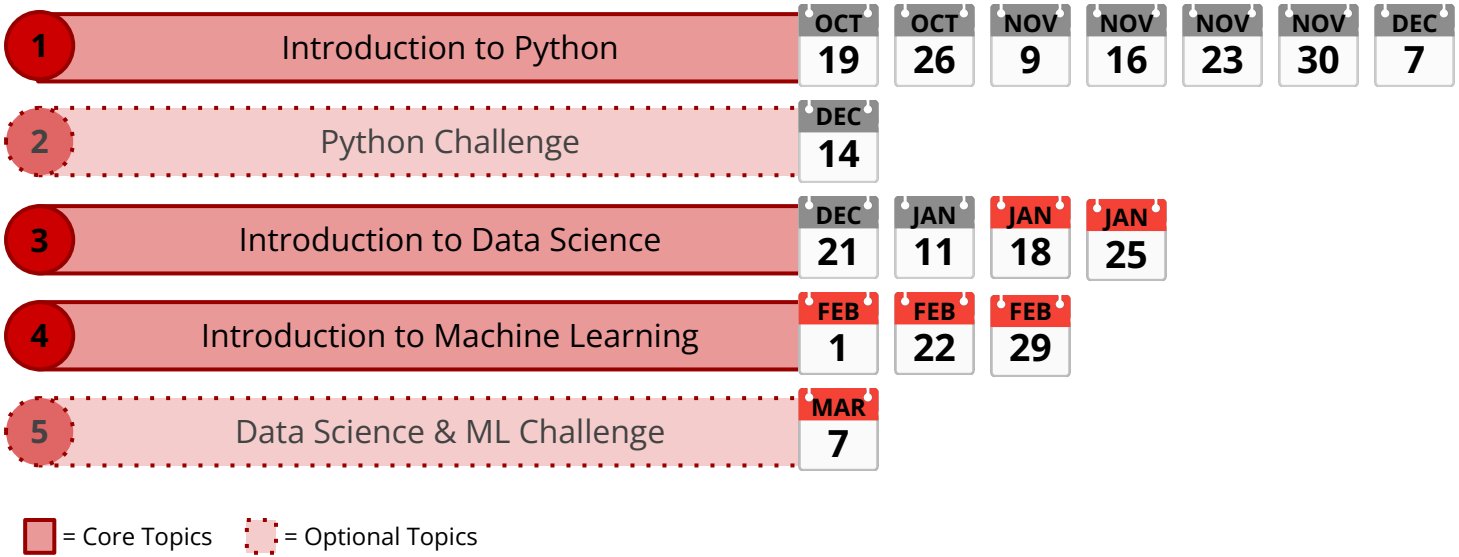
Python for Data Science and Machine Learning

School Year 2023-2024

IST

Course Structure



Jupyter Notebook Setup

In a browser:

```
192.168.10.4:8888
```

Password: **ist**

Recap: Pandas

Pandas is a powerful Python data analysis toolkit.

It provides flexible data structures like **Series** and **DataFrame**.

Widely used in data science, finance, and many other fields.

11.0

```
import pandas as pd
import numpy as np
```

Recap: DataFrame

A **DataFrame** is a two-dimensional data structure with labeled axes (rows and columns).

11.1

```
df = pd.read_csv("titanic_dataset.csv")
df
```

Recap: DataFrame

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
...	...	...	...	...	...	...	...	...	...	...	...	...
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	NaN	S
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	B42	S
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	NaN	S
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C148	C
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	NaN	Q

891 rows × 12 columns

Recap: Selecting DataFrame Data

- The **loc** method in Pandas can be used for selecting rows but also for columns.
- By specifying the row and column labels, you can access specific portions of the dataset.

```
df.loc[0, "Name"]
```

```
df.loc[0:4, "Name"]
```

```
df.loc[:4, "Name"]
```

```
df.loc[4, ["Name", "Age"]]
```

```
df.loc[0:4, ["Name", "Age"]]
```

```
df.loc[:, ["Name", "Age"]]
```

Recap: Boolean Indexing

- **Boolean indexing** in Pandas allows you to select data subsets based on the actual values in the data.

```
df[df.loc[0:9, "Age"] > 30]
```

- **SHORTHAND:** If you wish to **select specific columns** across **all rows** you can use the following:

```
df.loc[:, 'Age']
```

```
df['Age']
```

```
df[df.loc[:, "Age"] > 30]
```

```
df[df["Age"] > 30]
```

Recap: Chaining Indexing

You can **chain** multiple boolean indexing operations by using:

- | for “or”
- **&** for “and”

IMPORTANT! You must use **brackets**!

```
df[(df["Pclass"] == 1) | (df["Pclass"] == 2)]
```

```
df[(df["Pclass"] == 1) & (df["Age"] < 18)]
```

Recap Exercise

Complete the **11.2** , **11.3**, **11.4**, **11.5** & **11.6** programs.

- **11.2**: Select passengers who paid a fare greater than `$500`.
- **11.3**: Select all the third-class passengers that were under `6` years of age.
- **11.4**: Select passengers who are between `70` and `85` years old and paid a fare greater than `$25`.
- **11.5**: Select passengers that embarked from `Cherbourg` (`C`) or from `Queenstown` (`Q`).
- **11.6**: Select passengers that either embarked without siblings or spouses (`SibSp` is `0`), or that did embark with one sibling or spouse and are under the age of `18`.

Solution 11.2

```
df[df["Fare"] > 500]
```

Solution 11.3

```
df[(df["Pclass"] == 3) & (df["Age"] < 6)]
```

Solution 11.4

```
df[(df["Age"] >= 70) & (df["Age"] <= 85) & (df["Fare"] > 25)]
```

Solution 11.5

```
df[(df["Embarked"] == "C") | (df["Embarked"] == "Q")]
```

Solution 11.6

```
df[(df["SibSp"] == 0) | ((df["SibSp"] == 1) & (df["Age"] < 18))]
```

Data Analysis

We can use the **.mean()**, **.count()**, **.max()** and **.min()** functions to analyse our data.

11.7

```
df["Age"].mean()
```

11.8

```
df["Fare"].max()
```

11.9

```
df[df["Survived"] == 1]["Age"].min()
```

Exercise

Complete the **11.10** , **11.11** & **11.12** programs.

- **11.10**: Find the average age of passengers that survived the titanic and compare it to the average age of those that did not survive.
- **11.11**: How many passengers survived the titanic in total?
- **11.12**: What is the smallest fare paid by someone with a parent or a child (`Parch` is greater than 1)?

Solution 11.10

```
survived = df[df["Survived"] == 1]["Age"].mean()
not_survived = df[df["Survived"] == 0]["Age"].mean()

print(survived, not_survived)
```

Solution 11.11

```
df[df["Survived"] == 1]["Name"].count()
```

Solution 11.12

```
df[df["Parch"] >= 1]["Fare"].min()
```

Grouping

Before we analyse our data we can group pieces of information together. We use the **.groupby()** function. We pass in the **column** to group the data with.

11.13

```
df.groupby("Embarked")["Name"].count()
```

11.14

```
df.groupby("Pclass")["Survived"].mean()
```

Exercise

Complete the **11.15**, **11.16** & **11.17** programs.

- **11.15**: Find the average age of passengers that survived the titanic and compare it to the average age of those that did not survive.
- **11.16**: How many passengers survived the titanic in total?
- **11.17**: What is the smallest fare paid by someone with a parent or a child (`Parch` is greater than 1)?

Solution 11.15

```
df.groupby("Embarked")["Fare"].mean()
```

Solution 11.16

```
df.groupby("Pclass")["Age"].mean()
```

Solution 11.17

```
df[df["Age"] > 50].groupby("SibSp")["Age"].count()
```

Solution 11.17.1

```
df[df["Survived"] == 1].groupby("Sex")["Fare"].mean()
```

Quiz Time!

https://ahaslides.com/HFHY2

End of Class

See you all next week!