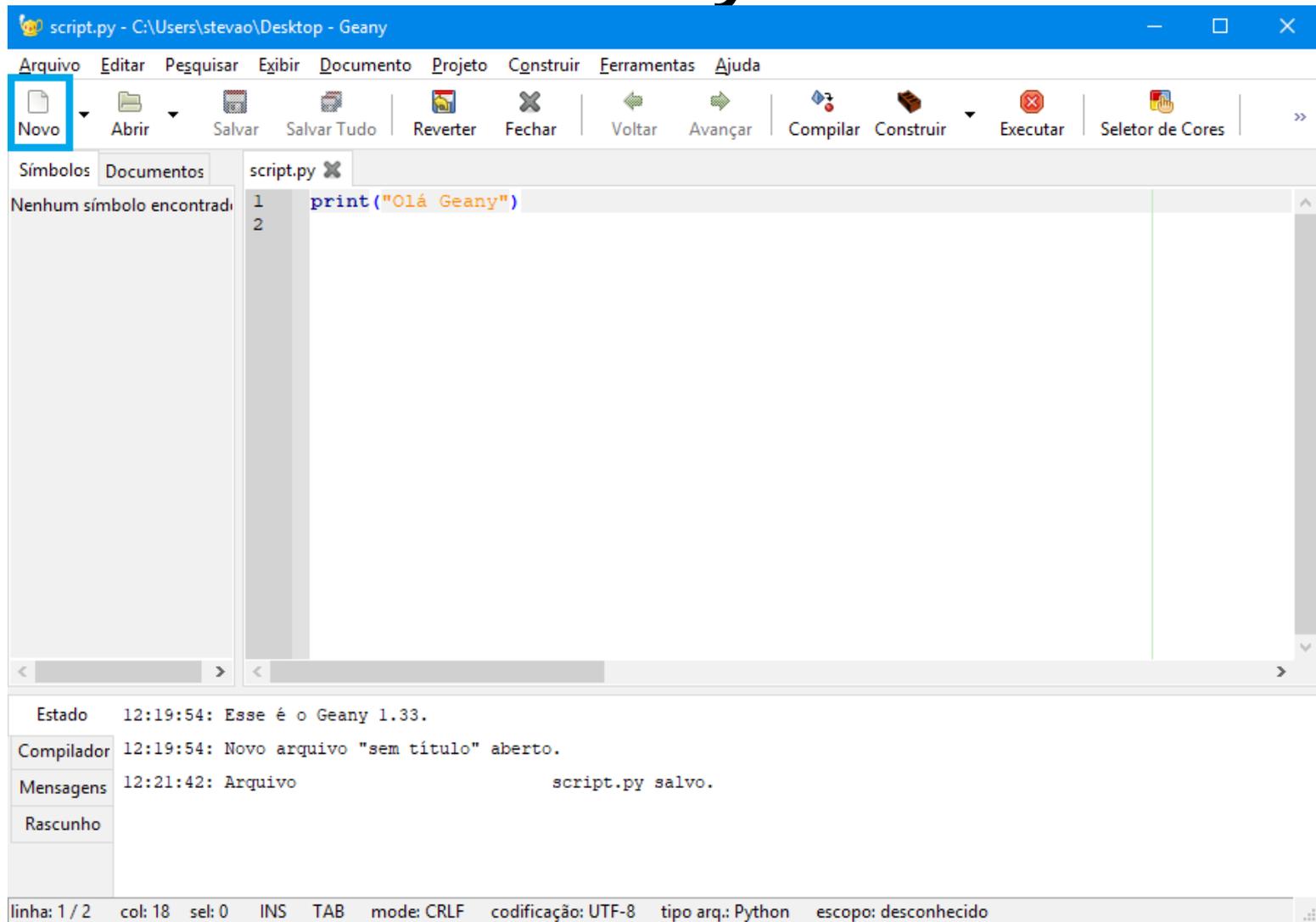
Antes de prosseguir

- Existem ambientes que facilitam a vida do programador
- Não precisa fazer tudo na linha de comando
- Facilita a visualização dos comandos
- Permite executar o programa

Geany

- <https://www.geany.org/Download/Releases>
- Fácil de usar
- Atende nossas necessidades
- Outras opções estão no apêndice do livro

Geany



Voltando ao bhaskara.py

```
import math
```

```
a = 308
```

```
b = 113
```

```
c = -1033
```

```
delta = b ** 2 - 4 * a * c
```

```
x1 = (-b + math.sqrt(delta)) / (2 * a)
```

```
x2 = (-b - math.sqrt(delta)) / (2 * a)
```

```
print("O valor da 1a raiz é ", x1)
```

```
print("O valor da 2a raiz é ", x2)
```

Qual o inconveniente que temos aqui?

Voltando ao bhaskara.py

```
import math
```

```
a = 1.5
```

```
b = 4
```

```
c = -1
```

```
delta = b ** 2 - 4 * a * c
```

```
x1 = (-b + math.sqrt(delta)) / (2 * a)
```

```
x2 = (-b - math.sqrt(delta)) / (2 * a)
```

```
print("O valor da 1a raiz é ", x1)
```

```
print("O valor da 2a raiz é ", x2)
```

Qual o inconveniente que temos aqui?

Voltando ao bhaskara.py

```
import math
```

```
a = 308
```

```
b = 113
```

```
c = -1033
```

Qual o inconveniente que temos aqui?

```
delta = b ** 2 - 4 * a * c
```

Como resolver?

```
x1 = (-b + math.sqrt(delta)) / (2 * a)
```

```
x2 = (-b - math.sqrt(delta)) / (2 * a)
```

```
print("O valor da 1a raiz é ", x1)
```

```
print("O valor da 2a raiz é ", x2)
```

Comando de entrada

- `s = input()`
 - Fica esperando o usuário digitar uma linha de texto. Esse string é atribuído à variável `s`
- Com isso podemos “ler” os valores dos coeficientes

Bhaskara com input

```
import math

a = float(input())
b = float(input())
c = float(input())

delta = b ** 2 - 4 * a * c

x1 = (-b + math.sqrt(delta)) / (2 * a)
x2 = (-b - math.sqrt(delta)) / (2 * a)
print("O valor da 1a raiz é ", x1)
print("O valor da 2a raiz é ", x2)
```

Bhaskara com input

```
import math

a = float(input("Digite o valor de a: "))
b = float(input("Digite o valor de b: "))
c = float(input("Digite o valor de c: "))

delta = b ** 2 - 4 * a * c

x1 = (-b + math.sqrt(delta)) / (2 * a)
x2 = (-b - math.sqrt(delta)) / (2 * a)
print("O valor da 1a raiz é ", x1)
print("O valor da 2a raiz é ", x2)
```

Formatando strings

- `format` é uma função que serve para criar um string no formato que desejamos.
- Muito útil para mostrar resultados numéricos no meio de um string
- Mas pode ser utilizado com qualquer tipo de dados

Como funciona

- `"string de formatação".format(par1,par2,...)`
- Cada parâmetro do format 'entra' em um ponto do string de formatação.
- `"{} -- {} : {}".format(8, 33.5, 'abc')`

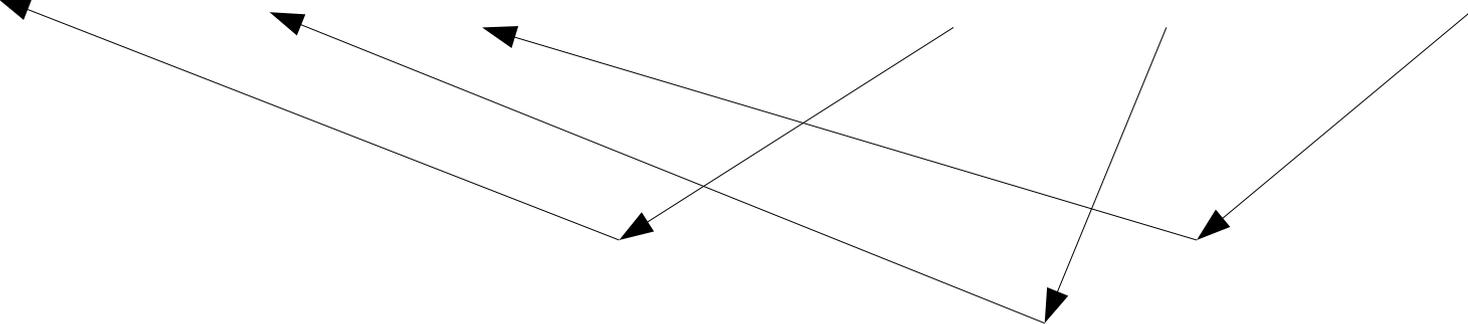
Como funciona

- `"string de formatação".format(par1,par2,...)`
- Cada parâmetro do format 'entra' em um ponto do strig de formatação.
- `"{} -- {} : {}".format(8, 33.5, 'abc')`

Cada um
substituído por
um parâmetro

Como funciona

- `"string de formatação".format(par1,par2,...)`
- Cada parâmetro do format 'entra' em um ponto do strig de formatação.
- `"{} -- {} : {}".format(8, 33.5, 'abc')`



Como funciona

- `"string de formatação".format(par1,par2,...)`
- Cada parâmetro do format 'entra' em um ponto do strig de formatação.
- `"{} -- {} : {}".format(8, 33.5, 'abc')`

`"8 -- 33.5 : abc"`

Para que serve

- Mostrar mensagens mais bonitas
- `print("Seu nome é {} e sua idade é {}".format(nome, idade))`
- `print("As raízes são: {} e {}".format(x1, x2))`

Formatação tipo int

- Em cada “{ }” é possível dizer o tipo de dado que vai ali. No caso de inteiro, o tipo é “d”
- `print("Sua idade é {:d}".format(idade))`

Formatação tipo int

- Em cada “{ }” é possível dizer o tipo de dado que vai ali. No caso de inteiro, o tipo é “d”
- `print("Sua idade é {:d}".format(idade))`

Quer dizer que o parâmetro que vai aqui tem que ser um número inteiro

Formatação tipo int

- Em cada “{ }” é possível dizer o tipo de dado que vai ali. No caso de inteiro, o tipo é “d”
- `print("Sua idade é {:d}".format(idade))`
- `print("Sua idade é {:8d}".format(idade))`

Quer dizer que o parâmetro que vai aqui tem que ser um número inteiro e vai ocupar 8 espaços na saída

Formatação tipo int

- Em cada “{ }” é possível dizer o tipo de dado que vai ali. No caso de inteiro, o tipo é “d”
- `print("Sua idade é {:d}".format(idade))`
- `print("Sua idade é {:8d}".format(idade))`
" 36"
- `print("Sua idade é {:08d}".format(idade))`

Quer dizer que o parâmetro que vai aqui tem que ser um número inteiro e vai ocupar 8 espaços na saída com as posições à esquerda preenchidas com zeros

Formatação tipo int

- Em cada “{ }” é possível dizer o tipo de dado que vai ali. No caso de inteiro, o tipo é “d”
- `print("Sua idade é {:d}".format(idade))`
- `print("Sua idade é {:8d}".format(idade))`
" 36"
- `print("Sua idade é {:08d}".format(idade))`
"00000036"

Formatação tipo float

- Mesma ideia do int
- `print("O valor da raiz é {:.f}".format(x))`
"3.300000" → 6 casas decimais
- `print("O valor da raiz é {:13f}".format(x))`
" 3.300000" → 6 decimais, 13 total
- `print("O valor da raiz é {:013f}".format(x))`
"000003.300000" → 6 decimais, 13 total

Formatação tipo float

- Mesma ideia do int
- `print("O valor da raiz é {:.13.3f}".format(x))`
" 3.300" → 3 decimais, 13 total
- `print("O valor da raiz é {:.3f}".format(x))`
"3.300" → 3 decimais

Exercícios

- Faça os exercícios do capítulo 9
- Entregue os exercícios 3, 4, 5 e 6 da Seção 9.4, usando a função de formatação
- Adicione comentários ao seus programas para explicar como eles funcionam