AN INITIATIVE OF:

science, technology & innovation
Department:
Science, Technology and Innovation
REPUBLIC OF SOUTH AFRICA

CSIR | 80th anniversary
Touching lives through innovation

nicis
NATIONAL INTEGRATED CYBERINFRASTRUCTURE SYSTEM
DIRISA

Introduction to Python and Pandas

23 June 2025

01

Introduction into Python

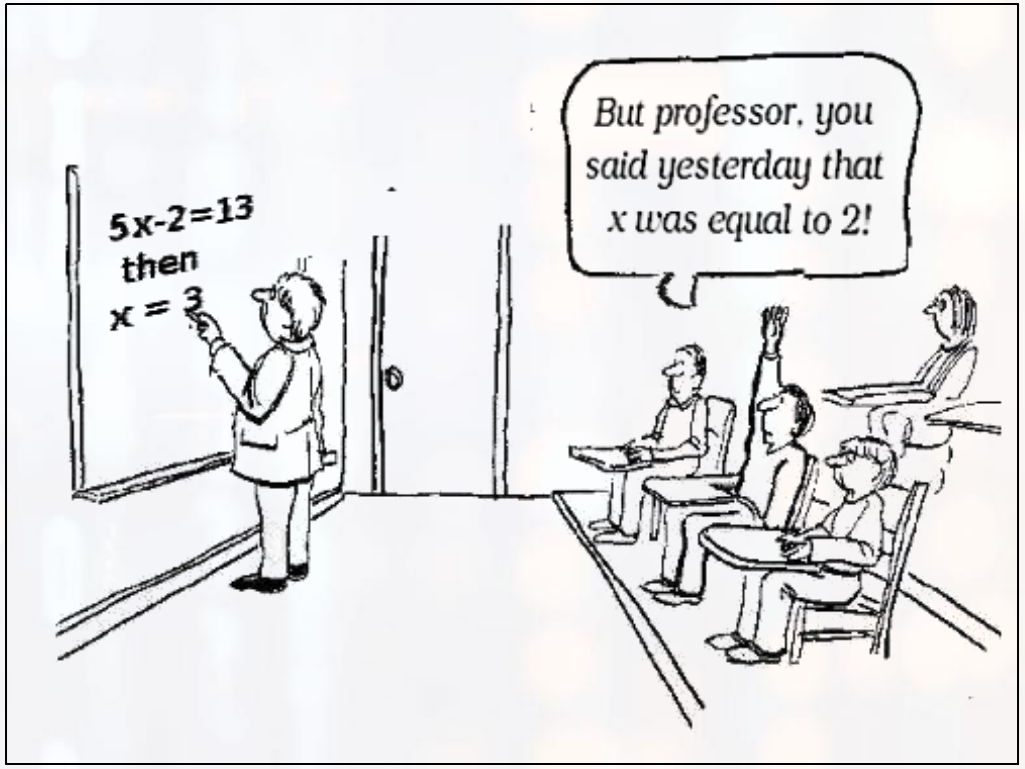
Topics to be discussed

- Variables, Data Types and Print Statements
- Functions
- Lists and Dictionaries
- If and For Statement
- Loop Statements

Variables

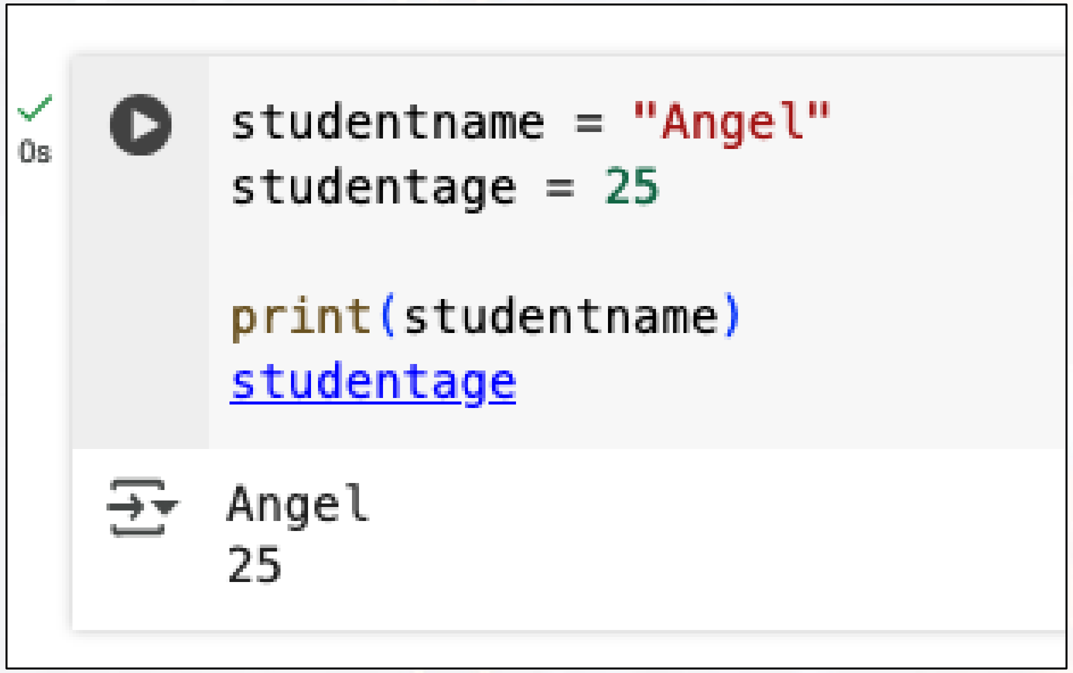
- Variables are containers for storing data values - so that a computer can use it later.

```
name = "Thabo"
```



Print Statement

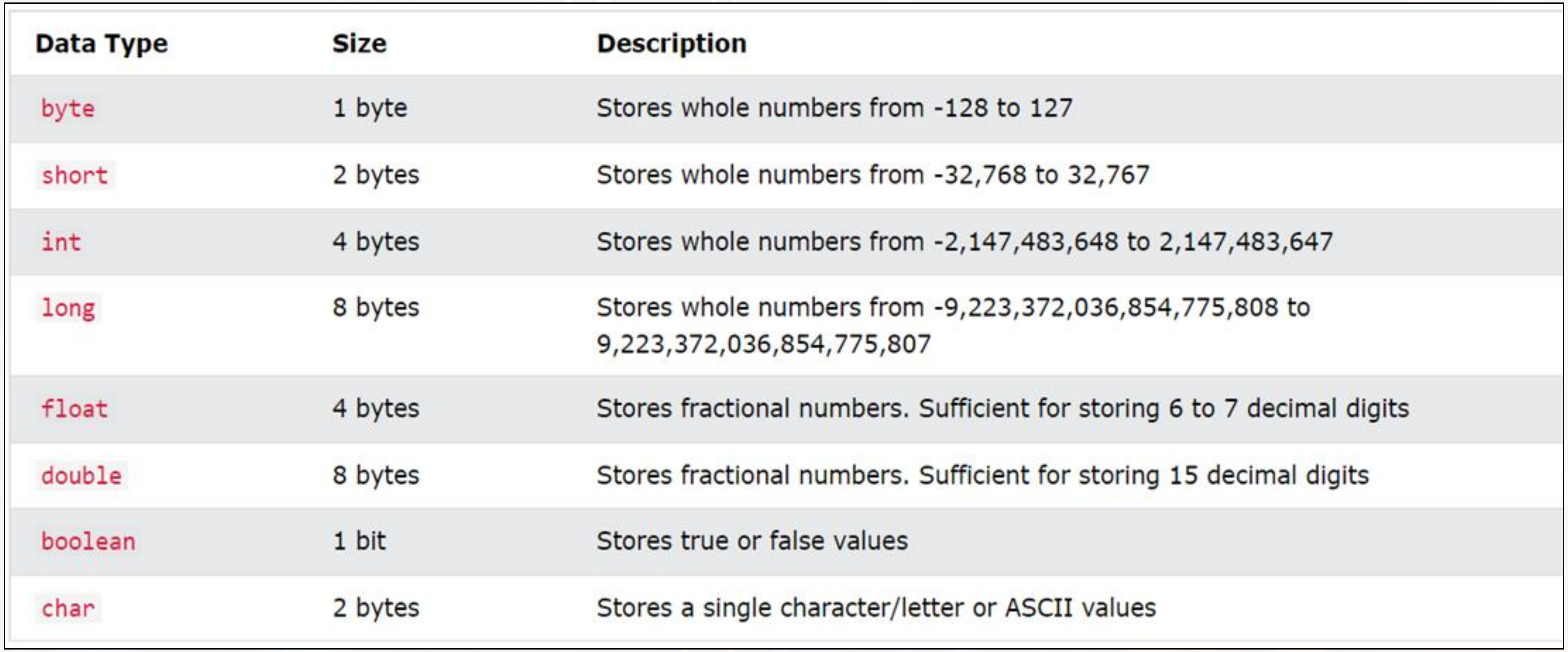
- You can display or print something using the ’Print Statement’ or without using it.



```
studentname = "Angel"
studentage = 25

print(studentname)
studentage
```

```
Angel
25
```

Data Types

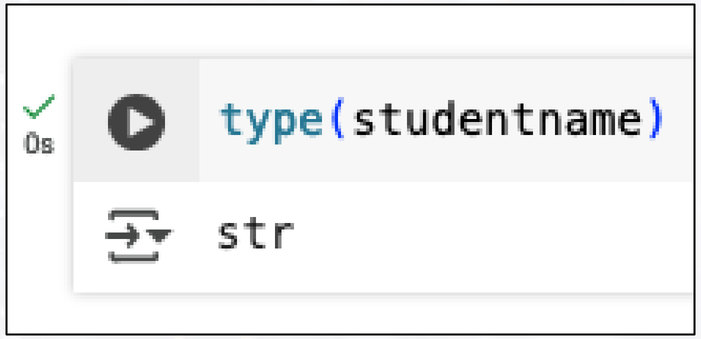


Data Type	Size	Description
`byte`	1 byte	Stores whole numbers from -128 to 127
`short`	2 bytes	Stores whole numbers from -32,768 to 32,767
`int`	4 bytes	Stores whole numbers from -2,147,483,648 to 2,147,483,647
`long`	8 bytes	Stores whole numbers from -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
`float`	4 bytes	Stores fractional numbers. Sufficient for storing 6 to 7 decimal digits
`double`	8 bytes	Stores fractional numbers. Sufficient for storing 15 decimal digits
`boolean`	1 bit	Stores true or false values
`char`	2 bytes	Stores a single character/letter or ASCII values

Data Types

To check variable type:

```
type(variablename)
```



```
type(studentname)
```

```
str
```

Arithmetic Operations

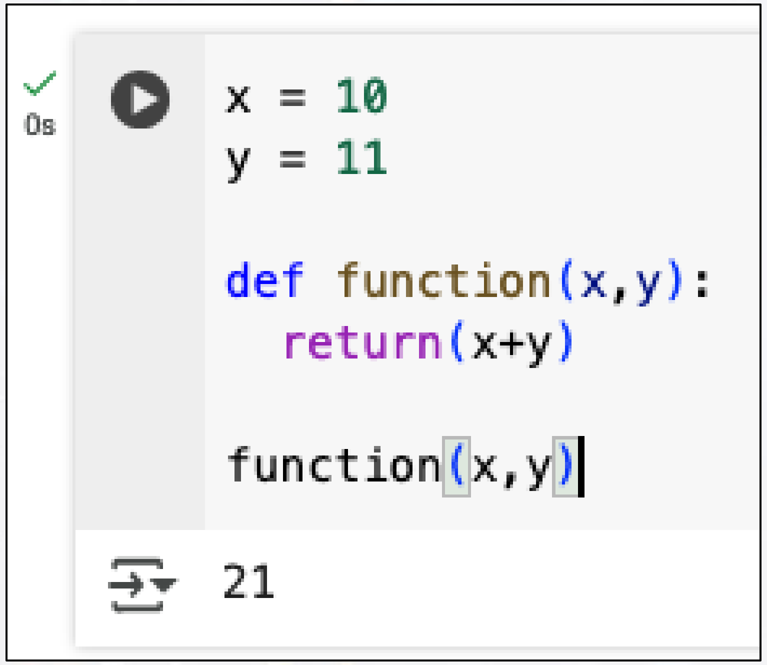
Python Symbol	Math Symbol	Operation	Example
+	+	Addition	15 + 2
-	-	Subtraction	15 – 2
*****	X	Multiplication	15 * 2
/	÷	Division	15 / 2
******	4^2	Exponent / Power	15 ** 2
()	()	Parentheses (brackets/grouping)	(15+2)*2

Other Operations

Python Symbol	Meaning in Plain English	Example	Reads As
=	Assignment (store a value)	**x = 10**	Set x to 10
==	Is equal to?	**x == 10**	Is x equal to 10?
!=	Is not equal to?	**x != 10**	Is x not equal to 10?
>	Greater than	**x > 10**	Is x greater than 10?
<	Less than	**x < 10**	Is x less than 10?
>=	Greater than or equal to	**x >= 10**	Is x greater than or equal to 10?
<=	Less than or equal to	**x <= 10**	Is x less than or equal to 10?
and	Both conditions must be true	**x > 5 and x < 10**	Is x between 5 and 10?
or	At least one condition is true	**x < 5 or x > 10**	Is x outside the range 5 to 10?
not	Negates a condition (opposite)	**not x == 5**	Is x **not** equal to 5?

Functions

- To declare a function, use the def key word.



```
x = 10
y = 11

def function(x,y):
  return(x+y)

function(x,y)
```

```
21
```

IF Statements

"If it’s raining, take an umbrella. Else I will wear shorts"

This is an if statement in real life!

IF Statement

```
weather = "rainy"

if weather == "rainy":
     print("Take an umbrella.")
else:
     print("Take Shorts and Sunglasses")
```

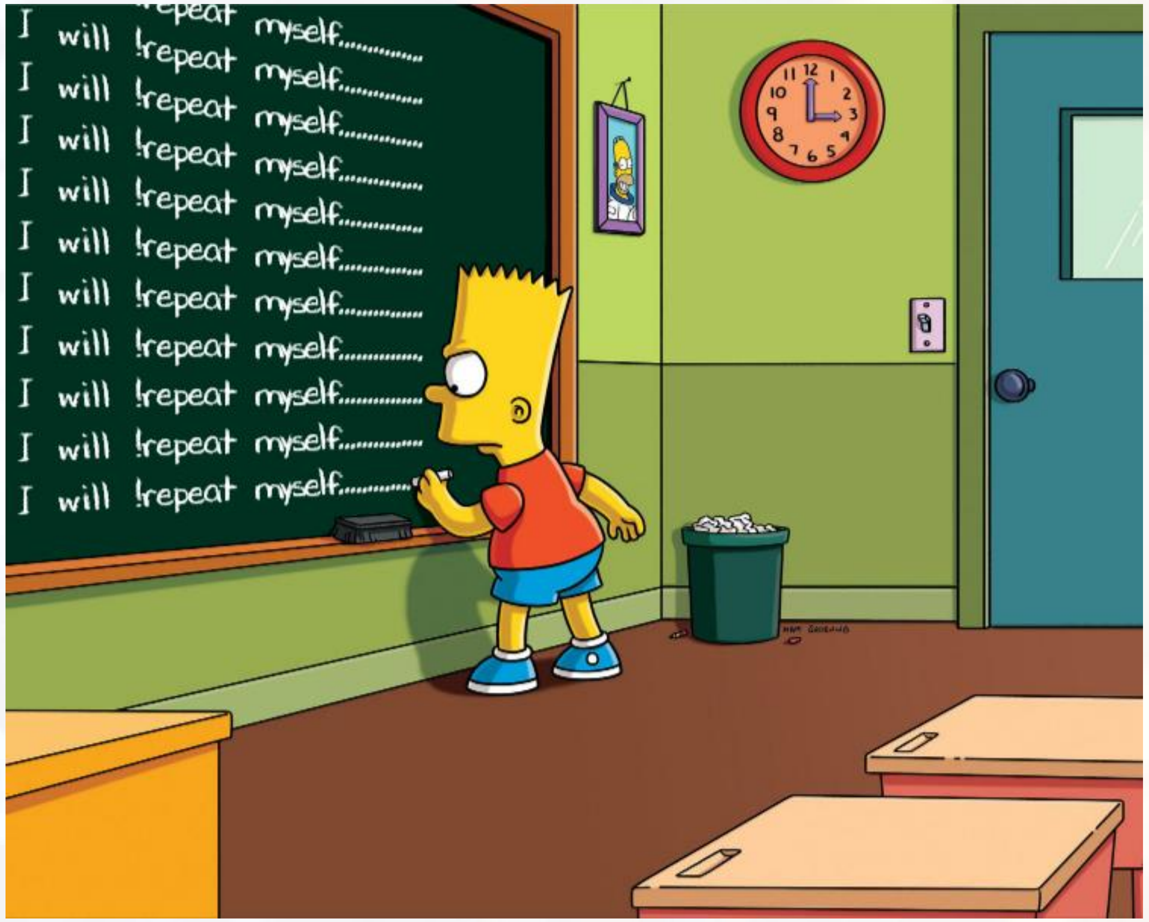
Advance ELIF Statement

```python
import random

# Step 1: Create a list (array) of weather conditions
weather_conditions = ["rainy", "sunny", "cloudy"]

# Step 2: Randomly select a weather condition
weather = random.choice(weather_conditions)

print(f'Today\'s weather is {weather} we should...')

if weather == "rainy":
    print("Take an umbrella.")
elif weather == "sunny" or weather == "hot":
    print("Take shorts and sunglasses.")
elif weather == "cloudy":
    print("You might want a light jacket.")
```

Loops

- A loop is a sequence of instructions that is continually repeated until a certain condition is reached.

- *A loop is like telling the computer: "Keep doing this until I say stop"*


I will repeat myself.........
I will repeat myself.........
I will repeat myself.........
I will repeat myself.........
I will repeat myself.........
I will repeat myself.........
I will repeat myself.........
I will repeat myself.........
I will repeat myself.........
I will repeat myself.........
I will repeat myself.........

While Loop

- **Print a countdown from 10 to 0**

```
# Print a countdown from 10 to 0
```

```
print(10)
print(9)
print(8)
print(7)
print(6)
print(5)
print(4)
print(3)
print(2)
print(1)
print(0)
print("🚀 Blast off!")
```

$\longrightarrow$

```
i = 10
while i > 0:
  print(“T-Minus: ”,i)
  i = i - 1

print("🚀 Blast off!")
```

While Loop

```
# Print a countdown from 10 to 0
i = 10
while i >= 0:
  print(“T-Minus: ”,i)
  i = i - 1

print("🚀 Blast off!")
```

For Loops

```python
# Count from 0 till 10

for i in range(0, 10):
    print("Number:", i)
```

For Loops

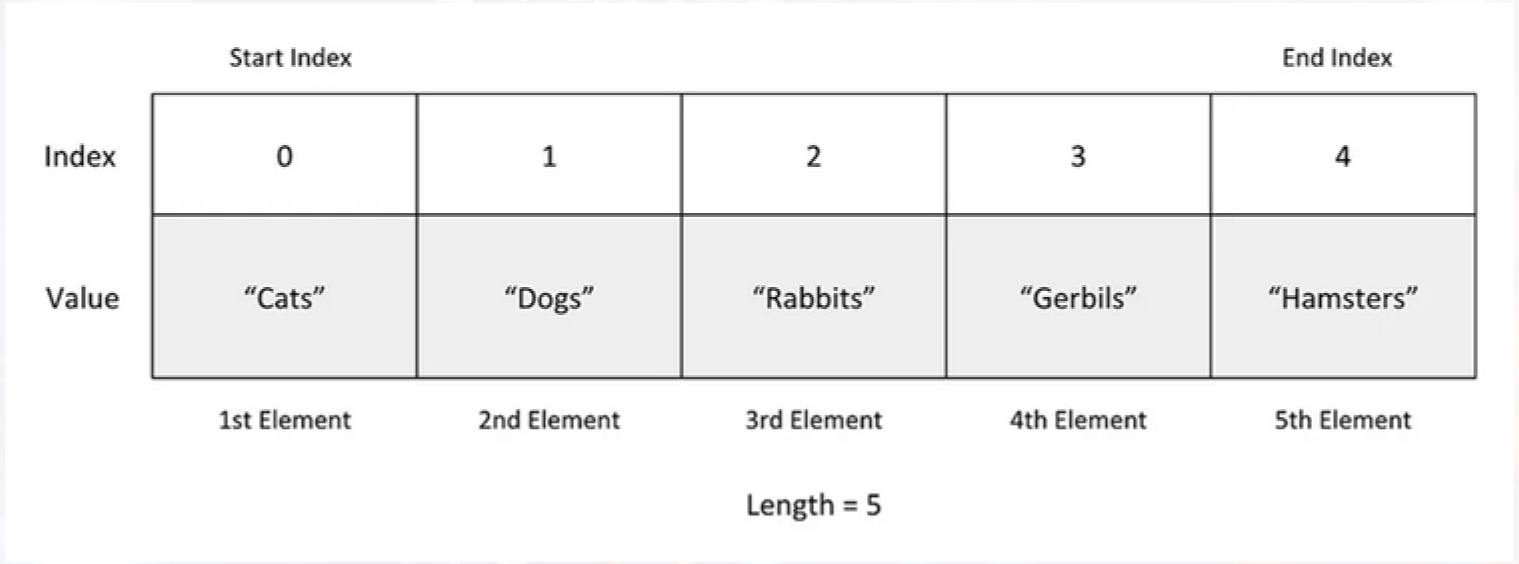
```python
# Count from 0 till 10

for i in range(0, 11):
    print("Number:", i)
```

For Loops - Steps

```python
# count from 0 till 10 in steps of 2

for i in range(0, 12, 2):
    print(f"We are now at {i}")
```

List/Arrays

- A list(array) is a special variable that can hold more than one value at a time.



	Start Index				End Index
Index	0	1	2	3	4
Value	“Cats”	“Dogs”	“Rabbits”	“Gerbils”	“Hamsters”
	1st Element	2nd Element	3rd Element	4th Element	5th Element

Length = 5

Creating and accessing List element

- These elements are separated by a comma and enclosed with square brackets.

```
# Creating an array

proudly_sa = ['Bathu', 'K-Way', 'Drip', 'Maxhosa', 'Drip']

print(proudly_sa[0]) #Bathu
print(proudly_sa[3]) #Maxhosa
```

Accessing All Items using For Loop

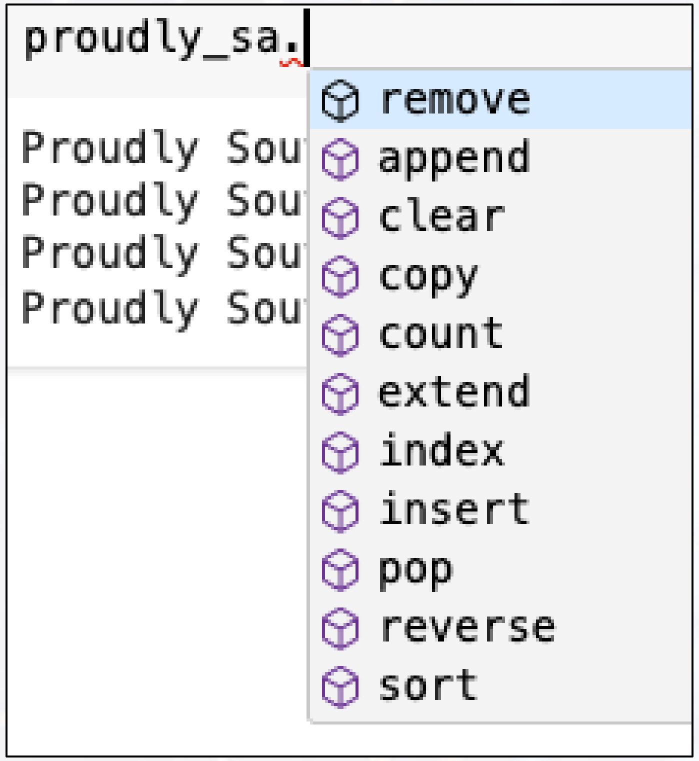
- Could simply type ‘proudly_sa’ to see all elements or use For Loop:

```
# Accessing elements by index

proudly_sa = ['Bathu', 'K-Way', 'Drip', 'Maxhosa', 'Drip']

for brand in proudly_sa:
    print("Proudly South African brand:", brand)
```

Modifying a List

```python
proudly_sa = ['Bathu', 'K-Way', 'Drip', 'Maxhosa', 'Drip']

proudly_sa[2] = "Drip Sportif" # Modify Drip to Drip Sportif
print(proudly_sa)

proudly_sa.remove("K-Way")  # Only removes the first "K-Way"
print(proudly_sa)

proudly_sa.append("Tshepo Jeans") # add a new brand
print(proudly_sa)
```

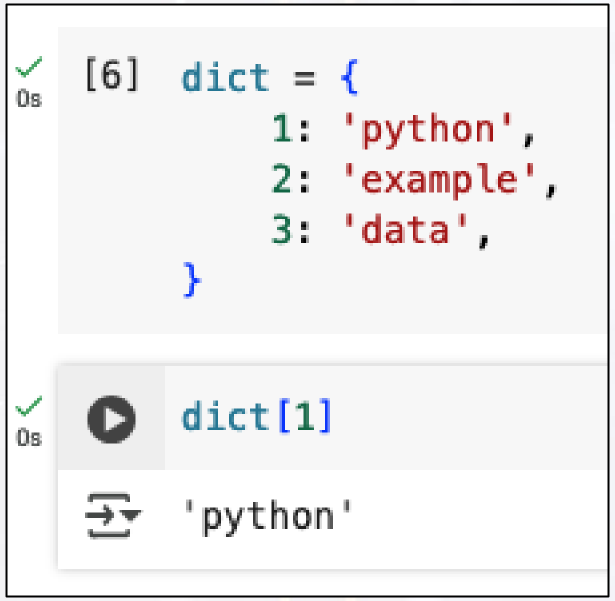
More list functions



```
proudly_sa.

Proudly Sou
Proudly Sou
Proudly Sou
Proudly Sou
```

Dictionary

- A Dictionary in Python is a collection of key-value pairs used to store data values.
- To declare a dictionary:



```
[6] dict = {
        1: 'python',
        2: 'example',
        3: 'data',
    }
```

```
dict[1]
```

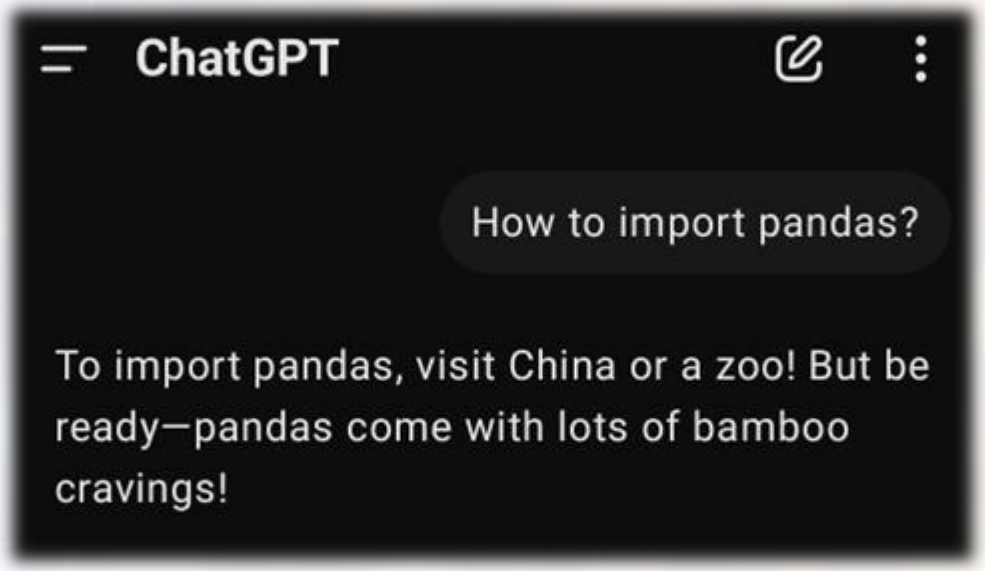
```
'python'
```

02

Introduction to Pandas

About Pandas

- Open-source Python library
- Provides high-performance data manipulation and analysis tools

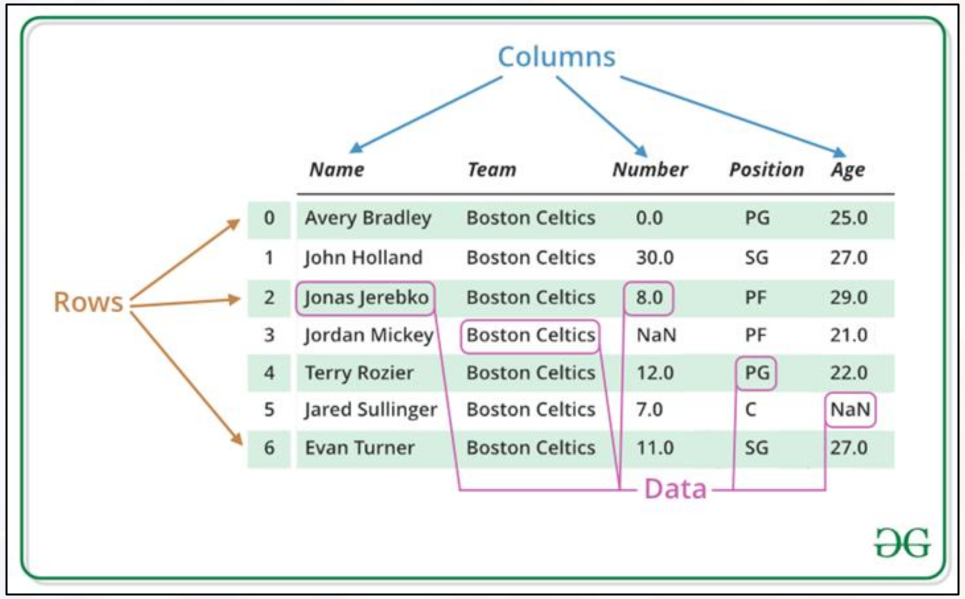


How to import pandas?

To import pandas, visit China or a zoo! But be ready—pandas come with lots of bamboo cravings!

DataFrames in Pandas

- Key Feature of Pandas
- 2-dimensional structure
- Consists of 3 principal components: data, rows and columns
- Labelled axes (rows and columns)
- Can Perform Arithmetic operations on rows and columns



	Name	*Team*	*Number*	*Position*	*Age*
0	Avery Bradley	Boston Celtics	0.0	PG	25.0
1	John Holland	Boston Celtics	30.0	SG	27.0
2	Jonas Jerebko	Boston Celtics	8.0	PF	29.0
3	Jordan Mickey	Boston Celtics	NaN	PF	21.0
4	Terry Rozier	Boston Celtics	12.0	PG	22.0
5	Jared Sullinger	Boston Celtics	7.0	C	NaN
6	Evan Turner	Boston Celtics	11.0	SG	27.0

Creating a Pandas DataFrame

- First you need to import pandas | `import pandas as pd`
- Can be created from:
  * List

  * Dictionary

  * List of Dictionaries

Creating a Pandas DataFrame from a List

```python
# We first import pandas

import pandas as pd


# using a list


proudly_sa = ['Bathu', 'Drip', 'K-Way', 'Maxhosa', 'Tshepo Jeans']


# Create a DataFrame
list_df = pd.DataFrame(proudly_sa)
```

Creating a Pandas DataFrame from a Dictionary

```python
# Now from a dictionary
import pandas as pd


# Define a dictionary


proudly_sa_dict = {
    'Brand': ['Galxboy', 'Bathu', 'Drip', 'Maxhosa', 'Tshepo Jeans'],
    'Year Founded': [2010, 2015, 2019, 2012, 2015],
    'Speciality': ['Streetwear/Fashion', 'Sneakers', 'Sneakers/Lifestyle', 'Luxury Knitwear', 'Premium Denim'],
    'Founder': ['Thatiso Dube', 'Theo Baloyi', 'Lekau Sehoana', 'Laduma Ngxokolo', 'Tshepo Mohlala']
}


# Create a dataframe from a dictionary


dict_df = pd.DataFrame(proudly_sa_dict)
```

Querying a Pandas DataFrame

```
# Now from a dictionary
import pandas as pd


# Define a dictionary


proudly_sa_dict = {
    'Brand': ['Galxboy', 'Bathu', 'Drip', 'Maxhosa', 'Tshepo Jeans'],
    'Founded': [2010, 2015, 2019, 2012, 2015],
    'Speciality': ['Streetwear/Fashion', 'Sneakers', 'Sneakers/Lifestyle', 'Luxury Knitwear', 'Premium Denim'],
    'Founder': ['Thatiso Dube', 'Theo Baloyi', 'Lekau Sehoana', 'Laduma Ngxokolo', 'Tshepo Mohlala']
}


# Create a dataframe from a dictionary


dict_df = pd.DataFrame(proudly_sa_dict)


# querying a dataframe column

founders = dict_df['Founder']
```

Querying a Pandas DataFrame

```python
# Querying a specific value

proudly_sa_dict = {
    'Brand': ['Galxboy', 'Bathu', 'Drip', 'Maxhosa', 'Tshepo Jeans'],
    'Founded': [2010, 2015, 2019, 2012, 2015],
    'Speciality': ['Streetwear/Fashion', 'Sneakers', 'Sneakers/Lifestyle', 'Luxury Knitwear', 'Premium
Denim'],
    'Founder': ['Thatiso Dube', 'Theo Baloyi', 'Lekau Sehoana', 'Laduma Ngxokolo', 'Tshepo Mohlala']
}

# Create a dataframe from a dictionary
dict_df = pd.DataFrame(proudly_sa_dict)
# Querying owner of Drip
drip_owner = dict_df['Founder'][2]
```

DataFrames from External Sources

- In the real world, a Pandas DataFrame is usually created from external sources such as:

Databases

Excel Files

CSV Files

And More..

DataFrame From CSV

```python
# imports
import pandas as pd


# specify the file path
my_file_path = 'drive/MyDrive/Colab Notebooks/hcd/data/sa_loan_eligibility_data.csv'


# Read the CSV file into a DataFrame
df = pd.read_csv(my_file_path)


df.head() # show the first 5 rows
df.tail() # last 5 rows
```

DataFrame From Excel

```python
# imports
import pandas as pd


# specify the file path
my_file_path = 'drive/MyDrive/Colab Notebooks/hcd/data/sa_loan_eligibility_data.xlsx


# Read the CSV file into a DataFrame
df = pd.read_excel(my_file_path)


df.head() # show the first 5 rows
df.tail() # last 5 rows
```

Dealing with Incomplete Data

Most real-world data is not perfect and sometimes:

1. Sheets don't have headers
2. There are more than one sheet in the Excel File
3. Information at the top of Excel sheet before table
4. Headers exists, but we would like to rename them to fit our narrative.

Excel/Data with no Headers

```
# You can specify to pandas that there are no headers:
df_no_headers = pd.read_excel(no_headers_file, header=None)

# It then defaults to numbering the columns
df_no_headers.head()
```

```
# You can then create your 'own' headers
header_names = ['Full_Name', 'Province' , 'Age' , 'Income',….']

# specify your header names
df_with_custom_headers = pd.read_excel(no_headers_file, header=None,
names=header_names)

df_with_custom_headers.head()
```

Excel with multiple sheets

```
# To get a specific sheet. You specify the sheet name

df_soccer_stats = pd.read_excel(many_sheets_file, sheet_name='soccer_probability')

df_soccer_stats.head()
```

Information at the top of Excel sheet before table

```
# To get the correct info, skip the rows that don't include the real data

df_clean_excel = pd.read_excel(dirty_excel_file, skiprows=15)

df_clean_excel.head()
```

Renaming Columns

```
# What if we wanted to rename 'Income' to 'Monthly Income'

df = df.rename(columns={'Income': 'Monthly Income'})

df.head()
```

Exporting DataFrames

After processing, data often needs to be exported for further use. We'll look into:

1. Exporting a DataFrame to an Excel File
2. Exporting a DataFrame to a CSV file

Exporting a DataFrame to a CSV file [df.to_csv()]

```
# Let us pretend the dictionary dataframe is from some process...

# first let us specify the file path
file_path = 'drive/MyDrive/Colab Notebooks/hcd/data/proudly_sa.csv'


# write to a csv
dict_df.to_csv(file_path)
```

Exporting a DataFrame to a CSV file [df.to_excel()]

```
# Let us pretend the dictionary dataframe is from some process...

# first let us specify the file path
file_path = 'drive/MyDrive/Colab Notebooks/hcd/data/proudly_sa.xlsx


# write to a excel
dict_df.to_excel(file_path)
```

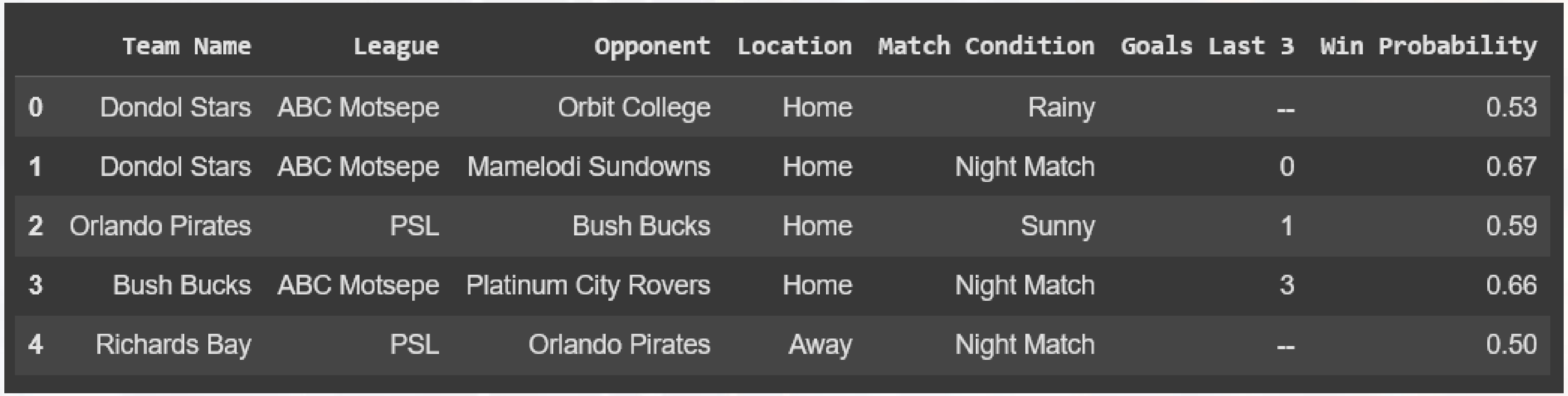
Modifying DataFrames

In most instances, the data from a source is not perfect requires modifications.

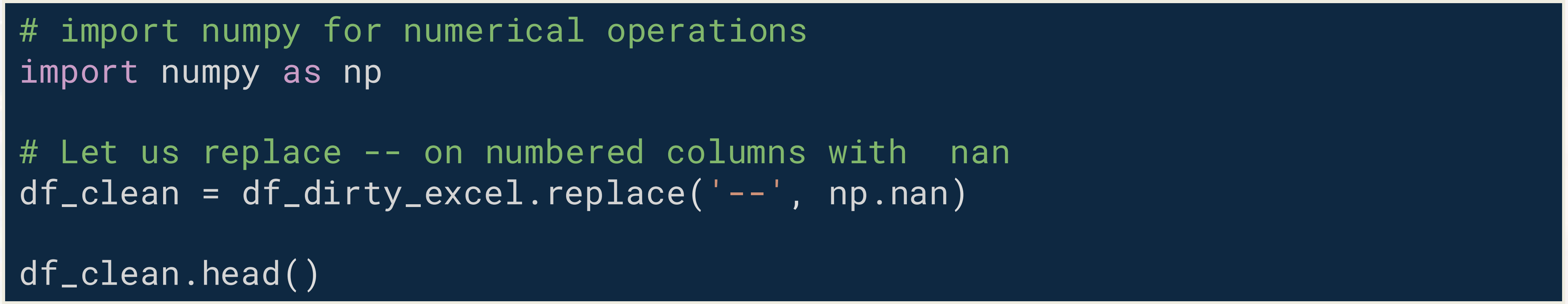
1. Replacing and removing entries
2. Removing unwanted rows and columns

[Covered in Data Exploration in more detail]

Replacing all df entries

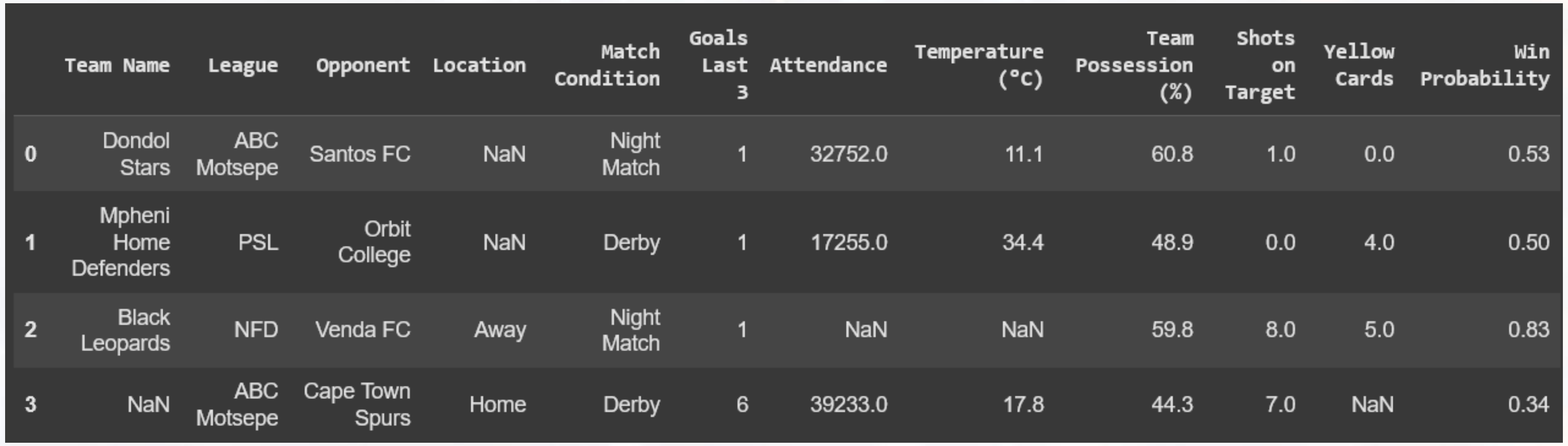


	Team Name	League	Opponent	Location	Match Condition	Goals Last 3	Win Probability
0	Dondol Stars	ABC Motsepe	Orbit College	Home	Rainy	--	0.53
1	Dondol Stars	ABC Motsepe	Mamelodi Sundowns	Home	Night Match	0	0.67
2	Orlando Pirates	PSL	Bush Bucks	Home	Sunny	1	0.59
3	Bush Bucks	ABC Motsepe	Platinum City Rovers	Home	Night Match	3	0.66
4	Richards Bay	PSL	Orlando Pirates	Away	Night Match	--	0.50



```python
# import numpy for numerical operations
import numpy as np

# Let us replace -- on numbered columns with  nan
df_clean = df_dirty_excel.replace('--', np.nan)

df_clean.head()
```

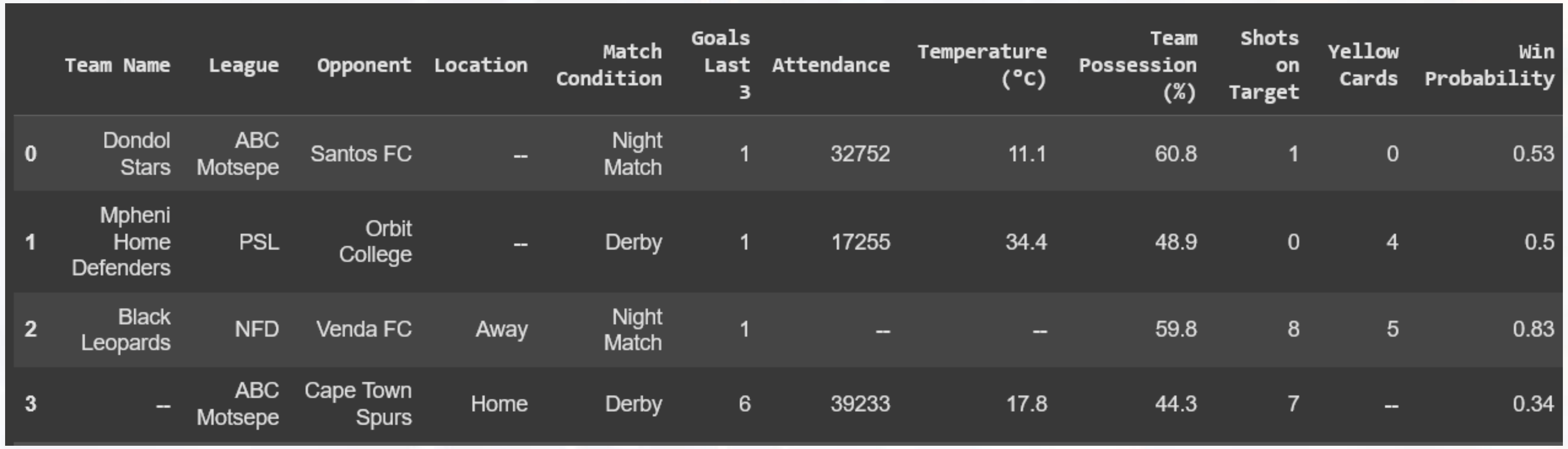
Replacing from a specific column



	Team Name	League	Opponent	Location	Match Condition	Goals Last 3	Attendance	Temperature (°C)	Team Possession (%)	Shots on Target	Yellow Cards	Win Probability
0	Dondol Stars	ABC Motsepe	Santos FC	NaN	Night Match	1	32752.0	11.1	60.8	1.0	0.0	0.53
1	Mpheni Home Defenders	PSL	Orbit College	NaN	Derby	1	17255.0	34.4	48.9	0.0	4.0	0.50
2	Black Leopards	NFD	Venda FC	Away	Night Match	1	NaN	NaN	59.8	8.0	5.0	0.83
3	NaN	ABC Motsepe	Cape Town Spurs	Home	Derby	6	39233.0	17.8	44.3	7.0	NaN	0.34

```python
# Replacing in specific columns.

df_clean['Team Name'] = df_clean['Team Name'].replace(np.nan, 'Unknown Team')
df_clean['Location'] = df_clean['Location'].replace(np.nan, 'Unknown Location')

df_clean.head()
```

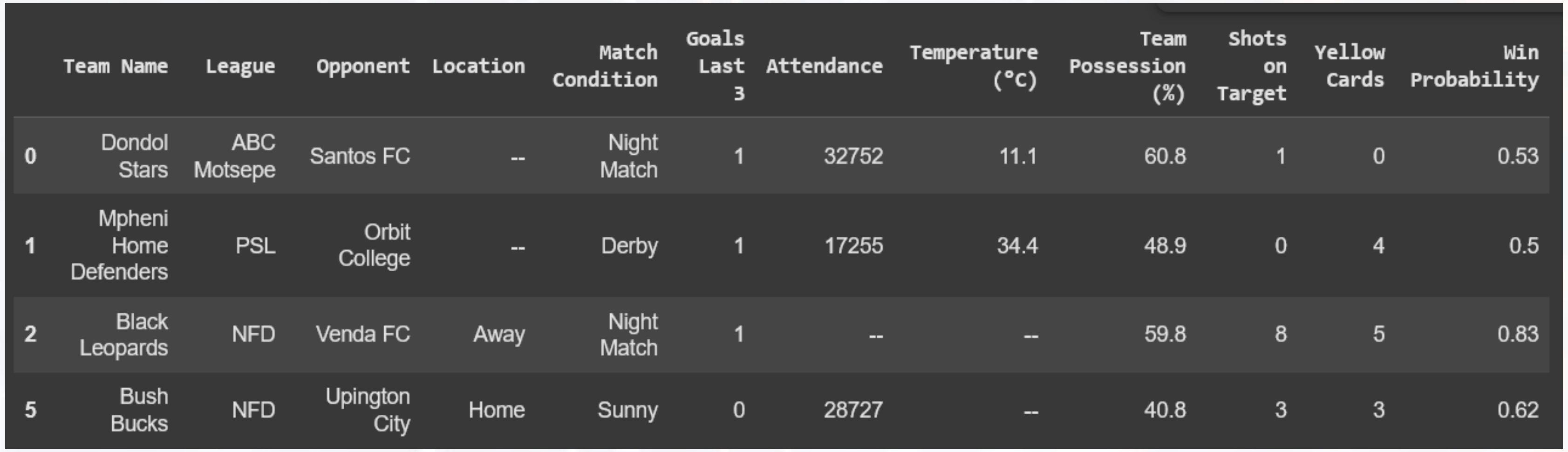
Removing unwanted rows



	Team Name	League	Opponent	Location	Match Condition	Goals Last 3	Attendance	Temperature (°C)	Team Possession (%)	Shots on Target	Yellow Cards	Win Probability
0	Dondol Stars	ABC Motsepe	Santos FC	--	Night Match	1	32752	11.1	60.8	1	0	0.53
1	Mpheni Home Defenders	PSL	Orbit College	--	Derby	1	17255	34.4	48.9	0	4	0.5
2	Black Leopards	NFD	Venda FC	Away	Night Match	1	--	--	59.8	8	5	0.83
3	--	ABC Motsepe	Cape Town Spurs	Home	Derby	6	39233	17.8	44.3	7	--	0.34

```
# If we include the original dataframe, we will only get instances where the result
is true


df_clean = df_dirty_excel[~df_dirty_excel['Team Name'].isin(['--'])]

df_clean
```

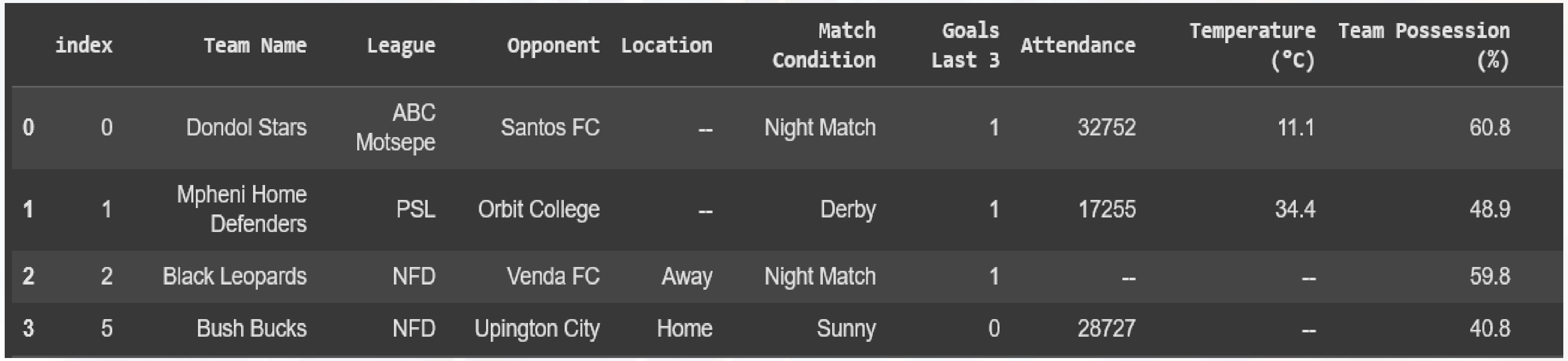
After removing the rows, the index needs to be reset



	Team Name	League	Opponent	Location	Match Condition	Goals Last 3	Attendance	Temperature (°C)	Team Possession (%)	Shots on Target	Yellow Cards	Win Probability
0	Dondol Stars	ABC Motsepe	Santos FC	--	Night Match	1	32752	11.1	60.8	1	0	0.53
1	Mpheni Home Defenders	PSL	Orbit College	--	Derby	1	17255	34.4	48.9	0	4	0.5
2	Black Leopards	NFD	Venda FC	Away	Night Match	1	--	--	59.8	8	5	0.83
5	Bush Bucks	NFD	Upington City	Home	Sunny	0	28727	--	40.8	3	3	0.62

```
# If we include the original dataframe, we will only get instances where the result
is true


df_clean = df_clean.rest_index()

df_clean
```

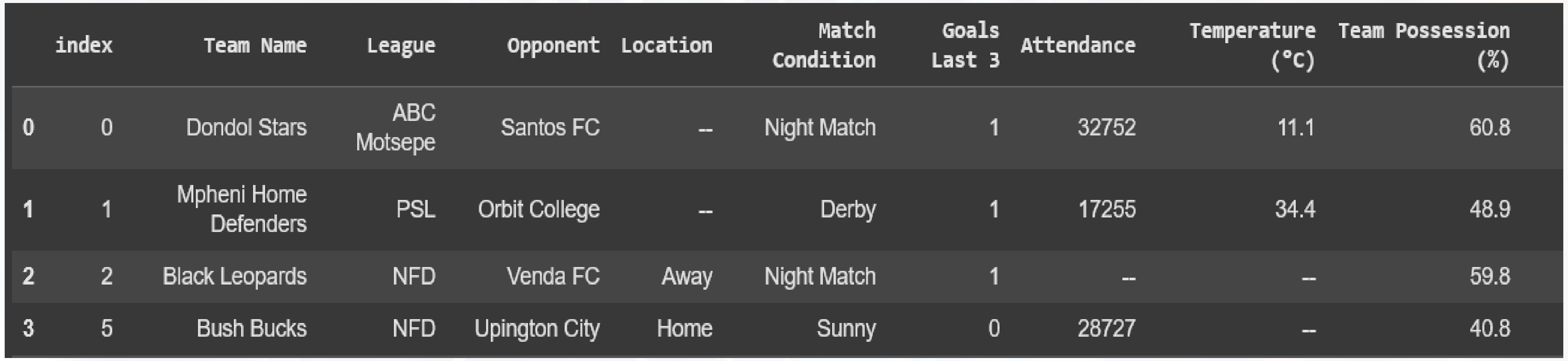
Removing Columns | Using Drop



	index	Team Name	League	Opponent	Location	Match Condition	Goals Last 3	Attendance	Temperature (°C)	Team Possession (%)
0	0	Dondol Stars	ABC Motsepe	Santos FC	--	Night Match	1	32752	11.1	60.8
1	1	Mpheni Home Defenders	PSL	Orbit College	--	Derby	1	17255	34.4	48.9
2	2	Black Leopards	NFD	Venda FC	Away	Night Match	1	--	--	59.8
3	5	Bush Bucks	NFD	Upington City	Home	Sunny	0	28727	--	40.8

```
# We want to remove League and index
# Using Drop

df = df_clean.drop(columns=['League', 'index'])
```

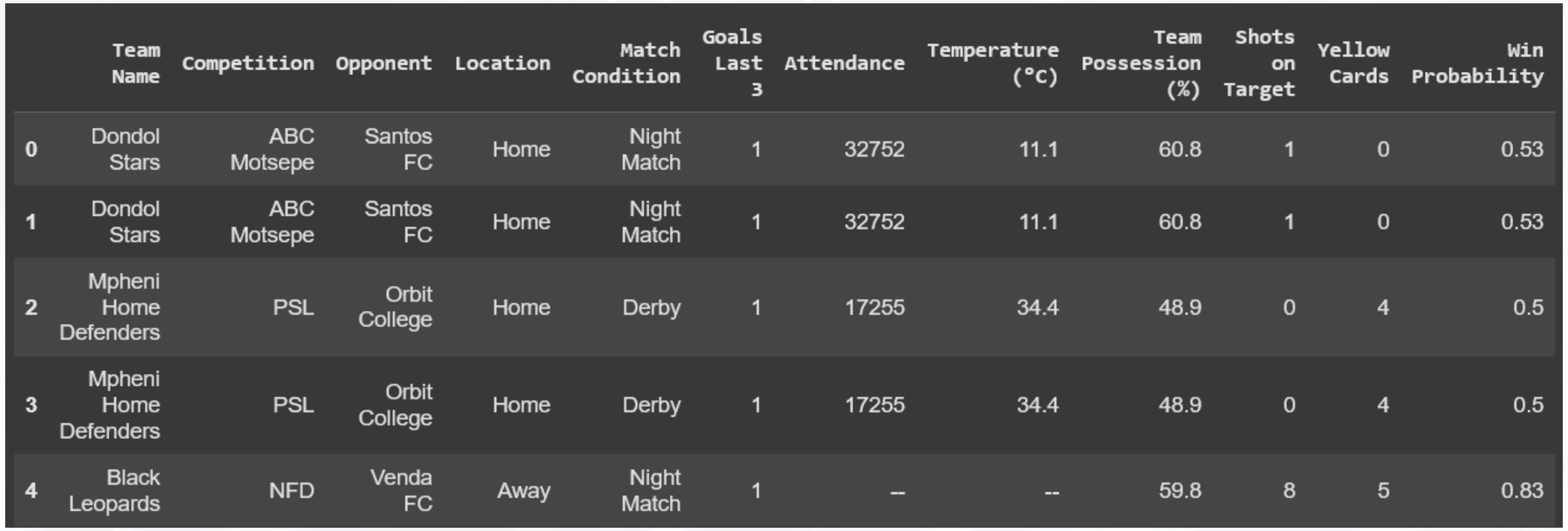
Removing Columns | Using Filter



	index	Team Name	League	Opponent	Location	Match Condition	Goals Last 3	Attendance	Temperature (°C)	Team Possession (%)
0	0	Dondol Stars	ABC Motsepe	Santos FC	--	Night Match	1	32752	11.1	60.8
1	1	Mpheni Home Defenders	PSL	Orbit College	--	Derby	1	17255	34.4	48.9
2	2	Black Leopards	NFD	Venda FC	Away	Night Match	1	--	--	59.8
3	5	Bush Bucks	NFD	Upington City	Home	Sunny	0	28727	--	40.8

```
# We want to keep only Team Name, Opponent and Win Probability
# Using Filter

df = df_clean.filter(['Team Name', 'Opponent', 'Win Probability'], axis=1) # axis=1 for colunms
```

Removing Duplicates



	Team Name	Competition	Opponent	Location	Match Condition	Goals Last 3	Attendance	Temperature (°C)	Team Possession (%)	Shots on Target	Yellow Cards	Win Probability
0	Dondol Stars	ABC Motsepe	Santos FC	Home	Night Match	1	32752	11.1	60.8	1	0	0.53
1	Dondol Stars	ABC Motsepe	Santos FC	Home	Night Match	1	32752	11.1	60.8	1	0	0.53
2	Mpheni Home Defenders	PSL	Orbit College	Home	Derby	1	17255	34.4	48.9	0	4	0.5
3	Mpheni Home Defenders	PSL	Orbit College	Home	Derby	1	17255	34.4	48.9	0	4	0.5
4	Black Leopards	NFD	Venda FC	Away	Night Match	1	--	--	59.8	8	5	0.83

```
# Removing Duplicates

df_no_dups = df.drop_duplicates(subset=['Team Name', 'Opponent', 'Location', 'Win
Probability'], keep='first')

# Have an option of completely dropping by setting keep = False
# You can keep the last entry by specifying keep = 'last'

df_no_dups.head()
```

Dealing with Dates

```
# Change the string date to type datetime

df['Match Date'] = pd.to_datetime(df['Match Date'], format="%Y-%m-%d")
```

```
# Let us combine all the separate columns of date into a single Match Date

df['Match Date'] = pd.to_datetime(df[['Year', 'Month', 'Day']], format="%Y-%m-%d",
errors='coerce')

# coerce check for invalid dates
```

return 0;

NATIONAL INTEGRATED
CYBERINFRASTRUCTURE SYSTEM

DIRISA