

Loop Over DataFrame (2) in Python

The screenshot shows a web browser window displaying a DataCamp exercise titled "Loop over DataFrame (2)". The browser tabs include "Data Analysis and AI - Data An...", "Loop over DataFrame (2) | Pyth...", "ChatGPT", "Lost Soul - YouTube Music", and "docs maker - Array comparison". The URL is "campus.datacamp.com/courses/intermediate-python/loops?ex=15".

The exercise page has a sidebar with "Learn / Courses / Intermediate Python" and a "Course Outline" button. The main content area is titled "Exercise" and "Loop over DataFrame (2)". It contains the following text:

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:

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

Below this is an "Instructions" section with a "100 XP" badge. The instructions are:

- 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.

There is a "Take Hint (-50 XP)" button. To the right is a code editor with a dark theme, showing the following Python code:

```
1 # Import cars data
2 import pandas as pd
3 cars = pd.read_csv('cars.csv', index_col=0)
4
5 # Adapt for loop
6 for lab, row in cars.iterrows():
7     print(lab)
8     print(row)
```

At the bottom of the code editor are buttons for "Run Code" and "Submit Answer". Below the code editor is a "Python Shell" section with a "Gitter" button and a text input area.

****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.