02

# Cadastrando perfis

## DOWNLOAD

Segue o link (https://s3.amazonaws.com/caelum-online-public/python/05-python.zip) com os arquivos utilizados nesta aula.

### Introdução

Vamos fazer um apanhado do que vimos nos capítulos anteriores para construirmos uma pequena aplicação. Ela vai rodar inteiramente no terminal. O problema é que ainda não aprendemos a capturar a entrada do usuário. Que tal aprendermos agora?

### Capturando o input de usuário

O Python possui uma função simples para capturar a entrada, o `raw_input()` . Imediatamente após sua chamada, a aplicação esperará até que o usuário digite algo. O conteúdo digitado será retornado pela função assim que o usuário teclar ENTER. Normalmente guardamos o retorno de `raw_input()` numa variável. Para ficarmos mais animados, imagine que estamos criando o cadastro de perfis de uma rede social (chique, não?) e começaremos capturando o nome do perfil. Para isso usaremos:

```
>>> nome = raw_input()
```

Nada acontece? Não se desespere, é assim mesmo! Como já foi dito, a função `raw_input()` travará seu console enquanto espera por uma entrada. Vamos digitar o nome "Nico", teclando ENTER logo em seguida. Será que a variável `nome` guarda o texto que acabamos de digitar? Só imprimindo-a para termos certeza:

```
>>> nome
'Nico'
```

Perfeito! Já sabemos capturar dados. Que tal capturar outra informação?

### Convertendo dados

Capturamos o nome do perfil de nossa rede social, mas precisamos capturar também o ano de nascimento:

```
>>> ano = raw_input()
```

Sensacional! Bem, nem tanto. Muitas vezes precisamos saber a **idade** do perfil, sendo assim, supondo que estamos no ano de 2015, basta subtrairmos deste número o ano digitado:

```
>> 2015 - ano
TypeError: unsupported operand type(s) for -: 'int' and 'str'
```

Recebemos um `TypeError`, porque não é possível subtrair uma `string` de um `integer`. Obrigado erro, por existir! Ele nos revela que o retorno da função `raw_input()` é sempre uma string, por isso o erro durante a operação matemática. Mas precisamos que o ano seja considerado um número!

Vimos que o Python possui uma série de funções já prontas para nos auxiliar no dia a dia e podemos contar com a função `int`, que realiza a conversão que precisamos:

```
>>> ano_como_int = int(ano)
>>> 2015 - ano_como_int
15
```

Agora o cálculo funcionou pois temos apenas números envolvidos na operação. Se quisermos converter uma `string` para `float` existe ainda a função `float()`.

## Função para cadastrar dados

No nosso sistema guardaremos os dados do perfil do usuário dentro de uma lista. Tudo bem que guardaremos apenas o nome do perfil, mas já é um bom começo. O código será bem simples: criaremos uma lista e através da já conhecida função `append` adicionaremos o nome do perfil digitado pelo usuário.

Primeiro, vamos implementar rapidamente um esboço do nosso código no terminal. Começaremos criando uma lista vazia. Em seguida, utilizaremos a função `raw_input()` para pegar a entrada. Por fim, adicionaremos na lista o nome do perfil:

```
>>> nomes = []
>>> nome = raw_input()
Nico
>>> nomes.append(nome)
>>> nomes
['Nico']
```

Para deixar o nosso código mais fácil de utilizar, criaremos uma função com esse propósito no próprio terminal! Vamos chamá-la de `cadastrar`. Não se preocupe, o console do Python é inteligente o suficiente para não sair executando nosso código a cada ENTER. Ele espera até que tenhamos uma linha vazia para então executar nosso código:

```
>>> def cadastrar(nomes):
...     print 'Digite o nome:'
...     nome = raw_input()
...     nomes.append(nome)
...
>>>
```

Pronto, nossa função está declarada e pronta para ser usada. Já temos nossa lista `nomes` criada e populada no terminal. Vamos passá-la como parâmetro para a função `cadastrar`:

```
>>> nomes = []
>>> cadastrar(nomes)
Digite o nome
Nico
```

Funcionou perfeitamente!