



# PYTHON

## PRACTICAL MODULE



### PROCESSING EXCEL-BASED STATISTICAL DATA



**PBO LANJUT**

**INFORMATIC ENGINEERING**

## **KATA PENGANTAR**

Dengan mengucapkan puja dan puji syukur kehadiran Allah SWT, atas limpahan rahmat dan hidayah-Nya, kami dapat menyusun Modul Praktikum Mengolah Data Statistik Berbasis Excel dengan Python ini. Kami berharap modul ini dapat memotivasi dan menjadi acuan bagi pembaca yang ingin mempelajari Data Statistik menggunakan Excel dan Python.

Bahan ajar ini disusun berdasarkan pengalaman serta berpedoman pada berbagai sumber referensi, baik dari buku, tutorial, website, maupun karya ilmiah. Modul ini berisi teori, contoh studi kasus, dan penjelasan mendetail yang memudahkan pembaca dalam mempelajarinya. Dalam modul ini, kami akan membahas tentang cara mengolah data statistik berbasis Excel dengan menggunakan Python. Kami berharap, dengan contoh-contoh yang diberikan secara sederhana, pembaca dapat memahami dan mengerti materi yang disampaikan.

Kami menyampaikan permohonan maaf apabila terdapat kesalahan dalam penulisan atau penyampaian materi dalam modul ini. Kami sangat mengharapkan kritik dan saran yang membangun dari para pembaca untuk kemajuan modul ini dan perbaikan di masa mendatang. Semoga modul ini bermanfaat bagi semua yang mempelajarinya. Terima kasih.

Cirebon, Mei 2024

Tim Penyusun

## DAFTAR ISI

KATA PENGANTAR .....	ii
DAFTAR ISI .....	iii
BAB 1 PENDAHULUAN.....	1
A. Latar Belakang.....	1
B. Rumusan Masalah .....	2
C. Tujuan.....	2
BAB 2 CODE EDITOR .....	3
A. Pengertian Code Editor.....	3
B. Mengenal Visual Studio Code (VS Code).....	3
C. Mengenal Jupyter Notebook.....	4
D. Langkah-langkah Penggunaan Jupyter Notebook dalam VS Code....	4
BAB 3 LIBRARY .....	6
A. Pengertian Library .....	6
B. Mengenal Pandas.....	6
C. Mengenal Numpy .....	10
D. Mengenal Matplotlib dan Seaborn .....	11
E. Mengenal Tabulate .....	15
F. Mengenal Openpyxl .....	16
BAB 4 LATIHAN MENGOLAH DATA STATISTIK EXCEL DENGAN PYTHON .....	18
A. Implementasi Library Pandas .....	18
B. Implementasi Openpyxl dengan Pandas .....	30
C. Fungsi yang Digunakan untuk Menambahkan, Menghapus, dan Mengupdate Data Statistik Excel.....	32
D. Input dan Output Menambahkan Data Statistik Excel .....	35
E. Input dan Output Menhapus Data Statistik Excel.....	36
F. Input dan Output Mengupdate Data Statistik Excel.....	36
BAB 5 PENUTUP .....	38
A. Kesimpulan.....	38
B. Saran .....	38
DAFTAR PUSTAKA.....	39
TENTANG PENULIS.....	40

# **BAB 1**

## **PENDAHULUAN**

### **A. Latar Belakang**

Python adalah bahasa pemrograman interpretatif yang dianggap mudah dipelajari serta berfokus pada keterbacaan kode. Dengan kata lain, Python diklaim sebagai bahasa pemrograman yang memiliki sintaksis yang sangat jelas, lengkap, dan mudah dipahami. Python mendukung berbagai paradigma pemrograman, termasuk pemrograman berorientasi objek, pemrograman imperatif, dan pemrograman fungsional. Sebagai bahasa pemrograman multi-paradigma, Python dapat digunakan untuk berbagai keperluan pengembangan perangkat lunak dan dapat berjalan di berbagai platform sistem operasi.

Jupyter adalah organisasi non-profit yang mengembangkan perangkat lunak interaktif dalam berbagai bahasa pemrograman. Salah satu produk utamanya adalah Jupyter Notebook, sebuah aplikasi web open-source yang memungkinkan pengguna untuk membuat dan berbagi dokumen interaktif yang berisi kode live, persamaan, visualisasi, dan teks naratif yang kaya. Ilustrasinya begini: sebelumnya, kita biasanya membagikan kode dan dokumen secara terpisah. Kode-kode disatukan dalam sebuah librari/aplikasi/proyek (seperti Visual Studio, Eclipse, dsb), sementara dokumen dibuat dengan penyunting kata. Dalam dokumen tersebut bisa ditampilkan cuplikan kode, hasil eksekusi, dan visualisasi lainnya dari program kita. Jupyter Notebook menyatukan semua ini dalam satu file interaktif, yang berisi teks/narasi, kode hidup, persamaan, hasil eksekusi, gambar statis, dan visualisasi grafis. Kelebihan lainnya adalah notebook dapat dijalankan ulang oleh siapapun yang membukanya, untuk mereproduksi eksekusi kode di dalamnya.

Pandas adalah library Python yang berfungsi untuk manipulasi dan analisis data, terutama data tabular seperti yang sering ditemui dalam spreadsheet atau relational database. Pandas, yang merupakan singkatan dari Python Data Analysis Library, adalah sebuah open source library dengan lisensi BSD yang menyediakan banyak alat untuk kebutuhan analisis data, dan manipulasi.

Excel adalah salah satu perangkat lunak pengolah data yang sangat populer dan sering digunakan untuk keperluan statistik, analisis data, dan pembuatan laporan. Dengan kemampuan yang luas dalam mengelola data tabular, Excel menyediakan berbagai fungsi dan fitur yang membantu dalam analisis data. Namun, untuk analisis data yang lebih kompleks dan pengolahan data yang lebih efisien, menggabungkan penggunaan Excel dengan kemampuan pemrograman Python dan library seperti Pandas menjadi solusi yang sangat efektif.

Dengan menggunakan Jupyter Notebook, kita dapat menyatukan seluruh proses tersebut dalam satu tempat. Kita dapat mengimpor data dari file Excel, memanipulasi dan menganalisis data tersebut menggunakan Pandas, serta mendokumentasikan seluruh proses dan hasilnya secara interaktif. Pendekatan ini tidak hanya meningkatkan efisiensi kerja, tetapi juga memudahkan reproduksi dan kolaborasi, karena seluruh alur kerja dapat disimpan dan dibagikan dalam satu notebook interaktif.

## **B. Rumusan Masalah**

1. Code Editor apa saja yang digunakan untuk mengolah data statistik pada Python?
2. Library apa saja yang digunakan untuk mengolah data statistik pada Python?
3. Bagaimana mengolah data statistik berbasis Excel dengan Python?

## **C. Tujuan**

Modul ini, diharapkan bagi pembaca dapat memahami berbagai macam rumusan masalah yang ada. Serta diharapkan bagi pembaca dapat melakukan praktikum sebagai bentuk latihan dari materi yang telah penulis sampaikan.

## **BAB 2**

### **CODE EDITOR**

#### **A. Pengertian Code Editor**

Code editor adalah sebuah program komputer yang digunakan untuk menulis, mengedit, dan mengelola kode-kode untuk pengembangan perangkat lunak. Code editor biasanya dirancang untuk memberikan lingkungan yang nyaman bagi para pengembang untuk bekerja, dengan fitur-fitur seperti penyorotan sintaksis, pemeriksaan kesalahan (error checking), tata letak kode yang rapi, dan integrasi dengan alat pengembangan lainnya seperti debugger dan kontrol versi. Code Editor yang akan kami gunakan untuk mengolah data statistik pada Python ada dua yaitu Visual Studio Code (VS Code) dan Jupyter Notebook.

#### **B. Mengenal Visual Studio Code (VS Code)**

Visual Studio Code (VS Code) adalah editor source code buatan Microsoft yang beroperasi pada komputer desktop dan kompatibel dengan sistem operasi Windows, macOS, dan Linux. Visual Studio Code merupakan editor kode sederhana dengan dukungan untuk operasi pengembangan seperti debugging, menjalankan tugas, dan kontrol versi. Cara menggunakan Visual Studio Code yaitu:

1. Download dan install VS Code untuk sistem operasi yang relevan dari website resmi, yaitu <https://code.visualstudio.com>.
2. Launch VS Code dan buka folder atau file terkait untuk memulai coding.
3. Tulis atau edit kode di area editor, misalnya dengan IntelliSense.
4. Tingkatkan fungsionalitas dengan meng-install extension yang sesuai.
5. Gunakan terminal terintegrasi untuk menjalankan dan men-debug kode.
6. Kelola version control dengan fitur Git bawaan.

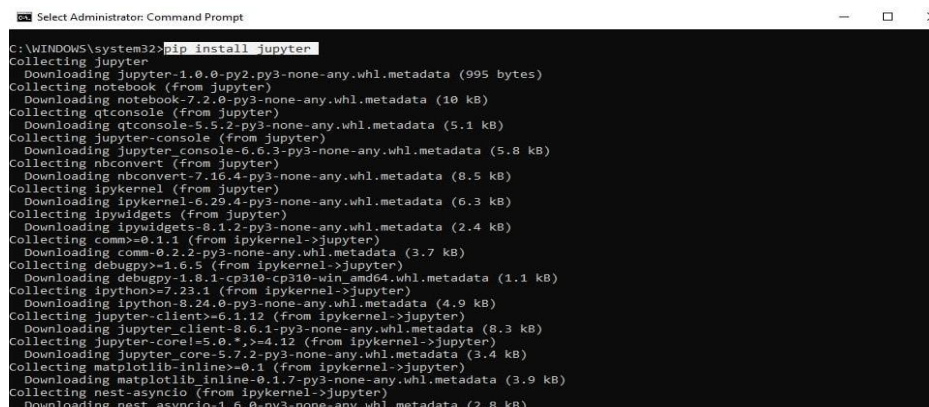
7. Personalisasi VS Code dengan tema dan setting lainnya.
8. Lakukan kolaborasi dengan orang lain menggunakan fitur Live Share jika perlu.
9. Simpan perubahan dan tutup VS Code setelah selesai.

### C. Mengenal Jupyter Notebook

Jupyter Notebook (file yang berekstensi .ipynb) adalah dokumen interaktif yang dihasilkan oleh aplikasi Jupyter Notebook, yang berisi kode komputer dan elemen rich text seperti paragraf, persamaan matematika, gambar, dan tautan. Jupyter Notebook adalah aplikasi pengembangan interaktif berbasis web yang berjalan di browser, membuatnya mudah diakses dari berbagai perangkat tanpa perlu instalasi perangkat lunak tambahan. Dokumen Jupyter Notebook terdiri dari blok-blok kode yang disebut sel, yang dapat dieksekusi secara independen. Setiap sel dapat berisi satu atau beberapa baris kode yang bisa dijalankan terpisah dari sel lainnya, memberikan fleksibilitas tinggi untuk eksperimen, debugging, dan iterasi pengembangan. Selain kode, pengguna dapat menambahkan teks naratif menggunakan Markdown, menulis persamaan matematis dengan LaTeX, serta menyisipkan gambar dan tautan, sehingga seluruh proses analisis data, visualisasi, dan dokumentasi dapat dilakukan dalam satu tempat. Dengan fitur-fitur ini, Jupyter Notebook menjadi alat yang sangat berguna untuk pengembangan, analisis data, pembelajaran mesin, pendidikan, dan penelitian ilmiah, memungkinkan integrasi kode, visualisasi, dan dokumentasi dalam satu dokumen interaktif.

### D. Langkah-langkah Penggunaan Jupyter Notebook dalam VS Code

1. Buka CMD dan Select as administrator, ketik “pip install jupyter”

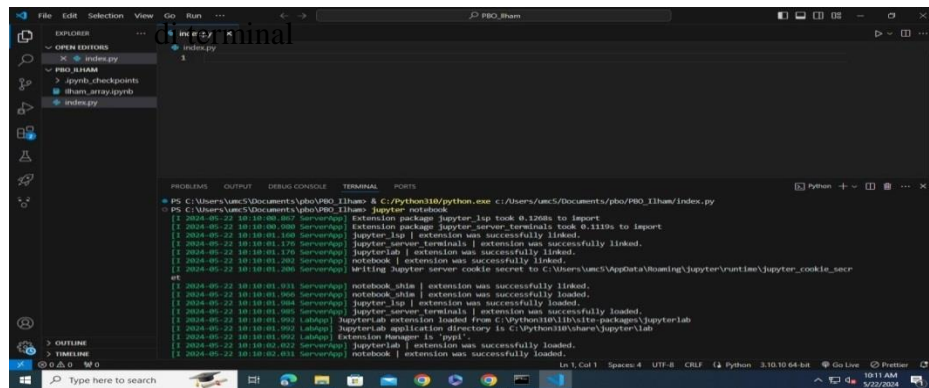


```

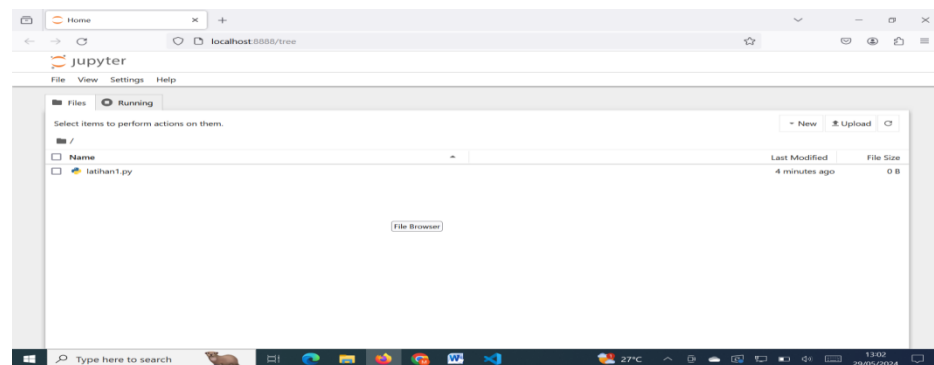
Select Administrator: Command Prompt
C:\WINDOWS\system32>pip install jupyter
Collecting jupyter
  Downloading jupyter-1.0.0-py2.py3-none-any.whl.metadata (995 bytes)
Collecting notebook (from jupyter)
  Downloading notebook-7.2.0-py3-none-any.whl.metadata (10 kB)
Collecting qtconsole (from jupyter)
  Downloading qtconsole-5.5.2-py3-none-any.whl.metadata (5.1 kB)
Collecting jupyter-console (from jupyter)
  Downloading jupyter_console-6.6.3-py3-none-any.whl.metadata (5.8 kB)
Collecting nbconvert (from jupyter)
  Downloading nbconvert-7.16.4-py3-none-any.whl.metadata (8.5 kB)
Collecting ipykernel (from jupyter)
  Downloading ipykernel-6.29.4-py3-none-any.whl.metadata (6.3 kB)
Collecting ipywidgets (from jupyter)
  Downloading ipywidgets-8.1.2-py3-none-any.whl.metadata (2.4 kB)
Collecting comm>=0.1.1 (from ipykernel->jupyter)
  Downloading comm-0.2.2-py3-none-any.whl.metadata (3.7 kB)
Collecting debugpy>=1.6.5 (from ipykernel->jupyter)
  Downloading debugpy-1.8.1-cp310-cp310-win_amd64.whl.metadata (1.1 kB)
Collecting ipython>=7.23.1 (from ipykernel->jupyter)
  Downloading ipython-8.24.0-py3-none-any.whl.metadata (4.9 kB)
Collecting jupyter-client>=6.1.12 (from ipykernel->jupyter)
  Downloading jupyter_client-8.6.1-py3-none-any.whl.metadata (8.3 kB)
Collecting jupyter-core>=5.0.*>=4.12 (from ipykernel->jupyter)
  Downloading jupyter_core-5.7.2-py3-none-any.whl.metadata (3.4 kB)
Collecting matplotlib-inline>=0.1 (from ipykernel->jupyter)
  Downloading matplotlib_inline-0.1.7-py3-none-any.whl.metadata (3.9 kB)
Collecting nest-asyncio (from ipykernel->jupyter)
  Downloading nest_asyncio-1.6.0-py3-none-any.whl.metadata (2.8 kB)

```

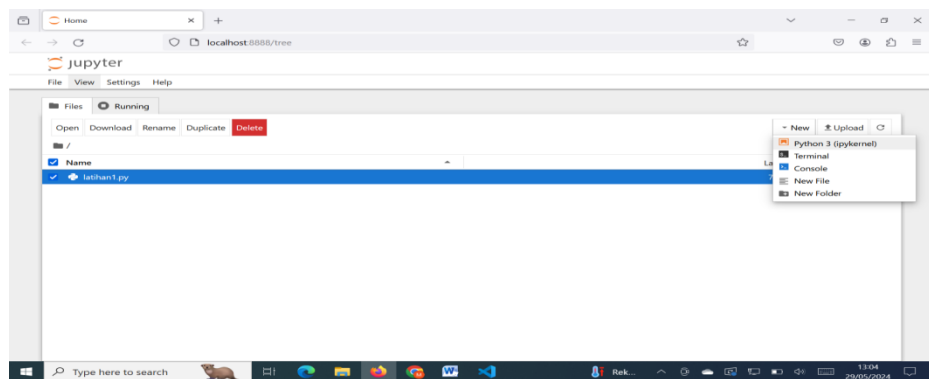
2. Buka terminal pada file python di vs code, lalu ketik "jupyter notebook"



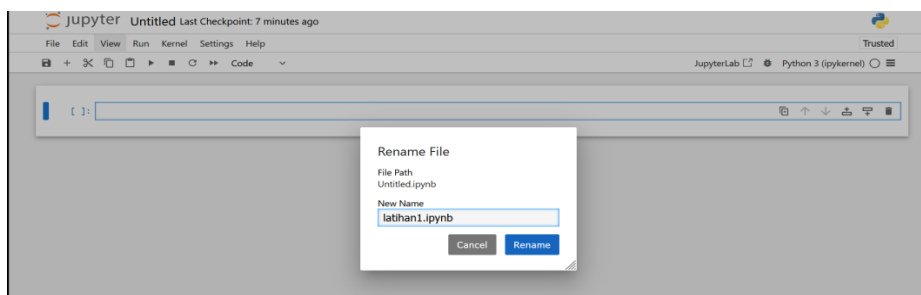
3. Lalu secara otomatis akan membuka web browser dan akan menampilkan seperti yang dibawah ini:



4. Klik New lalu pilih Python 3 (ipykernel)



5. Ubah nama File





## **BAB 3**

### **LIBRARY**

#### **A. Pengertian Library**

Library dalam konteks pemrograman adalah kumpulan kode atau fungsi yang telah ditulis sebelumnya dan dapat digunakan kembali dalam pengembangan perangkat lunak. Library berisi serangkaian fungsi, prosedur, dan/atau objek yang dapat digunakan oleh program utama untuk mencapai tujuan tertentu.

Secara umum, library menyediakan berbagai fungsionalitas yang dapat digunakan oleh pengembang untuk memperluas kemampuan program tanpa harus menulis ulang kode dari awal. Ini memungkinkan pengembang untuk fokus pada logika khusus aplikasi mereka sambil memanfaatkan solusi yang sudah ada dan terbukti.

Library Python adalah kumpulan modul dan paket yang disediakan untuk bahasa pemrograman Python. Python memiliki ekosistem library yang sangat kaya dan luas, yang mencakup berbagai bidang seperti pengembangan web, analisis data, kecerdasan buatan, pemrosesan bahasa alami, pemrograman jaringan, dan banyak lagi. Beberapa library Python yang populer dan sering digunakan untuk mengolah data statistik antara lain: Pandas, Numpy, Matplotlib dan seaborn, Tabulate, Openpyxl.

#### **B. Mengenal Pandas**

Pandas adalah library Python yang berfungsi untuk manipulasi dan analisis data, terutama data tabular seperti yang sering ditemui dalam spreadsheet atau relational database. Pandas merupakan singkatan dari Python Data Analysis Library. Pandas adalah sebuah open source python package/library dengan lisensi BSD yang menyediakan banyak perkakas untuk kebutuhan analisis data, manipulasi dan pembersihan data. Pandas mendukung pembacaan dan penulisan data dengan media berupa excel spreadsheet, CSV, dan SQL yang nantinya dijadikan sebagai objek python dengan rows dan columns.

Pandas dapat digunakan bersamaan dengan library lain dalam data science. Karena dibuat menggunakan Numpy, artinya ada banyak struktur library yang digunakan atau direplika di dalam Pandas. Selain itu, data yang diproduksi oleh Pandas sering kali digunakan sebagai input plotting functions untuk Matplotlib, analisis statistik di SciPy, serta algoritma machine learning dalam Scikit-learn. Program library Pandas dalam Python sendiri dapat dijalankan menggunakan berbagai code editor, namun sangat disarankan untuk menggunakan Jupyter Notebook dan VS Code.

Objek Pandas Series dan Data Frame:

a. Series

Series merupakan suatu objek satu dimensi yang dapat menyimpan berbagai jenis tipe data seperti integer, string, dan berbagai jenis tipe lainnya.

Contoh implementasi Series:

```
In [1]: import pandas as pd

        x = pd.Series([8,4,2,6])
        print(x)

Out[1]: 0    8
        1    4
        2    2
        3    6
        dtype: int64
```

b. Data Frame

Dataframe adalah suatu objek 2 dimensi yang mana nantinya akan digunakan sebagai tempat menyimpan data dengan lebih terstruktur. Dataframe memiliki 2 indeks yaitu indeks baris dan indeks kolom.

```
In [5]: import pandas as pd

        df = pd.DataFrame({'tipe_int': [135, 261], 'tipe_string': ['k', 'l']})
        print(df)

Out[5]:   tipe_int tipe_string
        0      135          k
        1      261          l
```

Pandas memiliki beberapa fitur utama:

1. Struktur data, Pandas menyediakan dua struktur data utama, yaitu Series dan DataFrame. (struktur data sudah dijelaskan sebelumnya)
2. Data cleaning, Pandas menyediakan fungsi untuk mengatasi masalah ini, seperti `dropna()` untuk menghapus data yang hilang, `fillna()` untuk mengisi data yang hilang, dan `drop_duplicates()` untuk menghapus data duplikat.

Fitur Utama Pandas bagian Data Cleaning yaitu:

Handling Missing Values:

```
# Menghitung missing values
print(df.isnull().sum())

# Mengisi missing values
df.fillna({'harga': df['harga'].mean(), 'jumlah': 0},
inplace=True)

# Menghapus baris dengan missing value
df.dropna(subset=['jumlah'], inplace=True)
```

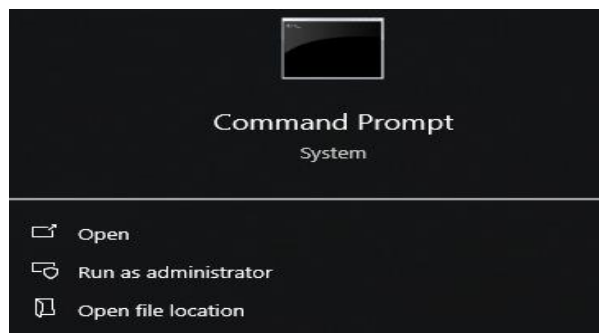
Removing Duplicate Values:

```
df.drop_duplicates(inplace=True)
```

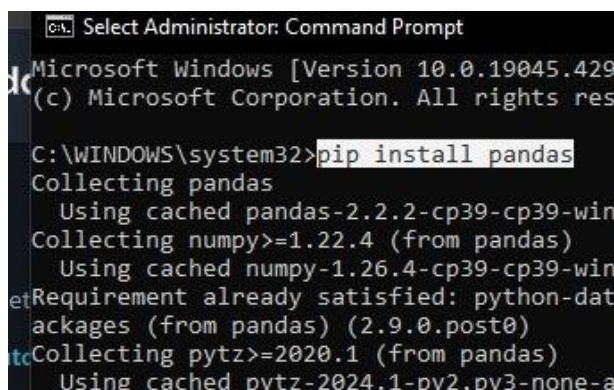
3. Transformasi data dengan Pandas, kita dapat dengan mudah mengubah bentuk data, menggabungkan beberapa set data, atau mengelompokkan data berdasarkan kriteria tertentu. Fungsi seperti `pivot()`, `melt()`, `groupby()`, dan `merge()` memudahkan proses transformasi data sesuai kebutuhan analisis.
4. Analisis data, kita bisa dengan cepat mendapatkan ringkasan statistik dari data dengan `describe()`, menghitung korelasi antar kolom dengan `corr()`, atau bahkan melakukan operasi yang lebih kompleks seperti agregasi dengan `agg()`.

5. Kemudahan dalam impor/ekspor data merupakan salah satu kelebihan Pandas, kemampuannya untuk bekerja dengan berbagai format data. Kamu bisa dengan mudah mengimpor data dari CSV, Excel, SQL, dan banyak format lainnya menggunakan fungsi seperti `read_csv()`. Begitu juga saat menyimpan data, Pandas menyediakan metode seperti `to_csv()` yang memudahkan menyimpan hasil analisis.

Untuk menjalankan library pandas, langkah pertama yang paling terpenting adalah menginstall packagenya menggunakan command prompt. buka command prompt dan pilih "Run as administrator."



Kemudian ketik "pip install pandas" di kolom command prompt. Lalu tunggu sampai proses download dan instalasi package library pandas berhasil (successfully).



### C. Mengenal Numpy

NumPy (Numerical Python) adalah library Python yang fokus pada scientific computing. NumPy memiliki kemampuan untuk membentuk objek N-dimensional array, yang mirip dengan list pada Python. Keunggulan NumPy array dibandingkan dengan list pada Python adalah konsumsi memory yang lebih kecil serta runtime yang lebih cepat.

#### Array pada NumPy

Lakukan import terlebih dahulu library numpy as np. Penggunaan as disini, artinya kita menggantikan pemanggilan numpy dengan prefix np untuk proses berikutnya.

```
In [3]: 1 import numpy as np

In [4]: 1 # membuat array
        2 a = np.array([1, 2, 3])
        3 a

Out[4]: array([1, 2, 3])
```

#### Type Data pada NumPy

Untuk melakukan pengecekan tipe pada array menggunakan fungsi type().

```
In [6]: 1 # mengecek tipe data array
        2 type(a)

Out[6]: numpy.ndarray
```

#### Tipe Data Element pada NumPy

Pengecekan tipe data element pada array menggunakan fungsi dtype

```
In [25]: 1 a = np.array([1, 2, 3])
        2 a.dtype

Out[25]: dtype('int32')
```

## Dimensi Array pada NumPy

NumPy array memiliki keunggulan mendukung operasi pada data dimensional seperti Vektor dan Matriks. Untuk mengetahui jumlah dimensi pada data menggunakan fungsi `ndim`.

```
In [34]: 1 # cek jumlah dimensi pada array
          2 a = np.array([1, 2, 3])
          3 a.ndim
```

```
Out[34]: 1
```

## Operasi Array pada NumPy

NumPy memudahkan kita untuk operasi elementwise pada Vektor dan Matriks seperti penjumlahan, perkalian, pangkat, dan operasi lainnya.

```
In [39]: 1 a = np.array([1, 2, 3])
          2 f = np.array([1.1, 2.2, 3.3])
          3 a + f
```

```
Out[39]: array([2.1, 4.2, 6.3])
```

```
In [40]: 1 a * f
```

```
Out[40]: array([1.1, 4.4, 9.9])
```

```
In [41]: 1 a ** f
```

```
Out[41]: array([ 1.          ,  4.59479342, 37.5405076 ])
```

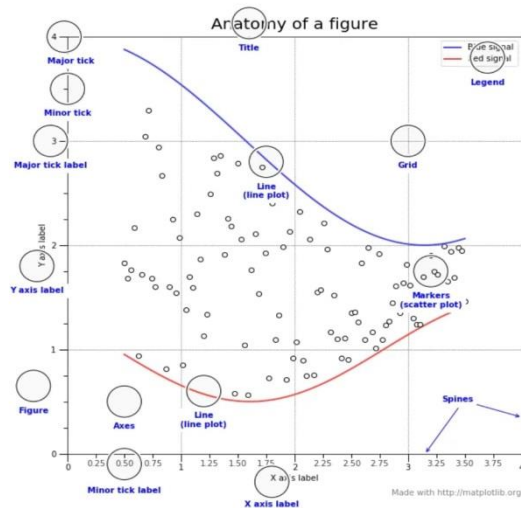
```
In [42]: 1 # universal function (ufuncs) pada numpy seperti sin
          2 np.sin(a)
```

```
Out[42]: array([0.84147098, 0.90929743, 0.14112001])
```

## D. Mengenal Matplotlib dan Seaborn

Matplotlib adalah library Python yang fokus pada visualisasi data seperti membuat plot grafik. Matplotlib pertama kali diciptakan oleh John D. Hunter dan sekarang telah dikelola oleh tim developer yang besar. Matplotlib dapat digunakan dalam skrip Python, Python dan IPython shell, server aplikasi web, dan beberapa toolkit graphical user interface (GUI) lainnya.

Gambar di bawah menunjukkan bagian-bagian dari visualisasi matplotlib dibuat oleh Nicolas P. Rougier.



## Menggunakan matplotlib

Untuk memulai menggunakan matplotlib, lakukan import terlebih dahulu library matplotlib.pyplot as plt. Penggunaan as disini, artinya kita menggantikan pemanggilan fungsi pyplot pada matplotlib dengan prefix plt untuk proses berikutnya. Disini terdapat magic command %matplotlib inline, untuk pengaturan pada backend matplotlib agar setiap grafik ditampilkan secara 'inline', yaitu akan ditampilkan langsung pada cell notebook.

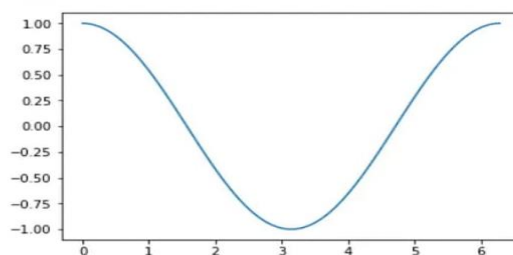
```
In [1]: 1 # import library
        2 import matplotlib.pyplot as plt
        3 %matplotlib inline
```

## Membuat Line Plot

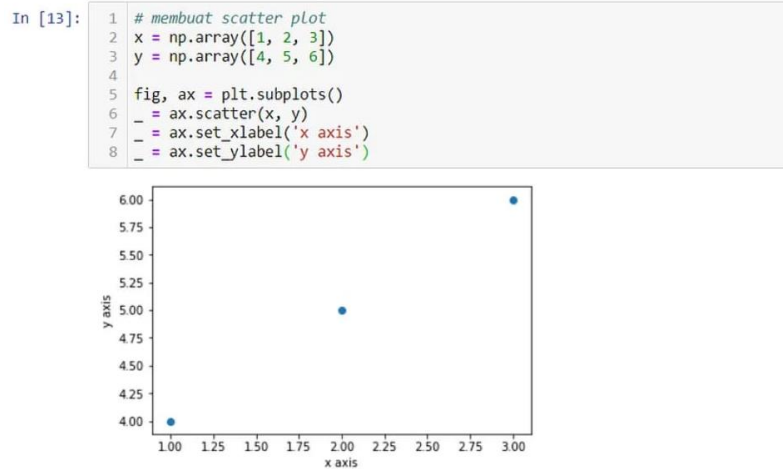
Tutorial kali ini akan membuat plot grafik line menggunakan gelombang cos. Kita akan menggunakan numpy untuk generate data gelombang cos dengan jumlah data 100 yang berjarak dari 0 sampai  $2\pi$ .

```
In [2]: 1 import numpy as np
        2
        3 x = np.linspace(0, 2*np.pi, 100)
        4 cos_x = np.cos(x)
```

```
In [3]: 1 # membuat line plot
        2 fig, ax = plt.subplots()
        3 _ = ax.plot(x, cos_x)
```



Matplotlib menyediakan fungsi `scatter()` untuk mempermudah dalam visualisasi scatter plot.



## Membuat Bar Plot

Matplotlib menyediakan fungsi `bar()` untuk mempermudah dalam visualisasi bar plot.



**Seaborn**, di sisi lain, adalah antarmuka tingkat tinggi untuk membuat grafik statistik. Itu dibangun di atas Matplotlib dan menyediakan antarmuka yang lebih sederhana dan intuitif untuk membuat plot statistik umum.



## Instalasi Seaborn

Setelah memastikan PIP berfungsi dengan baik, buka terminal atau command prompt dan ketik perintah berikut:

```
pip install seaborn
```

Untuk menggunakan Seaborn, kamu juga perlu mengimpor Matplotlib karena Seaborn dibangun di atasnya. Berikut cara mengimpornya:

```
import seaborn as sns
import matplotlib.pyplot as plt
```

Menggunakan dataset sampel dari Seaborn

Untuk memuat dataset sampel, kamu bisa menggunakan fungsi `load_dataset()`. Misalnya, untuk memuat dataset yang bernama 'tips', kamu bisa mengetik perintah

```
import seaborn as sns
data = sns.load_dataset('tips')
print(data.head())
```

Fungsi `head()` dari Pandas digunakan untuk menampilkan lima baris pertama dari dataset.

Menggunakan Data yang Sudah Ada

Setelah data dimuat ke dalam DataFrame Pandas, kamu akan lebih mudah memvisualisasikannya dengan Seaborn. Berikut contoh cara memuat data dari file CSV:

```
import pandas as pd
import seaborn as sns

data = pd.read_csv('lokasi_file.csv')
print(data.head())
```

## **E. Mengenal Tabulate**

Tabulate adalah perpustakaan sumber terbuka, artinya Anda dapat menggunakannya secara bebas tanpa khawatir akan plagiarisme. Atau, Tabulate adalah pembuatan tabel-tabel yang berisi data yang telah diberi kode sesuai dengan analisis yang dibutuhkan.

### **Cara Menginstall Tabulate:**

Ikuti langkah-langkah berikut untuk menginstallnya:

1. Buka Terminal atau Command Prompt:

Tergantung pada sistem operasi Anda (Windows, macOS, Linux), buka terminal atau command prompt.

2. Activate Virtual Environment (Optional but Recommended):

Merupakan praktik yang baik untuk bekerja dalam lingkungan virtual untuk menghindari konflik dengan paket Python lainnya . Jika Anda tidak menggunakan lingkungan virtual, Anda dapat melewati langkah ini. Buat lingkungan virtual (ganti myenv dengan nama lingkungan pilihan Anda): `python -m venv myenv`

Aktifkan lingkungan virtual:

- a. Di Windows : `myenv\Scripts\aktifkan`
- b. Di macOS dan Linux: `source myenv/bin/aktifkan`

3. Instal tabulasi menggunakan pip:

Setelah Anda berada di lingkungan virtual (jika Anda menggunakannya), Anda dapat menginstal pustaka tabulasi menggunakan perintah berikut: `"pip install tabulate"` Ini akan mengambil versi terbaru perpustakaan tabulasi dari Indeks Paket Python (PyPI) dan menginstallnya di sistem Anda.

#### 4. Verify Installation

Untuk memverifikasi bahwa instalasi berhasil, Anda dapat memulai sesi interaktif Python dengan mengetik `python` di terminal Anda. Kemudian, impor modul tabulasi dan periksa versinya: `import tabulate`  
`print(tabulate._version_)`

Jika Anda melihat nomor versi tanpa kesalahan apa pun, berarti perpustakaan telah diinstal dengan benar.

#### 5. Start Using 'tabulate'

Anda sekarang siap menggunakan perpustakaan tabulasi dalam proyek Python Anda. Impor ke skrip atau buku catatan Anda menggunakan:  
`from tabulate import tabulate.`

Anda telah berhasil menginstal pustaka tabulasi di sistem Anda dan sekarang siap membuat tabel berformat indah menggunakan Python.

### F. Mengenal Openpyxl

Openpyxl adalah perpustakaan Python untuk membaca/menulis file Excel 2010 xlsx/xlsm/xltx/xltn. Openpyxl merupakan perpustakaan Python yang memungkinkan pengguna membaca file Excel dan menulis ke dalamnya.

Untuk tutorial ini, Anda harus menggunakan Python 3.7 dan openpyxl 2.6.2. Untuk menginstal paket, Anda dapat melakukan hal berikut:



```
Kerang  
$ pip install openpyxl
```

Setelah Anda menginstal paketnya, Anda seharusnya dapat membuat spreadsheet super sederhana dengan kode berikut:

```
ular piton
from openpyxl import Workbook

workbook = Workbook()
sheet = workbook.active

sheet["A1"] = "hello"
sheet["B1"] = "world!"

workbook.save(filename="hello_world.xlsx")
```

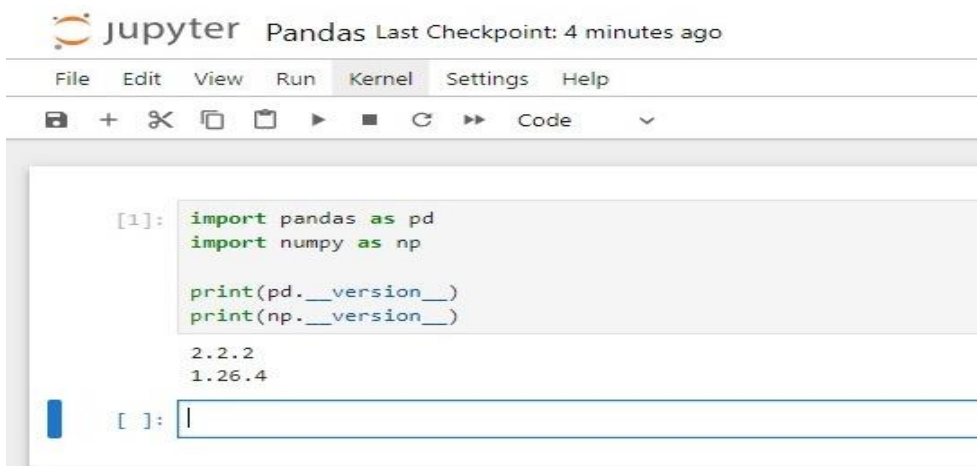
Ketika kita ingin membaca file spreadsheet excel agar terbaca di openpyxl, kita harus menyimpan file dengan contoh nama : "sample.xlsx", dan letakkan di lokasi yang mudah di akses agar mudah terbaca.

## BAB 4

### LATIHAN MENGOLAH DATA STATISTIK

### EXCEL DENGAN PYTHON

#### A. Implementasi Library Pandas



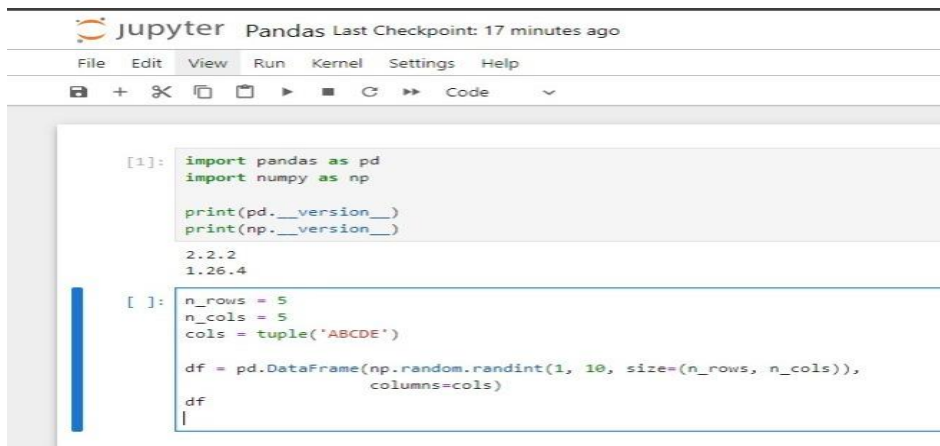
```
[1]: import pandas as pd
import numpy as np

print(pd.__version__)
print(np.__version__)

2.2.2
1.26.4

[ ]:
```

Setelah mengganti nama file di Jupyter, langkah berikutnya adalah mengimplementasikan library pandas. Pertama, pastikan pandas telah diunduh dan diinstal di perangkat Anda melalui command prompt. Setelah yakin pandas terinstal, impor library tersebut di Jupyter dengan mengetikkan `import pandas as pd`. Pastikan Anda menjalankan sel tersebut untuk memastikan tidak ada kesalahan. Anda bisa memeriksa versi pandas yang terinstal dengan menjalankan `pd.\_\_version\_\_` di Jupyter untuk memastikan bahwa instalasi berhasil.



```
[1]: import pandas as pd
import numpy as np

print(pd.__version__)
print(np.__version__)

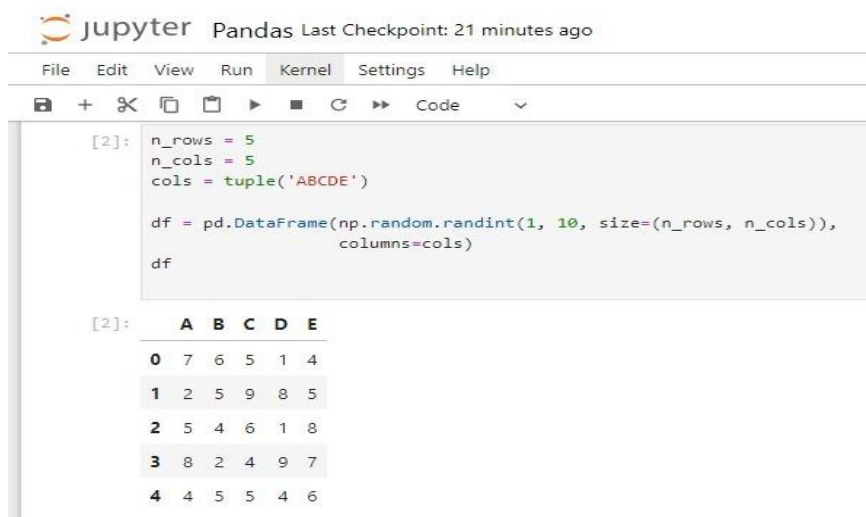
2.2.2
1.26.4

[ ]: n_rows = 5
n_cols = 5
cols = tuple('ABCDE')

df = pd.DataFrame(np.random.randint(1, 10, size=(n_rows, n_cols)),
                  columns=cols)
df
|
```

Untuk memulai pengenalan dengan library pandas, kita akan membuat data frame sederhana. Pertama, kita inialisasikan variabel `row` dan `col`. Kemudian, kita menggunakan pendekatan variabel `row` dan `col` untuk membuat baris dan kolom. Di sini, kita menggunakan tuple 'ABCDE' sebagai inisiasi nama kolom. Yang penting, kita perlu mendefinisikan variabel dan membuat fungsi DataFrame dengan panjang kolom 5 dan panjang baris 10. Berikut langkah-langkahnya:

1. Inialisasikan variabel `row` dan `col`.
2. Gunakan variabel `row` dan `col` untuk membuat baris dan kolom.
3. Gunakan tuple 'ABCDE' sebagai inisiasi nama kolom.
4. Buat DataFrame dengan panjang kolom 5 dan panjang baris 10.

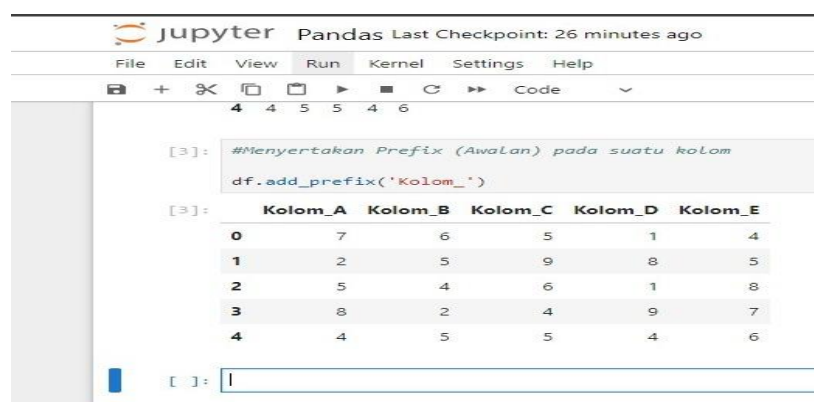


```
[2]: n_rows = 5
      n_cols = 5
      cols = tuple('ABCDE')

      df = pd.DataFrame(np.random.randint(1, 10, size=(n_rows, n_cols)),
                        columns=cols)
      df
```

	A	B	C	D	E
0	7	6	5	1	4
1	2	5	9	8	5
2	5	4	6	1	8
3	8	2	4	9	7
4	4	5	5	4	6

Setelah langkah-langkah sebelumnya, hasilnya akan seperti yang terlihat di gambar. Data frame telah dibuat dengan panjang kolom 5, diinisialisasi dengan A, B, C, D, E, dan panjang baris 10 dengan urutan indeks 0-9. Setiap kolom dan baris memiliki nilai pada setiap selnya. Nilai dari setiap sel dihasilkan dari perintah np.random.randint di kode sebelumnya.



```
[3]: #Menyertakan Prefix (Awalan) pada suatu kolom
      df.add_prefix('Kolom_')
```

```
[3]:
```

	Kolom_A	Kolom_B	Kolom_C	Kolom_D	Kolom_E
0	7	6	5	1	4
1	2	5	9	8	5
2	5	4	6	1	8
3	8	2	4	9	7
4	4	5	5	4	6

Selanjutnya, kita akan mencoba membuat prefix untuk suatu kolom dalam data frame. Prefix adalah awalan yang diberikan kepada nama kolom yang bertipe string. Prefix ini akan ditempatkan di awal inisialisasi kolom. Misalnya, jika kita awalnya membuat data frame dengan kolom bernama A, ketika kita menambahkan prefix, prefix tersebut akan ditempatkan di depan nama kolom yang sudah ada sebelumnya. Contoh pembuatan prefix pada suatu kolom dapat dilihat dalam gambar.

```
[4]: #Menyertakan Suffix (Akhiran) bertipe string pada suatu kolom
df.add_suffix('_Akhiran')
```

```
[4]:
```

	A_Akhiran	B_Akhiran	C_Akhiran	D_Akhiran	E_Akhiran
0	7	6	5	1	4
1	2	5	9	8	5
2	5	4	6	1	8
3	8	2	4	9	7
4	4	5	5	4	6

Selanjutnya, kita akan mencoba membuat suffix untuk suatu kolom dalam data frame. Suffix adalah akhiran yang diberikan kepada nama kolom yang bertipe string. Suffix ini akan ditempatkan di akhir inisialisasi kolom. Misalnya, jika kita awalnya membuat data frame dengan kolom bernama A, ketika kita menambahkan suffix, suffix tersebut akan ditempatkan di belakang nama kolom yang sudah ada sebelumnya. Contoh pembuatan suffix pada suatu kolom sangat sederhana dan hampir mirip dengan prefix.

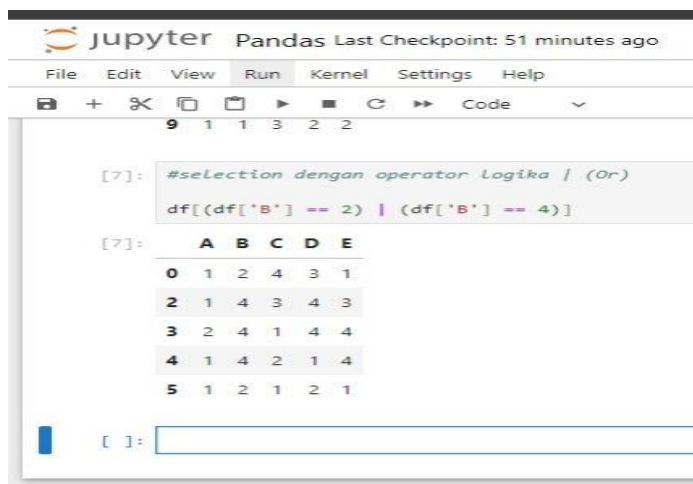
```
#Membuat DataFrame baru terlebih dahulu
n_rows = 10
n_cols = 5
cols = tuple('ABCDE')

df = pd.DataFrame(np.random.randint(1, 5, size=(n_rows, n_cols)),
                  columns=cols)
df
```

```
[5]:
```

	A	B	C	D	E
0	1	2	4	3	1
1	1	3	3	2	1
2	1	4	3	4	3
3	2	4	1	4	4
4	1	4	2	1	4
5	1	2	1	2	1
6	3	3	3	3	2
7	1	1	4	1	4
8	3	3	3	1	4
9	1	1	3	2	2

Kita akan mencoba melakukan seleksi pada library pandas, yang merupakan konsep terkait dengan logika matematika. Seleksi digunakan saat kita ingin memilih kolom atau baris dengan nilai sel tertentu yang sesuai dengan kebutuhan kita. Ini memungkinkan kita untuk memanipulasi dan menyortir data dengan cepat dan efisien. Untuk mencobanya, kita harus memiliki tabel data frame terlebih dahulu, yang telah dijelaskan sebelumnya. Hanya perlu menyesuaikan jumlah baris dan kolom sesuai kebutuhan.



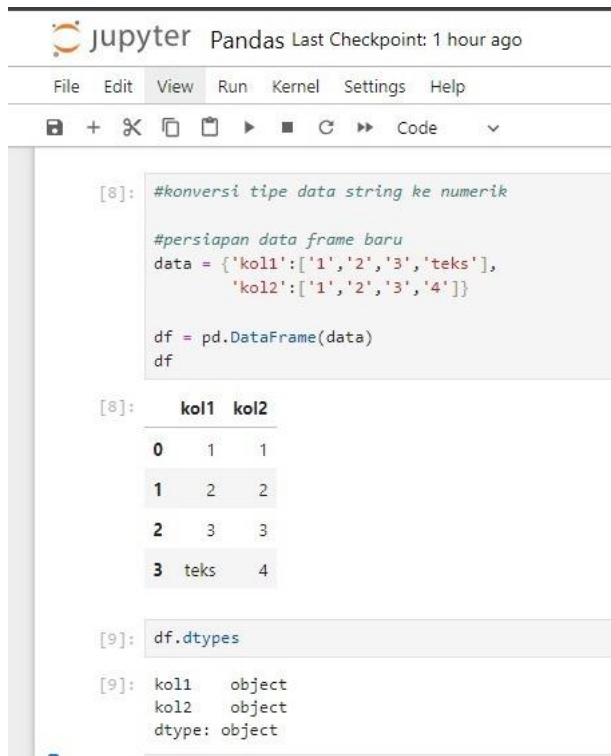
```
[7]: #selection dengan operator logika | (Or)
df[(df['B'] == 2) | (df['B'] == 4)]
```

```
[7]:
```

	A	B	C	D	E
0	1	2	4	3	1
2	1	4	3	4	3
3	2	4	1	4	4
4	1	4	2	1	4
5	1	2	1	2	1

Jika sudah memiliki tabel data frame, kita akan menggunakan syntax seperti pada contoh di gambar. kita menggunakan pendekatan logika or dengan lambang ( | ) di mana suatu data frame yang sudah kita definisikan sebagai df pada awal pembuatannya. di sini kita akan mencoba menyortir data frame df pada kolom B, di mana pada pada suatu cellNya memiliki value 2 (== 2) atau ( | ) value 4 (==4). jika sudah maka hasilnya bisa dilihat seperti pada di gambar, yang sebelumnya kita memiliki banyak value pada suatu baris, kini kita memiliki baris di mana kolom yang kita tentukan hanya memuat value 2 atau 4 saja. di sini kita memilih kolom B dengan nilai 2 atau 4.





The screenshot shows a Jupyter Notebook window with the title 'Pandas Last Checkpoint: 1 hour ago'. The interface includes a menu bar (File, Edit, View, Run, Kernel, Settings, Help) and a toolbar with icons for saving, adding, deleting, and running code. The notebook contains two code cells. The first cell, labeled [8], contains Python code to create a DataFrame with two columns, 'kol1' and 'kol2'. The second cell, labeled [9], contains the command 'df.dtypes' to check the data types of the columns. The output of the first cell is a table with four rows and two columns. The output of the second cell shows the data types for 'kol1', 'kol2', and the 'dtype' attribute.

```
[8]: #konversi tipe data string ke numerik

#persiapan data frame baru
data = {'kol1': ['1', '2', '3', 'teks'],
        'kol2': ['1', '2', '3', '4']}

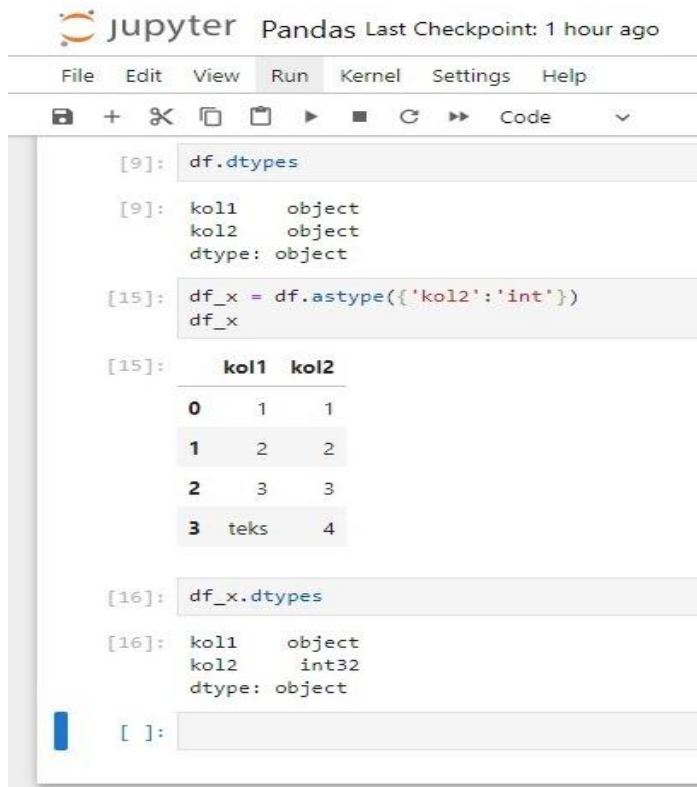
df = pd.DataFrame(data)
df
```

	kol1	kol2
0	1	1
1	2	2
2	3	3
3	teks	4

```
[9]: df.dtypes
```

```
[9]: kol1    object
     kol2    object
     dtype: object
```

Kemudian kita akan belajar untuk mengkonversi tipe data string dari suatu cell menjadi tipe data berupa numerik. mula mula kita harus menyiapkan satu data frame terlebih dahulu. bisa diikuti pada contoh di gambar. Pada gambar yang tertera, kita telah membuat data frame dengan inisialisasi kolom yaitu kol1 dan kol2. dan kita sudah mendefinisikan tiap tiap nilai dari kol1 dan kol2. jika sudah, maka hasil tabel data frame yang dibuat bisa kita lihat. kemudian kita cek tipe data dari masing masing kolom yang ada dengan mengetik `.dtypes` di sini kita membuat data frame dengan inisialisasi `df`, jadi syntaxnya `df.dtypes`, pastikan sudah sesuai dengan inisialisasi variabel data frame yang anda buat. maka akan tampil tipe data dari kol1 dan kol2. pada gambar tertulis `object`, yang artinya value cell dari kol1 dan kol2 memiliki type data string. di sini kita akan belajar mencoba tipe data string tersebut menjadi numerik.



The image shows a Jupyter Notebook interface with the title 'Pandas Last Checkpoint: 1 hour ago'. The interface includes a menu bar (File, Edit, View, Run, Kernel, Settings, Help) and a toolbar with icons for saving, adding, deleting, and running code. The notebook contains the following code cells:

```
[9]: df.dtypes
```

```
[9]: kol1    object
     kol2    object
     dtype: object
```

```
[15]: df_x = df.astype({'kol2':'int'})
     df_x
```

```
[15]:
```

	kol1	kol2
0	1	1
1	2	2
2	3	3
3	teks	4

```
[16]: df_x.dtypes
```

```
[16]: kol1    object
     kol2   int32
     dtype: object
```

```
[ ]:
```

Untuk mengkonversi tipe data string ke numerik kita bisa menggunakan pendekatan `astype`, bisa dilihat pada gambar. Pertama kita mendefinisikan suatu variabel terlebih dahulu supaya fungsi pendekatan `astype` tersebut bisa kita eksekusi. Bisa dilihat pada gambar, kita akan merubah tipe data pada data frame dengan inisialisasi yang kita berikan, yakni `df`. Jadi kita dapat mengetikkan `df.astype` lalu kita bisa mengisi parameter dari `astype` tersebut dengan nama kolom yang akan kita konversi tipe datanya, dan tipe data numerik apa yang akan kita berikan. di sini kita mencoba mengkonversi `kol2` yang sebelumnya string, menjadi `int`. pada parameter `astype`. jika sudah bisa kita eksekusi kode tersebut. secara tampilan data frame tidak ada yang berubah, tapi ketika anda periksa tipe datanya menggunakan `dtypes`, bisa dilihat tipe data pada kol 2 yang awalnya `object` (string) sekarang sudah dikonversi menjadi suatu tipe data integer (`int`). untuk konversi tipe numerik ini tidak hanya berlaku untuk konversi ke integer, bisa dikonversi ke tipe data float juga.

```
jupyter Pandas Last Checkpoint: 11 hours ago
File Edit View Run Kernel Settings Help

#membagi data frame menjadi 2 secara acak
#buat data frame baru
n_rows = 10
n_cols = 5
cols = tuple('ABCDE')

df = pd.DataFrame(np.random.randint(1, 20, size=(n_rows, n_cols)),
                  columns=cols)
df
```

	A	B	C	D	E
0	2	12	11	1	8
1	16	4	15	4	13
2	9	16	1	16	19
3	14	12	8	7	17
4	15	8	9	11	13
5	5	16	2	19	15
6	3	9	7	3	7
7	3	1	2	4	12
8	9	4	1	16	5
9	16	18	4	18	5

Berikutnya kita akan belajar cara membagi data frame menjadi 2 secara acak. pertama kita siapkan terlebih dahulu tabel data frame, untuk rentang kolom dan barisnya sesuaikan dengan kebutuhan kita. data yang telah kita siapkan ini nantinya akan kita bagi menjadi 2 tabel data frame secara acak.

```
jupyter Pandas Last Checkpoint: 1 hour ago
File Edit View Run Kernel Settings Help

[17]:
```

	A	B	C	D	E
0	2	12	11	1	8
1	16	4	15	4	13
2	9	16	1	16	19
3	14	12	8	7	17
4	15	8	9	11	13
5	5	16	2	19	15
6	3	9	7	3	7
7	3	1	2	4	12
8	9	4	1	16	5
9	16	18	4	18	5

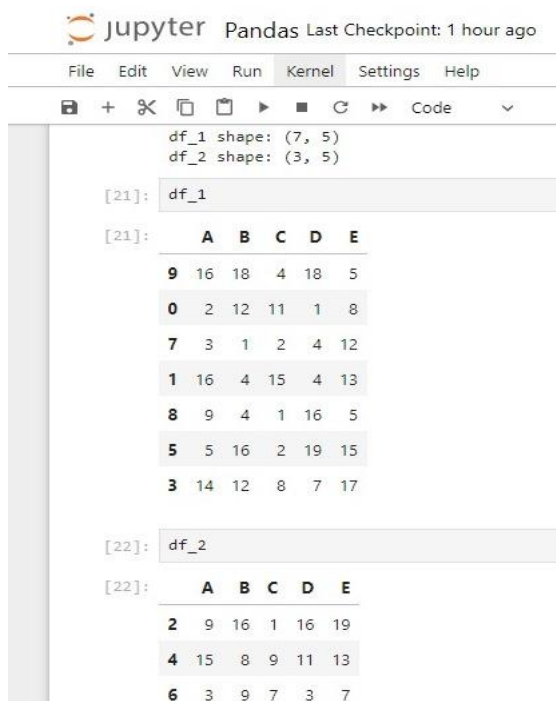
```
[20]: proporsi = 0.7
df_1 = df.sample(frac=proporsi)
df_2 = df.drop(df_1.index)

print(f'df_1 shape: {df_1.shape}')
print(f'df_2 shape: {df_2.shape}')

df_1 shape: (7, 5)
df_2 shape: (3, 5)
```

Pertama kita cek terlebih dahulu data shape pada data frame yang akan kita bagi, pada data frame sebelumnya yang kita buat yaitu (10, 5) artinya jumlah baris kita

adalah 10 dan kolomnya 5. lalu data frame df ini akan kita bagi menjadi 2 dengan proporsi 70% untuk data frame pertama, dan 30% untuk data frame kedua. caranya kita tetapkan proporsinya lalu kita masukkan ke dalam value variabel proporsi sebesar 0.7 lalu berikutnya kita panggil df.sample lalu menyertakan parameter frac (fraction=pecahan) dengan nilai proporsi, yakni 0.7 atau 70%. df.sample ini merupakan metode yang akan mengambil baris secara acak sejumlah 70% dari baris data yang dimiliki data frame df\_1 dan hasilnya akan dibentuk menjadi sebuah data frame baru yang akan kita tampung pada df\_1. Kemudian untuk mengisi df\_2 kita panggil dulu dfNya dan kita akan hapus atau df.drop berdasarkan df\_1.index, atau berdasarkan dataframe dataframe yang sudah diadopsi di df\_1. Lalu berikutnya kita print out shape dari kedua data framenya. kemudian eksekusi kodenya bisa kita lihat pada hasilnya, untuk df\_1 shapnya adalah (7,5) artinya jumlah barisnya ada 7 dan jumlah kolomnya tetap 5. Sedangkan untuk df\_2, shapnya adalah (3,5) artinya jumlah barisnya 3, dan jumlah kolomnya ada 5. dan hasil shape ini sudah sesuai dengan proporsi yang kita tetapkan yakni 0.7 / 70%. yaitu 70% untuk df\_1 dan 30% untuk df\_2.



```
df_1 shape: (7, 5)
df_2 shape: (3, 5)

[21]: df_1
[21]:
```

	A	B	C	D	E
9	16	18	4	18	5
0	2	12	11	1	8
7	3	1	2	4	12
1	16	4	15	4	13
8	9	4	1	16	5
5	5	16	2	19	15
3	14	12	8	7	17

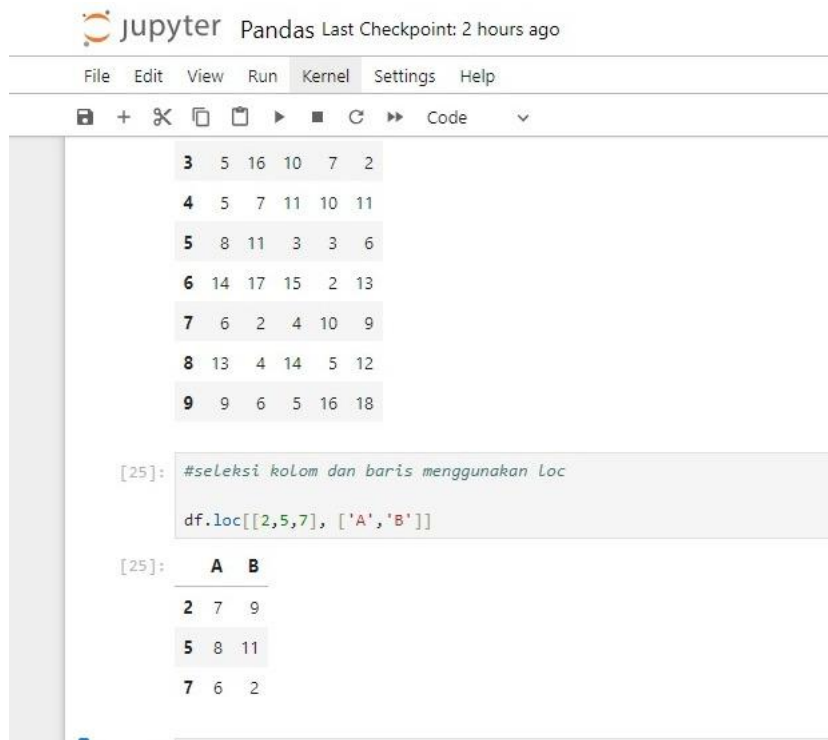
```

[22]: df_2
[22]:
```

	A	B	C	D	E
2	9	16	1	16	19
4	15	8	9	11	13
6	3	9	7	3	7

Lalu kita bisa cek data frame df\_1/df\_2 dengan cara seperti di gambar. Bisa kita lihat di sini bagaimana df\_1 maupun df\_2 terbentuk, kedua df ini terbentuk dari df sebelumnya. Yang perlu kita perhatikan adalah pada tiap tiap baris pada kedua df

ini sifatnya acak. meski diadopsi dari df sebelumnya, tapi proses adopsinya yaitu acak, bukan sistematis. pada proses semacam ini seringkali dibutuhkan pada beberapa teknik dalam bidang data science maupun machine learning.



The screenshot shows a Jupyter Notebook interface with the title "Pandas Last Checkpoint: 2 hours ago". The interface includes a menu bar (File, Edit, View, Run, Kernel, Settings, Help) and a toolbar with icons for file operations and execution. Below the toolbar, a DataFrame is displayed with 9 rows and 6 columns. The rows are indexed 3 through 9. The columns contain numerical values. Below the DataFrame, a code cell is shown with the following code:

```
[25]: #seleksi kolom dan baris menggunakan loc
df.loc[[2,5,7], ['A','B']]
```

The output of the code cell is displayed below the code, showing a new DataFrame with 3 rows and 2 columns. The rows are indexed 2, 5, and 7. The columns are labeled A and B. The values are 7, 8, and 6 for column A, and 9, 11, and 2 for column B.

	A	B
2	7	9
5	8	11
7	6	2

Berikutnya kita akan belajar menyeleksi baris dan kolom suatu data frame, pertama kita siapkan tabel data frame dengan rentang sesuai kebutuhan, kemudian untuk menyeleksi beberapa baris dan kolom kita bisa menggunakan pendekatan "loc".

Pertama kita panggil terlebih dahulu data frame kita, sebelumnya kita menginisialisasikan dataframe kita sebagai df, maka bisa kita ketikkan df.loc lalu diisi dengan parameter yang memuat baris, kemudian kolomnya. Pada gambar, kita akan menyeleksi baris 2,7,9 dan kolom A dan B. Maka hasilnya akan tampil data frame baru yang kita seleksi tersebut.

The screenshot shows a JupyterLab notebook with the following code and output:

```
[26]: #menggabungkan dua data frame

#buat dataframe terlebih dahulu
n_rows = 5
n_cols = 5
cols = tuple('ABCDE')

df = pd.DataFrame(np.random.randint(1, 20, size=(n_rows, n_cols)),
                  columns=cols)
df
```

	A	B	C	D	E
0	18	7	9	18	15
1	15	18	10	8	4
2	8	16	16	3	18
3	2	12	14	18	16
4	15	5	4	2	3

```
*[27]: #kita coba pecah terlebih dahulu dataframe utama

df1 = df.copy(deep = True)
df1 = df1.drop([1,4])
df1
```

Sebelumnya kita telah belajar cara membagi suatu tabel data frame menjadi 2, kali ini kita akan belajar menggabungkan 2 data frame menjadi satu dengan pendekatan inner dan outer. pertama kita siapkan terlebih dahulu satu data frame, kemudian untuk melakukan percobaan penggabungan data frame. kita harus memecah data frame tersebut menjadi 2 terlebih dahulu. untuk cara memecah data frame bisa dilihat pada contoh gambar, kita akan memecah menjadi df1 dan df2.

The screenshot shows a JupyterLab notebook with the following code and output:

```
*[27]: #kita coba pecah terlebih dahulu dataframe utama

df1 = df.copy(deep = True)
df1 = df1.drop([1,4])
df1
```

	A	B	C	D	E
0	18	7	9	18	15
2	8	16	16	3	18
3	2	12	14	18	16

```
[28]: df2 = df.copy(deep = True)
df2 = df2.drop([0,3])
df2
```

	A	B	C	D	E
1	15	18	10	8	4
2	8	16	16	3	18
4	15	5	4	2	3

Pada df drop di df1 dan df2, artinya kita akan memecah df1/df2 tanpa menyisipkan baris maupun kolom yang nilainya dimasukan pada parameter df.drop . bisa di lihat pada gambar.

Jupyter Pandas Last Checkpoint: 3 hours ago

File Edit View Run Kernel Settings Help

Code

```
[29]: # mulai menggabungkan dua data frame menggunakan pendekatan inner

df_inner = pd.merge(df1, df2, how='inner')
df_inner
```

	A	B	C	D	E
0	8	16	16	3	18

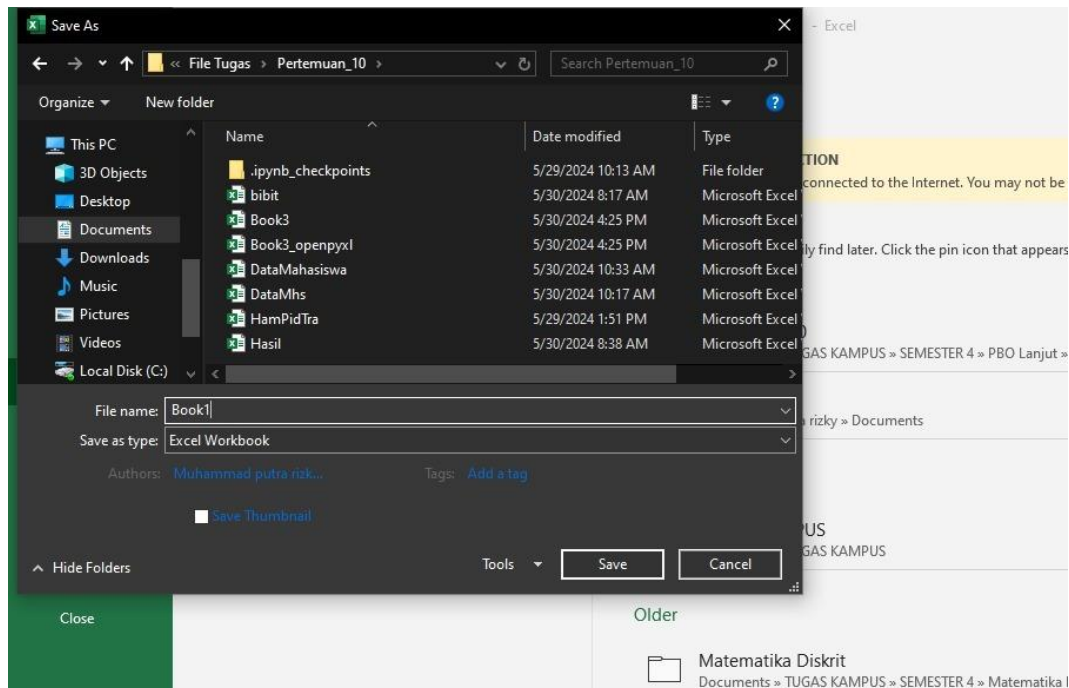
```
[30]: # mulai menggabungkan dua data frame menggunakan pendekatan inner

df_outer = pd.merge(df1, df2, how='outer')
df_outer
```

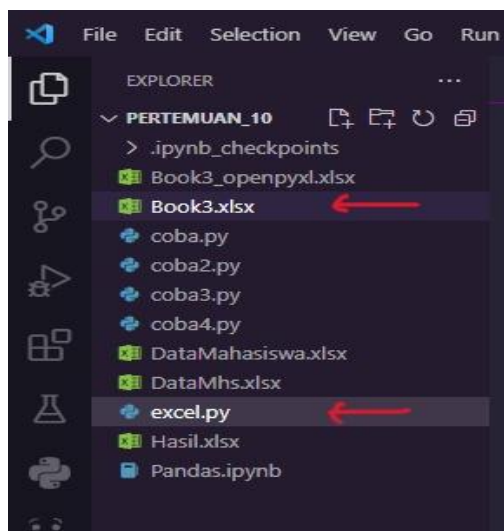
	A	B	C	D	E
0	2	12	14	18	16
1	8	16	16	3	18
2	15	5	4	2	3
3	15	18	10	8	4
4	18	7	9	18	15

Terakhir kita akan mencoba menggabungkan kedua data frame yang sebelumnya telah kita bagi menjadi 2 menggunakan pendekatan `pd.merge` yang memiliki parameter `how` dan bernilai `inner` maupun `outer`. Sebelum parameter `how`, kita akan menuliskan terlebih dahulu dataframe mana yang akan kita gabungkan.

Sebelum menjalankan fungsi `pd.merge`, kita definisikan suatu variabel yang akan memuat pendekatan merge tersebut terlebih dahulu. Untuk parameter `how` bernilai `"inner"` akan melakukan penggabungan antara data frame 1 dan data frame pada suatu baris dan kolomnya yang memiliki nilai cell yang sama. bisa dilihat pada gambar, berbeda dengan parameter `how` bernilai `"outer"`, dia akan melakukan penggabungan secara keseluruhan dari `df1` dan `df2`, tapi jika ada 2 baris kolom yang memiliki nilai cell serupa, akan di eliminasi menjadi 1 elemen baris kode saja. jadi pendekatan ini memungkinkan pencegahan pada baris kolom yang memiliki nilai yang sama antar cellnya.



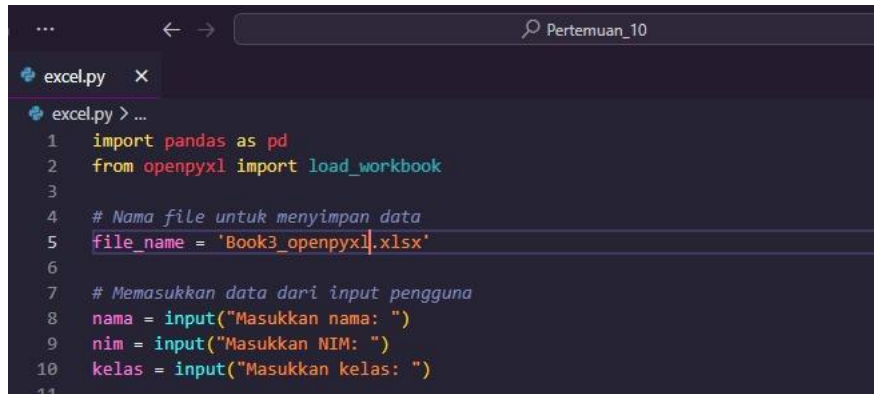
Pada tahap berikutnya kita akan belajar cara memanipulasi data excel menggunakan python x pandas, untuk code editor kita bisa menggunakan visual studio code maupun jupyter notebook. langkah pertama ialah kita buat dahulu file excel kosong yang tidak ada data di dalamnya, karna kita akan manipulasi data excel menggunakan python di vs code.



Pastikan file excel yang dibuat berada dalam satu direktori atau satu folder yang sama dengan file python kita, agar kita tidak bingung mencari /path file excel ketika hendak memanggilnya di vs code.

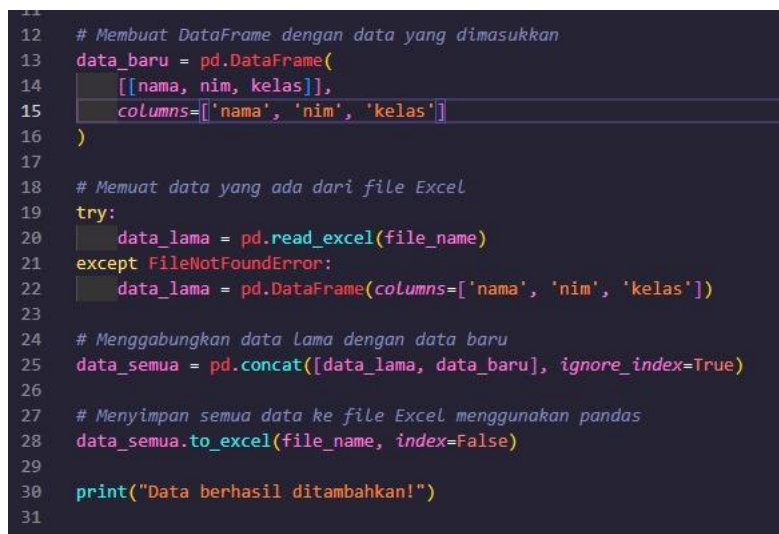


## B. Implementasi Openpyxl dengan Pandas



```
... ← → 🔍 Pertemuan_10
excel.py ×
excel.py > ...
1  import pandas as pd
2  from openpyxl import load_workbook
3
4  # Nama file untuk menyimpan data
5  file_name = 'Book3_openpyxl.xlsx'
6
7  # Memasukkan data dari input pengguna
8  nama = input("Masukkan nama: ")
9  nim = input("Masukkan NIM: ")
10 kelas = input("Masukkan kelas: ")
11
```

Lalu selanjutnya kita buka file python, kemudian seperti pada penjelasan di awal modul ini, kita harus mengimport library pandas terlebih dahulu dan openpyxl agar excel bisa memberikan hak akses untuk memanipulasi data di python. Selanjutnya kita akan membuat variabel untuk menyimpan nilai dari file excel kosong yang kita buat. bisa dilihat pada gambar, kita kali ini akan mencoba menambahkan data ke excel menggunakan input terminal python di vs code.



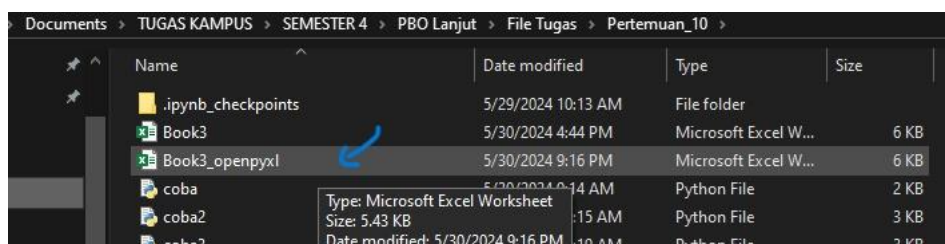
```
11
12 # Membuat DataFrame dengan data yang dimasukkan
13 data_baru = pd.DataFrame(
14     [[nama, nim, kelas]],
15     columns=['nama', 'nim', 'kelas']
16 )
17
18 # Memuat data yang ada dari file Excel
19 try:
20     data_lama = pd.read_excel(file_name)
21 except FileNotFoundError:
22     data_lama = pd.DataFrame(columns=['nama', 'nim', 'kelas'])
23
24 # Menggabungkan data lama dengan data baru
25 data_semua = pd.concat([data_lama, data_baru], ignore_index=True)
26
27 # Menyimpan semua data ke file Excel menggunakan pandas
28 data_semua.to_excel(file_name, index=False)
29
30 print("Data berhasil ditambahkan!")
31
```

Berikutnya kita akan membuat sebuah variable yang akan memuat data frame, karena kita akan membuat data frame excel di python. untuk kolom kolomnya bisa dilihat pada gambar. lalu kita akan memanggil fungsi concat untuk digunakan dalam konteks program Anda untuk menambah data baru tanpa menimpa data lama. Lalu kita buat variable yang akan memuat perintah to.excel, perintah ini berfungsi untuk menyimpan data frame yang kita buat sebelumnya di file excel, bisa di lihat pada gambar.

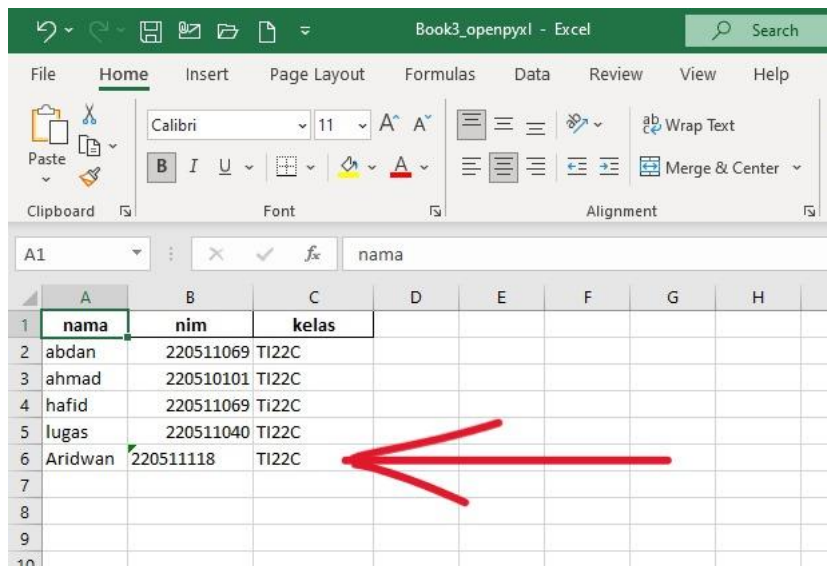
```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Tugas\Pertemuan_10> & C:
exe "c:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Tugas\Pertemuan_10\exc
Masukkan nama: Aridwan
Masukkan NIM: 220511118
Masukkan kelas: TI22C
Data berhasil ditambahkan!
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Tugas\Pertemuan_10>
```

Kemudian jika sudah mempraktikan seperti pada langkah langkah di gambar, anda bisa langsung menjalankan program tersebut. lalu masukkan data sesuai dengan inputannya.

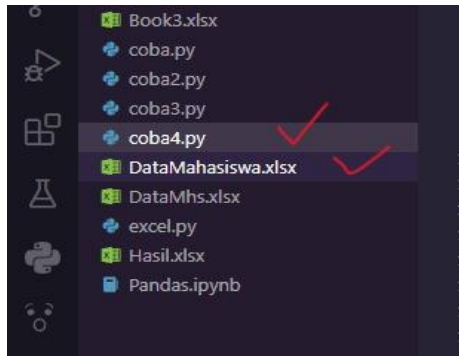


Jika data berhasil tersimpan, anda bisa membuka file excel yang sebelumnya sudah dibuat dan sudah dipindahkan menjadi satu direktori atau satu folder dengan file python kita.



Kemudian anda bisa melihat, kita telah berhasil membuat kolom header dan memasukkan data baru di excel menggunakan python.

### C. Fungsi yang Digunakan untuk Menambahkan, Menghapus, dan Mengupdate Data Statistik Excel

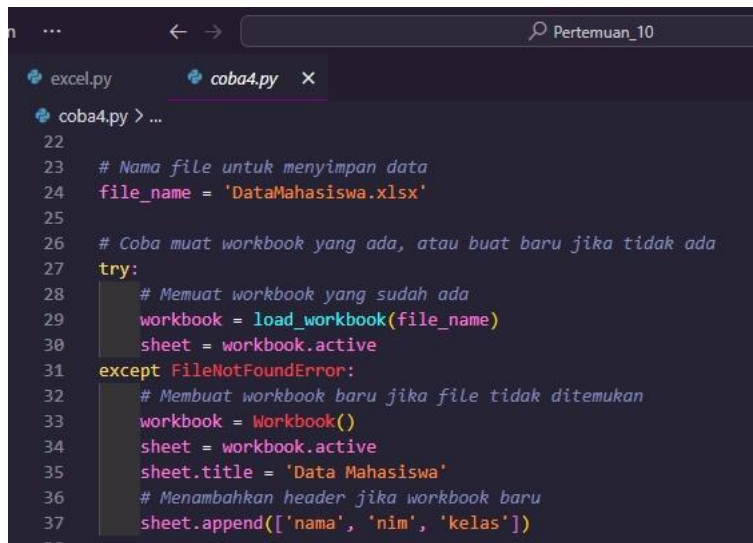


Sebelumnya kita telah belajar Menambah data pada file excel menggunakan python di Vs Code, kali ini kita akan mencoba belajar cara bagaimana agar kita bisa menghapus dan mengedit data pada file excel yang sudah ada. Pertama seperti biasa kita buat 1 file excel kosong dan 1 file python. pastikan kedua file ini berada pada folder atau direktori yang sama. contoh bisa dilihat pada gambar.

```
...      ← →      🔍 Pertemuan_10
coba4.py  excel.py  ×
coba4.py > ...
1  import pandas as pd
2  from openpyxl import load_workbook, Workbook
3
4  def dropna_from_dataframe(dataframe):
5      # Menghapus baris dari DataFrame berdasarkan input pengguna.
6      kolom = input("Masukkan nama kolom untuk menghapus data: ")
7      nilai = input(f"Masukkan nilai dalam kolom '{kolom}' yang ingin dihapus: ")
8
9      # Hapus baris yang sesuai dengan nilai dalam kolom yang ditentukan
10     dataframe = dataframe[dataframe[kolom] != nilai]
11     return dataframe
12
13 def edit_data(dataframe):
14     print("Data yang tersedia:")
15     print(dataframe)
16     idx = int(input("Masukkan indeks data yang ingin diedit: "))
17     kolom = input("Masukkan nama kolom yang ingin diubah (nama/nim/kelas): ")
18     nilai_baru = input("Masukkan nilai baru: ")
19
20     dataframe.at[idx, kolom] = nilai_baru
21     return dataframe
22
```

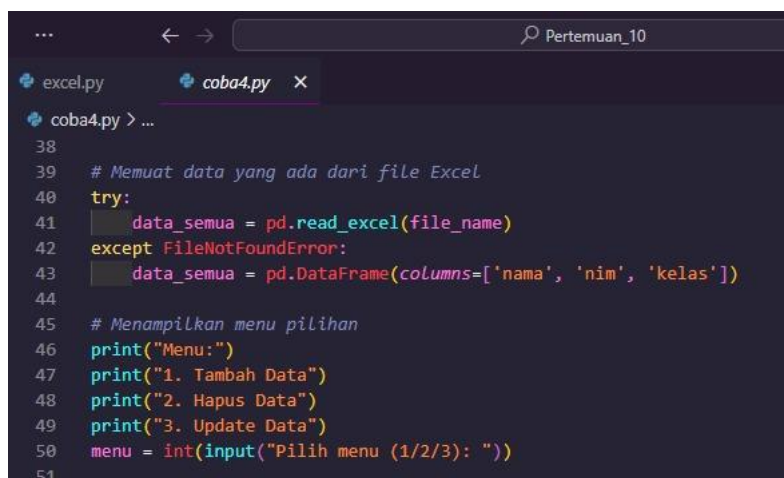
Pertama tama kita import terlebih dahulu library pandas dan openpyxl, karna di sini kita akan mengimplementasikan manipulasi data excel menggunakan python. untuk lebih jelas, perhatikan langkah pada gambar. Kita akan membuat fungsi dropna dengan parameter dataframe, fungsi dropna bertujuan untuk menghapus baris dari DataFrame berdasarkan nilai dalam kolom yang dimasukkan oleh pengguna. Lalu kita akan membuat variabel input kolom dan nilai, agar pengguna

dapat menginput mana kolom dan nilai yang akan dihapus. Kemudian buat variable dataframe untuk memilih dan menghapus baris yang kolomnya mengandung nilai tertentu. Selanjutnya kita akan membuat fungsi edit\_data, fungsi ini bertujuan untuk mengedit data yang sudah ada dalam DataFrame. Lalu kita buat variabel input yang nantinya akan digunakan oleh user ketika ingin menentukan mana data indeks, kolom, dan nilai baru yang akan diganti.



```
... ← → 🔍 Pertemuan_10
coba4.py ×
coba4.py > ...
22
23 # Nama file untuk menyimpan data
24 file_name = 'DataMahasiswa.xlsx'
25
26 # Coba muat workbook yang ada, atau buat baru jika tidak ada
27 try:
28     # Memuat workbook yang sudah ada
29     workbook = load_workbook(file_name)
30     sheet = workbook.active
31 except FileNotFoundError:
32     # Membuat workbook baru jika file tidak ditemukan
33     workbook = Workbook()
34     sheet = workbook.active
35     sheet.title = 'Data Mahasiswa'
36     # Menambahkan header jika workbook baru
37     sheet.append(['nama', 'nim', 'kelas'])
38
```

Lalu kita akan inisialisasikan variabel dengan nilainya yaitu nama file excel kita. pastikan file excel kosong yang kita buat di awal berada di satu direktori yang sama dengan file python. dalam contoh di gambar, kita membuat exception handling, di mana kita akan try memuat file workbook yang ada dengan load\_workbook, dan jika workbook tidak ada, maka workbook baru pada excel akan dibuat dan diberi judul serta header untuk kolom data.



```
... ← → 🔍 Pertemuan_10
coba4.py ×
coba4.py > ...
38
39 # Memuat data yang ada dari file Excel
40 try:
41     data_semua = pd.read_excel(file_name)
42 except FileNotFoundError:
43     data_semua = pd.DataFrame(columns=['nama', 'nim', 'kelas'])
44
45 # Menampilkan menu pilihan
46 print("Menu:")
47 print("1. Tambah Data")
48 print("2. Hapus Data")
49 print("3. Update Data")
50 menu = int(input("Pilih menu (1/2/3): "))
51
```

Lalu kita mencoba membuat try blok, agar memuat data dari excel ke dalam DataFrame "data\_semua", jika file tidak ditemukan pada excel, DataFrame kosong dengan kolom 'nama', 'nim', dan 'kelas' dibuat. itu sebabnya kita membuat file excel kosong di awal. Kemudian kita akan membuat menu pilihan yang akan ditampilkan kepada pengguna. variabel menu mengambil input pilihan dari pengguna untuk menentukan aksi yang akan dilakukan. apakah si pengguna ingin melakukan aksi tambah data, hapus data, maupun edit data excel.

```
...
← →
Pertemuan_10

excel.py  coba4.py x

coba4.py > ...
51
52 # Melakukan aksi sesuai pilihan menu
53 if menu == 1:
54     # Memasukkan data baru dari input pengguna
55     nama = input("Masukkan nama: ")
56     nim = input("Masukkan NIM: ")
57     kelas = input("Masukkan kelas: ")
58
59     # Menambahkan data baru ke DataFrame
60     data_baru = pd.DataFrame(
61         [[nama, nim, kelas]],
62         columns=['nama', 'nim', 'kelas']
63     )
64
65     # Menggabungkan data lama dengan data baru
66     data_semua = pd.concat([data_semua, data_baru], ignore_index=True)
67
```

Kemudian kita membuat kondisi, jika pengguna memilih menu 1, maka akan melakukan aksi tambah data. proses ini melibatkan pengambilan data dari input pengguna pada 'nama','nim','kelas'. kemudian membuat DataFrame di data\_baru, dengan kata yang diinputkan oleh pengguna. lalu menggunakan fungsi concat untuk menggabungkan data\_baru dan data\_semua. pendekatan concat ini juga memungkinkan ketika user menambah data baru, data baru akan terUpdate dan tidak akan menimpa atau menghilangkan data lama.

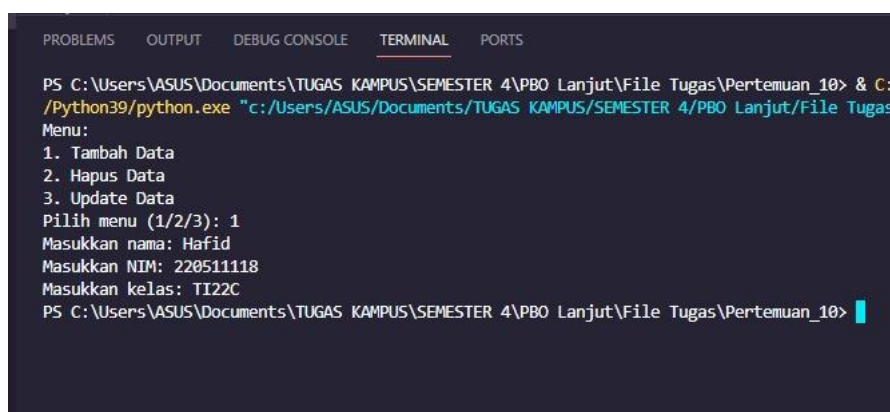
```
67
68 elif menu == 2:
69     # Memanggil fungsi dropna untuk menghapus data yang diinginkan
70     data_semua = dropna_from_dataframe(data_semua)
71
72 elif menu == 3:
73     # Memanggil fungsi edit_data untuk mengedit data yang sudah ada
74     data_semua = edit_data(data_semua)
75
76 # Menyimpan semua data ke file Excel menggunakan pandas
77 data_semua.to_excel(file_name, index=False)
78
```

Lalu kita membuat kondisi kedua, jika pengguna memilih opsi 2 (Hapus Data),



maka akan memanggil dan mengeksekusi fungsi `dropna_from_dataframe` yang sebelumnya dibuat, yang merupakan fungsi untuk menghapus data excel yang diinginkan pengguna. Selanjutnya kondisi ketiga, jika pengguna memilih opsi 3 (Edit Data), maka akan memanggil dan mengeksekusi fungsi `edit_data`, yang sebelumnya dibuat, yang fungsi tersebut merupakan fungsi untuk mengedit data excel yang diinginkan pengguna. Kemudian terakhir kita menggunakan pendekatan `to_excel` untuk menyimpan DataFrame ("data\_semua") yang telah diperbarui oleh pengguna ke file excel dengan nama "file\_name"

#### D. Input dan Output Menambahkan Data Statistik Excel

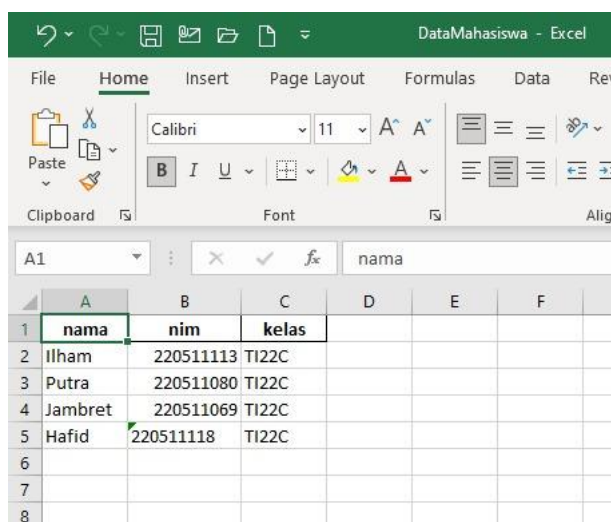


```

PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Tugas\Pertemuan_10> & C:/Python39/python.exe "c:/Users/ASUS/Documents/TUGAS KAMPUS/SEMESTER 4/PBO Lanjut/File Tugas/
Menu:
1. Tambah Data
2. Hapus Data
3. Update Data
Pilih menu (1/2/3): 1
Masukkan nama: Hafid
Masukkan NIM: 220511118
Masukkan kelas: TI22C
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Tugas\Pertemuan_10>

```

Kali ini kita akan melakukan uji coba pada program yang kita buat, pertama kita akan mencoba menambah data baru. Jika sudah melakukan input data baru, maka kita cek dengan membuka file excel yang kita buat.



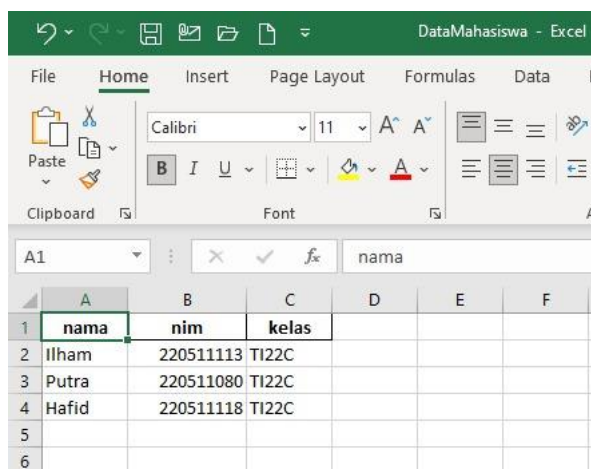
	A	B	C	D	E	F
1	nama	nim	kelas			
2	Ilham	220511113	TI22C			
3	Putra	220511080	TI22C			
4	Jambret	220511069	TI22C			
5	Hafid	220511118	TI22C			
6						
7						
8						

Bisa kita lihat, data yang kita input Nama "Hafid" di terminal, sekarang sudah bertambah dan masuk ke file excel.

## E. Input dan Output Menhapus Data Statistik Excel

```
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Tugas\Pertemuan_10> & C:/U
exe "c:/Users/ASUS/Documents/TUGAS KAMPUS/SEMESTER 4/PBO Lanjut/File Tugas/Pertemuan_10/coba4
Menu:
1. Tambah Data
2. Hapus Data
3. Update Data
Pilih menu (1/2/3): 2
Masukkan nama kolom untuk menghapus data: nama
Masukkan nilai dalam kolom 'nama' yang ingin dihapus: Jambret
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Tugas\Pertemuan_10>
```

Setelah kita berhasil menginput data dan sudah masuk ke file excel, sekarang kita akan mencoba menghapus data yang kita inginkan di file excel.



	A	B	C	D	E	F
1	nama	nim	kelas			
2	Ilham	220511113	TI22C			
3	Putra	220511080	TI22C			
4	Hafid	220511118	TI22C			
5						
6						

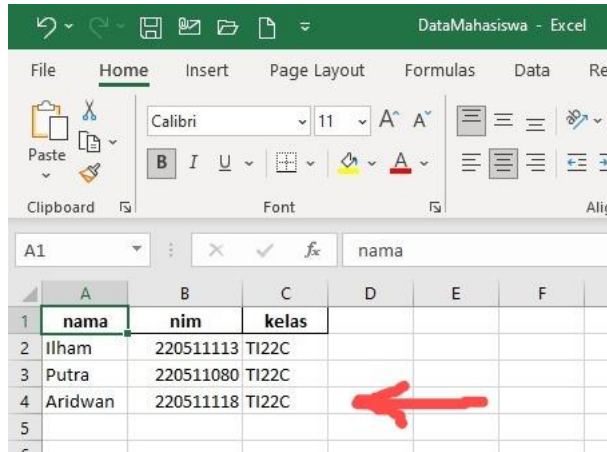
Seperti yang anda lihat, untuk kolom nama yang memiliki nilai/nama "Jambret" sudah terhapus dari file excel.

## F. Input dan Output Mengupdate Data Statistik Excel

```
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Tugas\Pertemuan_10> & C
exe "c:/Users/ASUS/Documents/TUGAS KAMPUS/SEMESTER 4/PBO Lanjut/File Tugas/Pertemuan_10/c
Menu:
1. Tambah Data
2. Hapus Data
3. Update Data
Pilih menu (1/2/3): 3
Data yang tersedia:
    nama      nim      kelas
0  Ilham  220511113  TI22C
1  Putra  220511080  TI22C
2  Hafid  220511118  TI22C
Masukkan indeks data yang ingin diedit: 2
Masukkan nama kolom yang ingin diubah (nama/nim/kelas): nama
Masukkan nilai baru: Aridwan
PS C:\Users\ASUS\Documents\TUGAS KAMPUS\SEMESTER 4\PBO Lanjut\File Tugas\Pertemuan_10>
```

Lalu yang terakhir kita akan mencoba untuk mengedit data pada file excel yang sebelumnya telah diinput. Pada gambar kita akan memilih baris mana (indeks),

dan kolom mana yang akan diedit nilai/namanya. kemudian jika sudah memilih atau menyeleksi baris dan kolom yang akan diedit, kita akan memperbarui nama/nilai baru pada baris dan kolom yang kita seleksi.



Bisa dilihat pada file excel, nama dan baris kolom yang sebelumnya memiliki nama "Hafid" sekarang sudah diUpdate menjadi nama Aridwan.



## **BAB 5**

### **PENUTUP**

#### **A. Kesimpulan**

Python adalah bahasa pemrograman yang mudah dipelajari dan berfokus pada keterbacaan kode. Diklaim memiliki sintaksis yang jelas, lengkap, dan mudah dipahami. Mendukung berbagai paradigma pemrograman, termasuk pemrograman berorientasi objek, imperatif, dan fungsional. Sebagai bahasa multi-paradigma, Python digunakan untuk berbagai keperluan pengembangan perangkat lunak dan dapat berjalan di berbagai platform sistem operasi. Jupyter, salah satu produk utamanya, adalah aplikasi web open-source yang memungkinkan pembuatan dan berbagi dokumen interaktif yang menggabungkan kode live, persamaan, visualisasi, dan teks naratif. Pandas, library Python, menyediakan alat untuk manipulasi dan analisis data tabular, seperti spreadsheet atau database relasional. Excel, perangkat lunak pengolah data populer, sering digunakan untuk statistik, analisis data, dan pembuatan laporan.

Namun, untuk analisis data yang lebih kompleks dan efisien, penggunaan Excel dengan Python dan library seperti Pandas menjadi solusi yang efektif. Dengan menggunakan Jupyter Notebook, seluruh proses dapat disatukan dalam satu tempat, memungkinkan impor data, manipulasi, analisis menggunakan Pandas, serta dokumentasi interaktif.

#### **B. Saran**

Untuk meningkatkan kemampuan dalam mengolah data statistik menggunakan Python, ada beberapa saran yang bisa dipertimbangkan, perlu memperdalam pemahaman tentang bahasa pemrograman Python. Selanjutnya, eksplorasi dan kuasai library lain dalam ekosistem Python yang relevan dengan analisis data. Gunakan Jupyter Notebook untuk pengembangan interaktif yang efektif. Kolaborasi dengan rekan tim dan komunitas juga sangat bermanfaat. Dan pastikan untuk memelihara dan memperbarui lingkungan pengembangan Anda secara teratur. Dengan mengikuti saran ini, diharapkan dapat meningkatkan efisiensi dan kualitas dalam mengolah data statistik menggunakan Python.

## DAFTAR PUSTAKA

### Website:

<https://revou.co/panduan-teknis/python-read-excel>  
<https://dqlab.id/mengulik-pandas-python-library-powerful-dalam-pengolahan-data>  
<https://revou.co/panduan-teknis/python-data-cleaning>  
<https://revou.co/kosakata/visual-studio-code>  
<https://medium.com/@yasirabd/pengenalan-numpy-pandas-matplotlib-b90bafd36c0>  
<https://revou.co/panduan-teknis/seaborn-python>  
<https://360digitmg.com/blog/python-tabulate>  
<https://revou.co/panduan-teknis/library-python#:~:text=Pandas%20adalah%20library%20Python%20yang,utama%2C%20yaitu%20Series%20dan%20DataFrame.>

### Jurnal:

Ardianto, Dicky, and Arif Tri Widiyatmoko. "Color Detector in an Image using Python and Computer Vision Library." *Journal of Intelligent Systems and Information Technology* 1.1 (2024): 25-30.

Enterprise, Jubilee. *Python untuk Programmer Pemula*. Elex media komputindo, 2019.

Hendri, Rantau. *Pembuatan Kode Program Dan Simulasi Skema Masakan ACD Menggunakan Jupyter Notebook Pada Pemrograman Python Anaconda*. Diss. Politeknik LPP Yogyakarta, 2022.

Mait, Charolina Debora, et al. "Sistem Pendukung Keputusan Menggunakan Fuzzy Logic Tahani Untuk Penentuan Golongan Obat Sesuai Dengan Penyakit Diabetes." *Jurnal Media Infotama* 18.2 (2022): 344-353.

Setiabudidaya, Dedi. "Penggunaan Piranti Lunak Jupyter Notebook dalam Upaya Mensosialisasikan Open Science." (2018).

## TENTANG PENULIS



NIM : 220511113  
Nama : Muhammad Ilham Ramdhani  
Kelas : TI22C  
Prodi : Teknik Informatika  
Fakultas : Teknik  
Instansi : Universitas Muhammadiyah Cirebon  
GitHub : [https://github.com/Antares023/PBO\\_Lanjut](https://github.com/Antares023/PBO_Lanjut)



NIM : 220511080  
Nama : M Putra Rizky Julianto  
Kelas : TI22C  
Prodi : Teknik Informatika  
Fakultas : Teknik  
Instansi : Universitas Muhammadiyah Cirebon  
GitHub : <https://github.com/MptrJ1807/PutraRizki-PBO-Lanjut.git>



NIM : 220511118  
Nama : Muhammad Hafid Aridwan  
Kelas : TI22C  
Prodi : Teknik Informatika  
Fakultas : Teknik  
Instansi : Universitas Muhammadiyah Cirebon  
GitHub : <https://github.com/muhhafidaridwan01/PEMROGRAMAN>



**PBO LANJUT**

