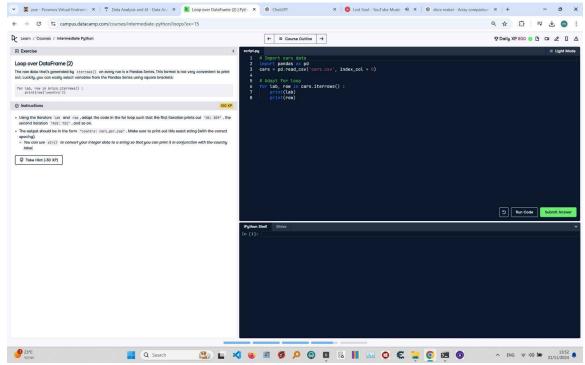
Loop Over DataFrame (2) in Python



Question:

The row data that's generated by `iterrows()` on every run is a Pandas Series. This format is not very convenient to print out. Luckily, you can easily select variables from the Pandas Series using square brackets:

```
```python
for lab, row in brics.iterrows():
 print(row['country'])
```

## \*\*Instructions:\*\*

Using the iterators `lab` and `row`, adapt the code in the for loop such that the first iteration prints out:

`US: 809`, the second iteration `AUS: 731`, and so on.

The output should be in the form `country: cars\_per\_cap`. Make sure to print out this exact string (with the correct spacing).

- You can use `str()` to convert your integer data to a string so that you can print it in conjunction with the country label.

```
Answer:
```

Here is the Python code that solves the problem:

```
Import pandas
import pandas as pd

Load the cars DataFrame
cars = pd.read_csv('cars.csv', index_col=0)

Iterate over rows of cars
for lab, row in cars.iterrows():
 print(f"{lab}: {row['cars_per_cap']}")
```

- \*\*Explanation:\*\*
- 1. \*\*Importing pandas\*\*: The `pandas` library is imported to handle the DataFrame.
- 2. \*\*Loading the DataFrame\*\*: The `cars` DataFrame is loaded from a CSV file (`cars.csv`) with the first column set as the index using `index col=0`.
- 3. \*\*Iterating over rows\*\*: The `iterrows()` method is used to iterate over the rows of the DataFrame. On each iteration:
  - `lab` contains the row label (country code).
  - `row` contains the contents of the row as a Series.
- 4. \*\*Printing formatted output\*\*: The `print()` function uses an f-string to format the output as `country: cars\_per\_cap`, where `country` is the row label and `cars\_per\_cap` is the value from the column `cars\_per\_cap`. The integer value is converted to a string implicitly within the f-string.