

**MODUL**

**PRAKTIKUM PBO**

Modul ini Disusun untuk Memenuhi Tugas Kelompok Mata Kuliah

Pemrograman Berorientasi Objek (PBO)

Dosen : Freddy Wicaksono, M.Kom



Nama Kelompok :

1. Ahmad Maulana (220511138) **TI22A**
2. Sugianto Tegar Samudra (220511162) **TI22A**
3. Daffa Ramadani Putra  (220511150) **TI22A**

**UNIVERSITAS MUHAMMADIYAH CIREBON**

**FAKULTAS TEKNIK**

**Teknik Informatika**

Kampus 1 : Jl. Tuparev No. 70 Cirebon 45153

Kampus 2 dan 3: Jl. Fatahillah - Watubelah - Cirebon

Telp. +62-231-209608, +62-231-204276, Fax: +62-231-209608

Email: info@umc.ac.id Website: [www.umc.ac.id](http://www.umc.ac.id)KATA PENGANTAR

Puji dan syukur kehadirat Allah Swt, karena berkat rahmat dan karunia-Nya kami dapat menyelesaikan modul ini tepat pada waktunya. Shalawat dan salam tidak lupa kami kirimkan selalu kepada baginda tercinta Nabi Muhamad Saw, sahabat, keluarga dan semua umatnya. Amiin

Modul ini memberikan dokumentasi tentang “Modul Praktikum Pbo” yang disusun untuk memenuhi salah satu tugas Mata Kuliah Pemrograman Berorientasi Objek (PBO). Dalam penyusunan modul ini, kami mendapat berbagai tantangan dan hambatan. Akan tetapi, dengan bantuan dan dukungan beberapa pihak tantangan itu bisa teratasi. Oleh karena itu, kami mengucapkan terima kasih yang sebesar besarnya kepada semua pihak yang telah membantu dan memberikan dukungan dalam penyusunan modul ini, terutama kepada Dosen Pengampu Mata Kuliah Pemrograman Berorientasi Objek (PBO) yang selalu memberikan arahan dan masukannya, semoga mendapat balasan yang setimpal dari Allah Swt. Kami menyadari modul ini jauh dari kata kesempurnaan, baik dari segi materi maupun penyusunannya. Oleh karena itu, kritik dan saran yang bersifat membangun kami harapkan dari semua pihak untuk penyempurnaan modul selanjutnya. Kami jauh dari kata sempurna. Dan ini merupakan langkah yang baik dari studi yang sesungguhnya. Oleh karena itu, keterbatasan waktu dan kemampuan kami, maka kritik dan saran yang membangun senantiasa kami harapkan semoga modul ini dapat berguna bagi kami pada khususnya dan pihak lain yang berkepentingan pada umumnya.

Akhir kata, kami berharap semoga modul ini dapat bermanfaat bagi kita semua. Akhirnya kepada Allah jualah kami serahkan semua pengorbanan serta perjuangan, demi langkah dan ayunan tangan kami semoga senantiasa mendapatkan rahmat dan maghfirah-Nya. Amin ya rabbal ‘alamin

Cirebon, 15 Januari 2024

Penulis

# PENDAHULUAN

Pemrograman Berorientasi Objek (PBO) merupakan paradigma pemrograman yang memandang program komputer sebagai kumpulan objek yang berinteraksi satu sama lain. Melalui mata kuliah Pemrograman II, mahasiswa akan diperkenalkan pada konsep dan praktik PBO dengan menggunakan bahasa pemrograman Python. Python dipilih karena kemudahan sintaksisnya, kemampuan PBO yang kuat, dan popularitasnya di berbagai bidang pengembangan perangkat lunak.

Mata kuliah ini bertujuan untuk membekali mahasiswa dengan pemahaman mendalam tentang konsep-konsep Pemrograman Berorientasi Objek serta keterampilan dalam menerapkan konsep tersebut dalam konteks pengembangan perangkat lunak. Selama satu semester, mahasiswa akan mengembangkan modul secara bertahap untuk mengasah kemampuan pemrograman mereka dan mengaplikasikan prinsip-prinsip PBO yang telah dipelajari.

Modul ini akan dirancang sedemikian rupa agar kami sebagai mahasiswa dapat mengasah keterampilan pemrograman, memahami konsep-konsep Pemrograman Berorientasi Objek, dan mampu mengaplikasikannya dalam pengembangan perangkat lunak yang kompleks. Dengan demikian, diharapkan bahwa mahasiswa akan mampu mengembangkan pemahaman yang kokoh tentang PBO dan memiliki keterampilan praktis yang dapat diterapkan dalam dunia industri.

# DAFTAR ISI

[PENDAHULUAN iv](#_Toc157275231)

[DAFTAR ISI v](#_Toc157275232)

[PERTEMUAN 1 1](#_Toc157275233)

[PERTEMUAN 2 8](#_Toc157275234)

[PERTEMUAN 3 13](#_Toc157275235)

[PERTEMUAN 4 16](#_Toc157275236)

[PERTEMUAN 5 : QUIZ 19](#_Toc157275237)

[PERTEMUAN 6 20](#_Toc157275238)

[PERTEMUAN 7 26](#_Toc157275239)

[PERTEMUAN 8 : UTS 29](#_Toc157275240)

[PERTEMUAN 9 30](#_Toc157275241)

[PERTEMUAN 10 36](#_Toc157275242)

[PERTEMUAN 11 42](#_Toc157275243)

[PERTEMUAN 12 54](#_Toc157275244)

[PERTEMUAN 13 65](#_Toc157275245)

[PERTEMUAN 14 Membuat modul praktikum PBO 81](#_Toc157275246)

# 

# PERTEMUAN 1

**Pengenalan Bahasa Pemrograman Python**

1. **Apa aitu Python?**

Python adalah bahasa pemrograman yang diinterpretasikan, artinya sebagai pengembang Anda menulis file Python (.py) di editor teks dan kemudian memasukkan file-file itu ke dalam interpreter python untuk dieksekusi. Bahasa Python dibuat oleh Guidovan Rossum, dan dirilis pada tahun 1991. Python dapat digunakan untuk Pengembangan web (sisi server), Pengembangan perangkat lunak atau membuat aplikasi (software), Menyelesaikan persamaan Matematika, Pembuatan skrip sistem dan Pemrograman Mikrokontroller (MicroPython).

Alasan untuk menggunakan dan mempelajari Python adalah Python dapat bekerja pada platform yang berbeda (Windows, Mac, Linux, Raspberry Pi, dll), Python memiliki sintaks sederhana yang mirip dengan bahasa Inggris, Python memiliki sintaks yang memungkinkan pengembang untuk menulis program dengan lebih sedikit baris daripada beberapa Bahasa pemrograman lainnya, Python berjalan pada sistem interpreter, artinya kode dapat dieksekusi segera setelah ditulis. Ini berarti pembuatan prototipe bisa sangat cepat, Python dapat diperlakukan dengan cara prosedural, cara berorientasi objek atau cara fungsional, dan Python memiliki banyak Pustaka.

1. **Instalasi Python**

Sebelum Anda menggunakan Python, Anda harus menginstalnya terlebih dahulu di sistem operasi komputer Anda. Saat ini Python memiliki 2 versi yang berbeda, yaitu Python versi 3.4.3 dan Python versi 2.7.10.

Dibawah adalah panduan cara instal python di platform Linux, Windows dan Mac OS:

* Linux

1. Buka browser, kunjungi http://www.python.org/downloads/source/
2. Download versi terbaru Python berbentuk file zip untuk Unix/Linux
3. Ekstrak file zip yang baru saja di download
4. Edit file Modules/Setup jika Anda ingin kostumisasi Python
5. Jalankan ./configure script
6. make
7. make install

Langkah ini akan menginstal Python di lokasi standar /usr/local/bin dan library di /usr/local/lib/pythonXX dimana XX adalah versi terbaru Python yang anda gunakan.

* Windows

1. Buka browser, kunjungi http://www.python.org/downloads/windows/
2. ATAU, klik direct link https://www.python.org/ftp/python/3.8.1/python-3.8.1.exe
3. Buka (klik 2x) file installer python yang baru saja di download
4. Ikuti langkah instalasi sampai selesai

* Mac OS

1. Buka browser, kunjungi http://www.python.org/download/mac/
2. Download versi terbaru Python untuk Macintosh
3. Buka file yang baru saja di download
4. Ikuti langkah instalasi sampai selesai
5. **Menjalankan Python**

Untuk menjalankan Python ada banyak cara yang bisa dilakukan. Anda bisa menggunakan shell, terminal atau menggunakan IDE (Integrated Development Environment). Di bawah ini adalah langkah-langkah menjalankan Python dengan cara yang paling mudah.

* Linux

1. Buka terminal CTRL+ALT+T
2. Ketik python maka Anda akan masuk ke Python shell.
3. Tuliskan script Python Anda, contoh: print("Selamat datang di Python"). jika sudah tekan tombol ENTER, dan script Python akan dijalankan/eksekusi.
4. Untuk keluar dari Python shell ketik exit()

*Atau*

1. Gunakan teks editor, misalnya gedit.
2. Buat file baru, dan ketikan script python Anda, contoh: print("Selamat datang di Python").
3. Save As dengan ekstensi .py (contoh: cetak.py).
4. Jalankan file dengan menggunakan Terminal.
5. Buka terminal CTRL+ALT+T.
6. Masuk ke direktori dimana file Python Anda disimpan (contoh: cd /Users/admin/Desktop/).
7. Jalankan script Python dengan menggunakan python diikuti dengan nama file (contoh: python cetak.py).
8. Script Python Anda akan dieksekusi/dijalankan.

* Windows

1. Untuk menjalankan script yang disimpan dalam file, buka IDLE (python shell di windows), Anda bisa mencarinya di tombol START.
2. Klik menu File - New File
3. Tulis script Python pada window yang muncul, contoh:

print("Belajar Python")

print("di belajarpython.com")

1. Simpan script lewat menu File - Save
2. Jalankan program dengan klik menu Run - Run Module

* Mac OS

1. Buka terminal.
2. Ketik python maka Anda akan masuk ke Python shell.
3. Tuliskan script Python Anda, contoh: print("Selamat datang di Python"). jika sudah tekan tombol ENTER, dan script Python akan dijalankan/eksekusi.
4. Untuk keluar dari Python shell ketik exit()

*Atau*

1. Gunakan teks editor.
2. Buat file baru, dan ketikan script python Anda, contoh: print("Selamat datang di Python").
3. Save As dengan ekstensi .py (contoh: cetak.py).
4. Jalankan file dengan menggunakan Terminal.
5. Buka terminal CTRL+ALT+T
6. Masuk ke direktori dimana file Python Anda disimpan (contoh: cd /Users/admin/Desktop/).
7. Jalankan script Python dengan menggunakan python diikuti dengan nama file (contoh: python cetak.py).
8. Script Python Anda akan dieksekusi/dijalankan.
9. **Tipe Data**

Tipe data adalah suatu media atau memori pada komputer yang digunakan untuk menampung informasi.

|  |  |
| --- | --- |
| Tipe Data | Contoh |
| Boolean | x = True |
| String | x = "Hello World" |
| Integer | x = 20 |
| Float | x = 20.5 |
| Hexadecimal | x = 9a atau 1d3 |
| Complex | x = 1j |
| List | x = ["apple", "banana", "cherry"] |
| Tuple | x = ("apple", "banana", "cherry") |
| Dictionary | x = {'nama': 'adi','id':2} |
| Range | x = range(6) |
| Set | x = {"apple", "banana", "cherry"} |
| Frozenset | x = frozenset({"apple", "banana", "cherry"}) |
| Bytes | x = b"Hello" |
| Bytearray | x = bytearray(5) |
| Memoryview | x = memoryview(bytes(5)) |

1. **Variable**

Variabel adalah lokasi memori yang dicadangkan untuk menyimpan nilai-nilai. Ini berarti bahwa ketika Anda membuat sebuah variabel Anda memesan beberapa ruang di memori. Variabel menyimpan data yang dilakukan selama program dieksekusi, yang nantinya isi dari variabel tersebut dapat diubah oleh operasi - operasi tertentu pada program yang menggunakan variabel.

Penulisan variabel Python sendiri juga memiliki aturan tertentu, yaitu :

1. Karakter pertama harus berupa huruf atau garis bawah/underscore \_
2. Karakter selanjutnya dapat berupa huruf, garis bawah/underscore \_ atau angka
3. Karakter pada nama variabel bersifat sensitif (case-sensitif). Artinya huruf kecil dan huruf besar dibedakan. Sebagai contoh, variabel namaDepan dan namadepan adalah variabel yang berbeda.

Dibawah ini adalah contoh penggunaan variabel dalam bahasa pemrograman Python :

#proses memasukan data ke dalam variabel

nama = "John Doe"

#proses mencetak variabel

print(nama)

#nilai dan tipe data dalam variabel dapat diubah

umur = 20 #nilai awal

print(umur) #mencetak nilai umur

type(umur) #mengecek tipe data umur

umur = "dua puluh satu" #nilai setelah diubah

print(umur) #mencetak nilai umur

type(umur) #mengecek tipe data umur

namaDepan = "Budi"

namaBelakang = "Susanto"

nama = namaDepan + " " + namaBelakang

umur = 22

hobi = "Berenang"

print("Biodata\n", nama, "\n", umur, "\n", hobi)

#contoh variabel lainya

inivariabel = "Halo"

ini\_juga\_variabel = "Hai"

\\_inivariabeljuga = "Hi"

inivariabel222 = "Bye"

panjang = 10

lebar = 5

luas = panjang \\* lebar

print(luas)

1. **Operator**

Operator adalah konstruksi yang dapat memanipulasi nilai dari operan. Sebagai contoh operasi 3 + 2 = 5. Disini 3 dan 2 adalah operan dan + adalah operator.

Bahasa pemrograman Python mendukung berbagai macam operator, diantaranya :

1. Operator Aritmatika Python

Operator aritmatika digunakan dengan nilai numerik untuk melakukan operasi matematika umum:

|  |  |  |
| --- | --- | --- |
| **Contoh** | **Nama** | **Operator** |
| + | Tambahan | x + y |
| - | Pengurangan | x – y |
| \* | Perkalian | x \* y |
| / | Divisi | x / y |
| % | Modulus | x % y |
| \*\* | Eksponen | x \*\* y |
| // | Loor division | x // y |

1. Operator Penugasan Python

Operator penugasan digunakan untuk menetapkan nilai ke variabel:

|  |  |  |
| --- | --- | --- |
| Contoh | Operator | Sama seperti |
| = | x = 5 | x = 5 |
| + = | x + = 3 | x = x + 3 |
| - = | x - = 3 | x = x - 3 |
| \* = | x \* = 3 | x = x \* 3 |
| / = | x / = 3 | x = x / 3 |
| % = | x % = 3 | x = x % 3 |
| // = | x // = 3 | x = x // 3 |
| \*\* = | x \*\* = 3 | x = x \*\* 3 |
| & = | x & = 3 | x = x & 3 |
| | = | x | = 3 | x = x | 3 |
| ^ = | x ^= 3 | x = x ^ 3 |
| >> = | x >> = 3 | x = x >> 3 |
| << = | x << = 3 | x = x << 3 |

1. Operator Perbandingan Python Operator

perbandingan digunakan untuk membandingkan dua nilai

|  |  |  |
| --- | --- | --- |
| **Contoh** | **Nama** | **Operator** |
| = = | Equal | x = = y |
| ! = | Not equal | x != y |
| > | Greater than | x > y |
| < | Less than | x < y |
| > = | Greater than or equal to | x > = y |
| < = | Less than or qequal to | x < = y |

1. Operator Logika Python

Operator logika digunakan untuk menggabungkan pernyataan kondisional:

|  |  |  |
| --- | --- | --- |
| **Contoh** | **Deskripsi** | **Operator** |
| And | Mengembalikan True jika kedua pernyataan benar | x < 5 and x < 10 |
| or | Mengembalikan True jika salah satu pernyataan benar | x < 5 or x < 4 |
| not | Membalikkan hasilnya, mengembalikan False jika hasilnya benar | not(x < 5 and x < 10) |

1. Operator Identitas Python

Operator identitas digunakan untuk membandingkan objek, bukan jika mereka sama, tetapi jika mereka sebenarnya adalah objek yang sama, dengan lokasi memori yang sama:

|  |  |  |
| --- | --- | --- |
| **Contoh** | **Deskripsi** | **Operator** |
| Is | Mengembalikan True jika kedua variabel adalah objek yang sama | x is y |
| Is not | Mengembalikan True jika kedua variabel bukan objek yang sama | x is not y |

1. Operator Keanggotaan Python

Operator keanggotaan digunakan untuk menguji apakah urutan disajikan dalam suatu objek:

|  |  |  |
| --- | --- | --- |
| **Contoh** | **Deskripsi** | **Opertor** |
| In | Mengembalikan True jika urutan dengan nilai yang ditentukan ada di objek | x in y |
| Not in | Mengembalikan True jika urutan dengan nilai yang ditentukan tidak ada dalam objek | x not in y |

1. Operator Bitwise Python

Operator bitwise digunakan untuk membandingkan angka (biner):

|  |  |  |
| --- | --- | --- |
| **Deskripsi** | **Nama** | **Operator** |
| & | AND | Set setiap bit ke 1 jika kedua bit adalah 1 |
| | | OR | Set setiap bit ke 1 jika salah satu dari dua bit adalah 1 |
| ^ | XOR | Setel setiap bit menjadi 1 jika hanya satu dari dua bit yang bernilai 1 |
| ~ | NOT | Membalikkan semua bit |
| << | Zero fill left shift | Geser ke kiri dengan menekan angka nol dari kanan dan biarkan bit paling kiri jatuh |
| >> | Signed right shift | Geser ke kanan dengan mendorong salinan bit paling kiri ke dalam dari kiri, dan biarkan bit paling kanan jatuh |

# PERTEMUAN 2

**IF, Loop, Array**

* **IF**

1. Kondisi If

Pengambilan keputusan (kondisi if) digunakan untuk mengantisipasi kondisi yang terjadi saat jalanya program dan menentukan tindakan apa yang akan diambil sesuai dengan kondisi.

Pada python ada beberapa statement/kondisi diantaranya adalah if, else dan elif Kondisi if digunakan untuk mengeksekusi kode jika kondisi bernilai benar True.

Jika kondisi bernilai salah False maka statement/kondisi if tidak akan di-eksekusi.

Dibawah ini adalah contoh penggunaan kondisi if pada Python

#Kondisi if adalah kondisi yang akan dieksekusi oleh program jika bernilai benar atau TRUE

nilai = 9

#jika kondisi benar/TRUE maka program akan mengeksekusi perintah dibawahnya

if(nilai > 7):

    print("Sembilan Lebih Besar Dari Angka Tujuh") # Kondisi Benar, Dieksekusi

#jika kondisi salah/FALSE maka program tidak akan mengeksekusi perintah dibawahnya

if(nilai > 10):

    print("Sembilan Lebih Besar Dari Angka Sepuluh") # Kondisi Salah, Maka tidak tereksekusi

Dari contoh diatas, jika program dijalankan maka akan mencetak string "Sembilan Lebih Besar Dari Angka Tujuh" sebanyak 1 kali yaitu pada if pertama. Di if kedua statement bernilai salah, jadi perintah print("Sembilan Lebih Besar Dari Angka Sepuluh") tidak akan dieksekusi.

1. Kondisi If Else

Pengambilan keputusan (kondisi if else) tidak hanya digunakan untuk menentukan tindakan apa yang akan diambil sesuai dengan kondisi, tetapi juga digunakan untuk menentukan tindakan apa yang akan diambil/dijalankan jika kondisi tidak sesuai.

Pada python ada beberapa statement/kondisi diantaranya adalah if, else dan elif Kondisi if digunakan untuk mengeksekusi kode jika kondisi bernilai benar.

Kondisi if else adalah kondisi dimana jika pernyataan benar True maka kode dalam if akan dieksekusi, tetapi jika bernilai salah False maka akan mengeksekusi kode di dalam else.

Dibawah ini adalah contoh penggunaan kondisi if else pada Python

# Kondisi if else adalah jika kondisi bernilai TRUE maka akan dieksekusi pada if,

# tetapi jika bernilai FALSE maka akan dieksekusi kode pada else

nilai = 3

# Jika pernyataan pada if bernilai TRUE maka if akan dieksekusi,

# tetapi jika FALSE kode pada else yang akan dieksekusi.

if(nilai > 7):

    print("Selamat Anda Lulus")

else:

    print("Maaf Anda Tidak Lulus")

Pada contoh diatas, jika program dijalankan maka akan mencetak string "Maaf Anda Tidak Lulus" karena pernyataan pada if bernilai False

1. Kondisi Elif

Pengambilan keputusan (kondisi if elif) merupakan lanjutan/percabangan logika dari "kondisi if". Dengan elif kita bisa membuat kode program yang akan menyeleksi beberapa kemungkinan yang bisa terjadi. Hampir sama dengan kondisi "else", bedanya kondisi "elif" bisa banyak dan tidak hanya satu.

Dibawah ini adalah contoh penggunaan kondisi elif pada Python

#Contoh penggunaan kondisi elif

hari\_ini = "Minggu"

if(hari\_ini == "Senin"):

    print("Saya akan kuliah")

elif(hari\_ini == "Selasa"):

    print("Saya akan kuliah")

elif(hari\_ini == "Rabu"):

    print("Saya akan kuliah")

elif(hari\_ini == "Kamis"):

    print("Saya akan kuliah")

elif(hari\_ini == "Jumat"):

    print("Saya akan kuliah")

elif(hari\_ini == "Sabtu"):

    print("Saya akan kuliah")

elif(hari\_ini == "Minggu"):

    print("Saya akan libur")

Pada contoh diatas, jika program dijalankan maka akan mencetak string "Saya akan libur"

* **LOOP**

1. Loop Python

Secara umum, pernyataan pada bahasa pemrograman akan dieksekusi secara berurutan. Pernyataan pertama dalam sebuah fungsi dijalankan pertama, diikuti oleh yang kedua, dan seterusnya. Tetapi akan ada situasi dimana Anda harus menulis banyak kode, dimana kode tersebut sangat banyak. Jika dilakukan secara manual maka Anda hanya akan membuang-buang tenaga dengan menulis beratus-ratus bahkan beribu-ribu kode. Untuk itu Anda perlu menggunakan pengulangan di dalam bahasa pemrograman Python.

Di dalam bahasa pemrograman Python pengulangan dibagi menjadi 3 bagian, yaitu :

* While Loop
* For Loop
* Nested Loop

1. While Loop

Pengulangan While Loop di dalam bahasa pemrograman Python dieksesusi statement berkali-kali selama kondisi bernilai benar atau True.

Dibawah ini adalah contoh penggunaan pengulangan While Loop.

#Contoh penggunaan While Loop

#Catatan: Penentuan ruang lingkup di Python bisa menggunakan tab alih-alih menggunakan tanda kurung

count = 0

while (count < 9):

  print ("The count is: ", count)

  count = count + 1

print ("Good bye!")

1. For Loop

Pengulangan for pada Python memiliki kemampuan untuk mengulangi item dari urutan apapun, seperti list atau string.

Dibawah ini adalah contoh penggunaan pengulangan For Loop.

#Contoh pengulangan for sederhana

angka = [1,2,3,4,5]

for x in angka:

  print(x)

#Contoh pengulangan for

buah = ["nanas", "apel", "jeruk"]

for makanan in buah:

  print ("Saya suka makan", makanan)

1. Nested Loop

Bahasa pemrograman Python memungkinkan penggunaan satu lingkaran di dalam loop lain. Bagian berikut menunjukkan beberapa contoh untuk menggambarkan konsep tersebut.

Dibawah ini adalah contoh penggunaan Nested Loop.

#Contoh penggunaan Nested Loop

#Catatan: Penggunaan modulo pada kondisional mengasumsikan nilai selain nol sebagai True(benar) dan nol sebagai False(salah)

i = 2

while(i < 100):

  j = 2

while(j <= (i/j)):

  if not(i%j): break

  j = j + 1

  if (j > i/j) : print(i, " is prime")

    i = i + 1

print("Good bye!")

* **ARRAY**

Array adalah variabel khusus yang dapat menampung lebih dari satu nilai dalam satu waktu. Jika Anda memiliki daftar item (misalnya daftar nama mobil), menyimpan mobil dalam variabel tunggal akan terlihat seperti ini:

Catatan: **“Python tidak memiliki dukungan bawaan untuk Array, namun Daftar Python dapat digunakan sebagai gantinya.”**

Contoh

Buat array yang berisi nama mobil:

cars = ["Ford", "Volvo", "BMW"]

Metode Array

|  |  |
| --- | --- |
| Method | Description |
| append() | Menambahkan elemen di akhir daftar |
| clear() | Menghapus semua elemen dari daftar |
| copy() | Mengembalikan salinan daftar |
| count() | Mengembalikan jumlah elemen dengan nilai yang ditentukan |
| extend() | Tambahkan elemen daftar (atau elemen apa pun yang dapat diubah), ke akhir daftar saat ini |
| index() | Mengembalikan indeks elemen pertama dengan nilai yang ditentukan |
| insert() | Menambahkan elemen pada posisi yang ditentukan |
| pop() | Menghapus elemen pada posisi yang ditentukan |
| remove() | Menghapus item pertama dengan nilai yang ditentukan |
| reverse() | Membalikkan urutan daftar |
| sort() | Urutkan daftarnya |

# PERTEMUAN 3

**Function**

Fungsi/function adalah blok kode terorganisir dan dapat digunakan kembali yang digunakan untuk melakukan sebuah tindakan/action. Mendefinisikan Fungsi Python Anda dapat menentukan fungsi untuk menyediakan fungsionalitas yang dibutuhkan.

Berikut adalah aturan sederhana untuk mendefinisikan fungsi dengan Python.

* Fungsi blok dimulai dengan def kata kunci diikuti oleh nama fungsi dan tanda kurung (()).
* Setiap parameter masukan atau argumen harus ditempatkan di dalam tanda kurung ini. Anda juga dapat menentukan parameter di dalam tanda kurung ini.
* Pernyataan pertama dari sebuah fungsi dapat berupa pernyataan opsional - string dokumentasi fungsi atau docstring.
* Blok kode dalam setiap fungsi dimulai dengan titik dua (:) dan indentasi.
* Pernyataan kembali [ekspresi] keluar dari sebuah fungsi, secara opsional menyampaikan kembali ekspresi ke pemanggil. Pernyataan pengembalian tanpa argumen sama dengan return None.

Contoh fungsi

def kalikan(bilangan1, bilangan2):

    hasil= bilangan1\*bilangan2

    return hasil

A=5

B=8

Nilai = kalikan(A, B)

print("A =", A)

print("B =", B)

print('Nilai =', Nilai)

Contoh fungsi dalam aplikasi kalkulator balok :

import tkinter as tk

from tkinter import Frame, Label, Entry, Button, END, W

def hitung\_luas():

    p = float(txtpanjang.get())

    l = float(txtlebar.get())

    t = float(txttinggi.get())

    Luas = 2\*(p\*l) + 2\*(p\*t) + 2\*(l\*t)

    txtLuas.delete(0,END)

    txtLuas.insert(END,Luas)

def hitung\_volume():

    p = float(txtpanjang.get())

    l = float(txtlebar.get())

    t = float(txttinggi.get())

    Volume = p \* l \* t

    txtvolume.delete(0,END)

    txtvolume.insert(END,Volume)

def hitung():

    hitung\_luas()

    hitung\_volume()

*# Create tkinter object*

app = tk.Tk()

*# Tambahkan judul*

app.title("Kalkulator Luas dan Volume Balok")

*# Windows*

frame = Frame (app)

frame.pack(*padx*=50, *pady*=50)

*#Label Nama*

Nama =Label(frame, *text*="AHMAD MAULANA 220511138")

Nama.grid(*row*=0, *column*=0, *sticky*=W, *padx*=5, *pady*=5)

*# Label Panjang*

panjang= Label (frame, *text*="Panjang: ")

panjang.grid(*row*=1, *column*=0, *sticky*=W, *padx*=5, *pady*=5)

*# Label Lebar*

lebar = Label(frame, *text*="Lebar:")

lebar.grid(*row*=2, *column*=0, *sticky*=W, *padx*=5, *pady*=5)

*# Label Tinggi*

tinggi = Label(frame, *text*="Tinggi")

tinggi.grid(*row*=3, *column*=0, *sticky*=W, *padx*=5, *pady*=5)

*# Textbox Panjang*

txtpanjang = Entry (frame)

txtpanjang.grid(*row*=1, *column*=1)

*# Textbox Lebar*

txtlebar = Entry (frame)

txtlebar.grid(*row*=2, *column*=1)

*# Textbox Tinggi*

txttinggi = Entry (frame)

txttinggi.grid(*row*=3, *column*=1)

*# Button*

hitung\_button = Button (frame, *text*="Hitung", *command*=hitung)

hitung\_button.grid(*row*=4, *column*=1, *sticky*=W, *padx*=5, *pady*=5)

*# Output Label Luas*

luas = Label(frame, *text*="Luas: ")

luas.grid(*row*=5, *column*=0, *sticky*=W, *padx*=5, *pady*=5)

*# Output Label Volume*

volume = Label (frame, *text*="Volume: ")

volume.grid(*row*=6, *column*=0, *sticky*=W, *padx*=5, *pady*=5)

*# Output Textbox Luas*

txtLuas = Entry (frame)

txtLuas.grid(*row*=5, *column*=1, *sticky*=W, *padx*=5, *pady*=5)

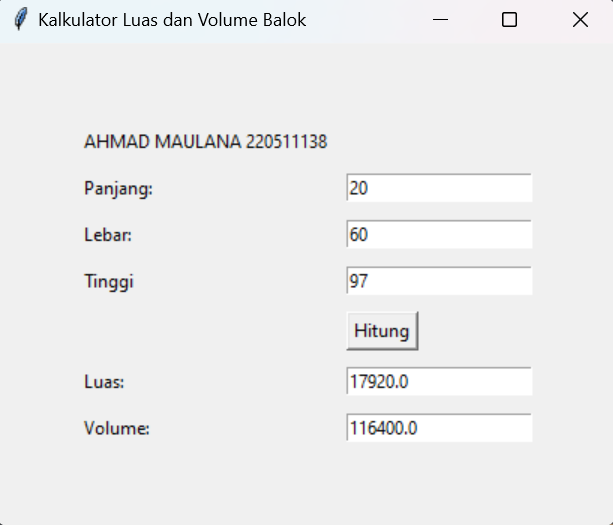
*# Output Textbox Volume*

txtvolume = Entry (frame)

txtvolume.grid(*row*=6, *column*=1, *sticky*=W, *padx*=5, *pady*=5)

app.mainloop()

Output dari aplikasi kalkulator balok :



# PERTEMUAN 4

**Module**

Modul adalah objek Python dengan atribut yang diberi nama yang bisa Anda bind dan dijadikan referensi. Secara sederhana modul adalah file yang terdiri dari kode Python. Modul dapat mendefinisikan fungsi, kelas dan variabel. Modul juga bisa menyertakan kode yang bisa dijalankan "runable".

Pada pertemuan ke-4 ini kami membuat aplikasi Perhitungan Luas dan Volume Bangun Ruang dengan Modul Python, berikut Langkah-langkahnya:

1. Membuat file berisi fungsi-fungsi modul, disini saya membuat file dengan nama main.py:

from math import pi

def volume\_kubus(rusuk):

    return rusuk \* rusuk \* rusuk

def luas\_volume(rusuk):

    return 6 \* rusuk \* rusuk

1. Membuat file untuk membuat aplikasinya:

Pada modul ini, Anda akan belajar cara membuat Kalkulator Kubus sederhana menggunakan pustaka Tkinter dalam bahasa pemrograman Python. Kalkulator ini dapat menghitung luas permukaan dan volume dari sebuah kubus berdasarkan panjang sisi yang dimasukkan oleh pengguna.

* **Pembagian Kode:**

from main import \*

import tkinter as tk

from tkinter import Frame, Label, Entry, Button, END, W

Kode mengimpor modul yang diperlukan, termasuk `main`, yang merupakan file lain dengan fungsionalitas tambahan.

* **Definisi Fungsi:**

def hitung\_luas():

    rusuk = float(txtrusuk.get())

    luas = round(6\*rusuk\*\*2)

    txtLuas.delete(0, END)

    txtLuas.insert(END,luas)

def hitung\_volume():

    rusuk = float(txtrusuk.get())

    Volume = round(rusuk\*\*3)

    txtVolume.delete(0,END)

    txtVolume.insert(END, Volume)

def hitung():

    hitung\_luas()

    hitung\_volume()

Fungsi-fungsi ini mendefinisikan perhitungan luas permukaan, volume, dan fungsi gabungan untuk melakukan kedua perhitungan tersebut.

* **Membuat Aplikasi Tkinter:**

app = tk.Tk()

app.title ("Kalkulator Mencari Luas dan Volome Kubus")

* **Membuat Elemen GUI:**

frame = Frame (app)

frame.pack(padx=50, pady=50)

Sebuah frame dibuat dan dipacking ke dalam jendela aplikasi dengan padding yang ditentukan.

rusuk =Label(frame, text="Rusuk:")

rusuk.grid(row=1, column=0, sticky=W, padx=5, pady=5)

* **Label untuk panjang sisi input.**

txtrusuk = Entry(frame)

txtrusuk.grid(row=1, column=1,  sticky=W, padx=5, pady=5)

* **Widget Entry untuk pengguna memasukkan panjang sisi.**

hitung\_button = Button(frame, text="Hitung", command=hitung)

hitung\_button.grid(row=2, column=1, sticky=W, padx=5, pady=5)

* **Tombol untuk memicu perhitungan.**

luas= Label(frame, text="Luas: ")

luas.grid(row=3, column=0, sticky=W, padx=5, pady=5)

* **Label untuk menampilkan luas permukaan.**

volume = Label (frame, text="volume: ")

volume.grid(row=4, column=0, stick=W, padx=5, pady=5)

* **Label untuk menampilkan volume.**

txtLuas = Entry(frame)

txtLuas.grid(row=3, column=1, sticky=W, padx=5, pady=5)

* **Widget Entry untuk menampilkan luas permukaan yang dihitung.**

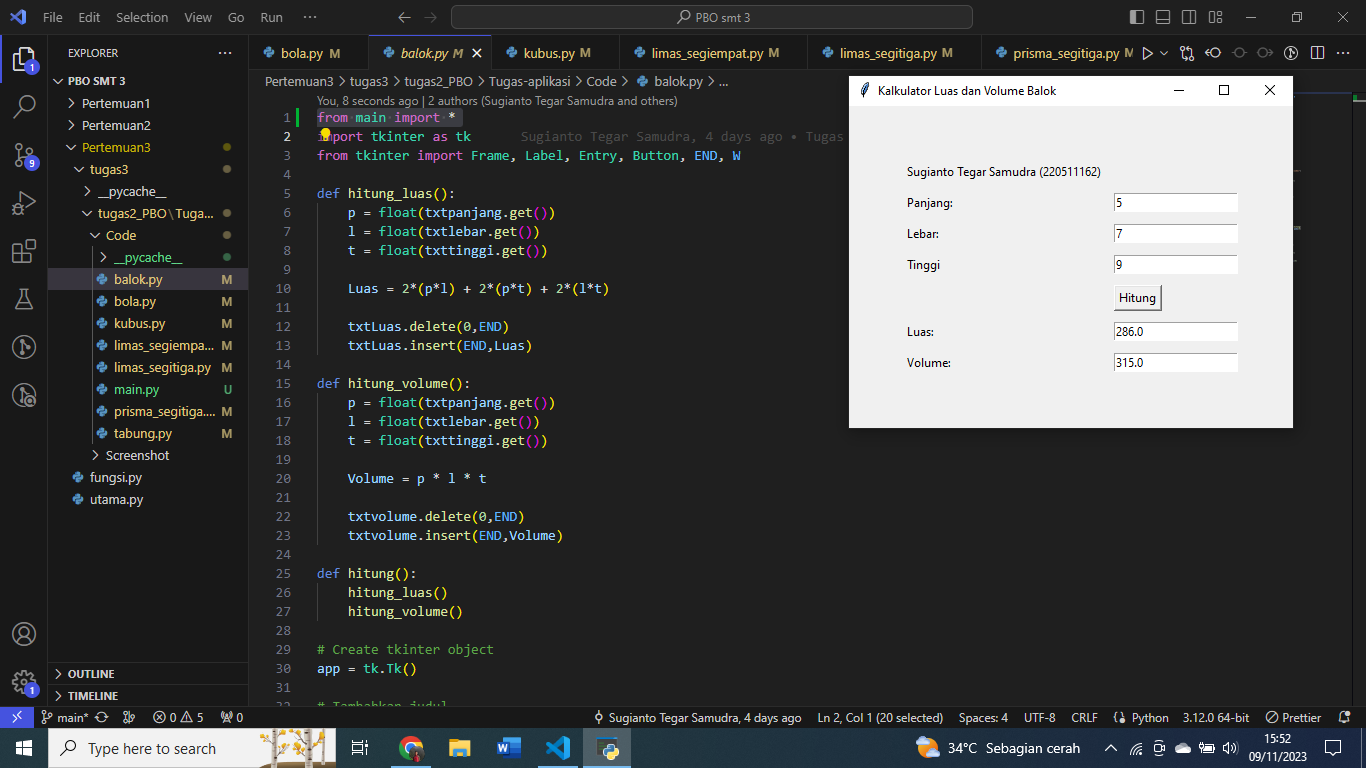
txtVolume = Entry (frame)

txtVolume.grid(row=4, column=1, sticky=W, padx=5, pady=5)

* **Fungsi untuk menjalan aplikasi Tkinter**

app.mainloop()

1. Output Source Code



Hasil output dari code di atas menampilkan sebuah aplikasi, di dalam aplikasi tersebut user dapat menghitung Luas dan Volume Balok dengan menginput Panjang, Lebar, dan Tinggi lalu klik tombol “Hitung” dan hasilnya akan terlihat di kolom bawahnya.

# PERTEMUAN 5 : QUIZ

# PERTEMUAN 6

**Membuat Aplikasi Bantu (Tools)**

* Membuat Aplikasi untuk menjalankan MP3

1. Install library “playsound” dapat menggunakan pip, manajer paket Python. Buka terminal atau command prompt, dan ketik perintah berikut: pip install playsound
2. Siapkan file lagu yang akan digunakan
3. Import modul

from playsound import playsound

import tkinter as tk

from tkinter import ttk

1. Definisi Fungsi play\_mp3:

def play\_mp3():

    playsound('tugas6/Gunslinger.mp3')

Fungsi ini dibuat untuk memutar file MP3 yang disebut "Gunslinger.mp3". File MP3 ini harus berada di direktori yang benar atau diberikan path lengkapnya.

1. Inisialisasi Tkinter:

app = tk.Tk()

app.title("MP3 Player")

Membuat objek aplikasi Tkinter (app) dan menetapkan judul jendela aplikasi menjadi "MP3 Player".

1. Membuat tombol Play:

button = ttk.Button(app, text="Play MP3", command=play\_mp3)

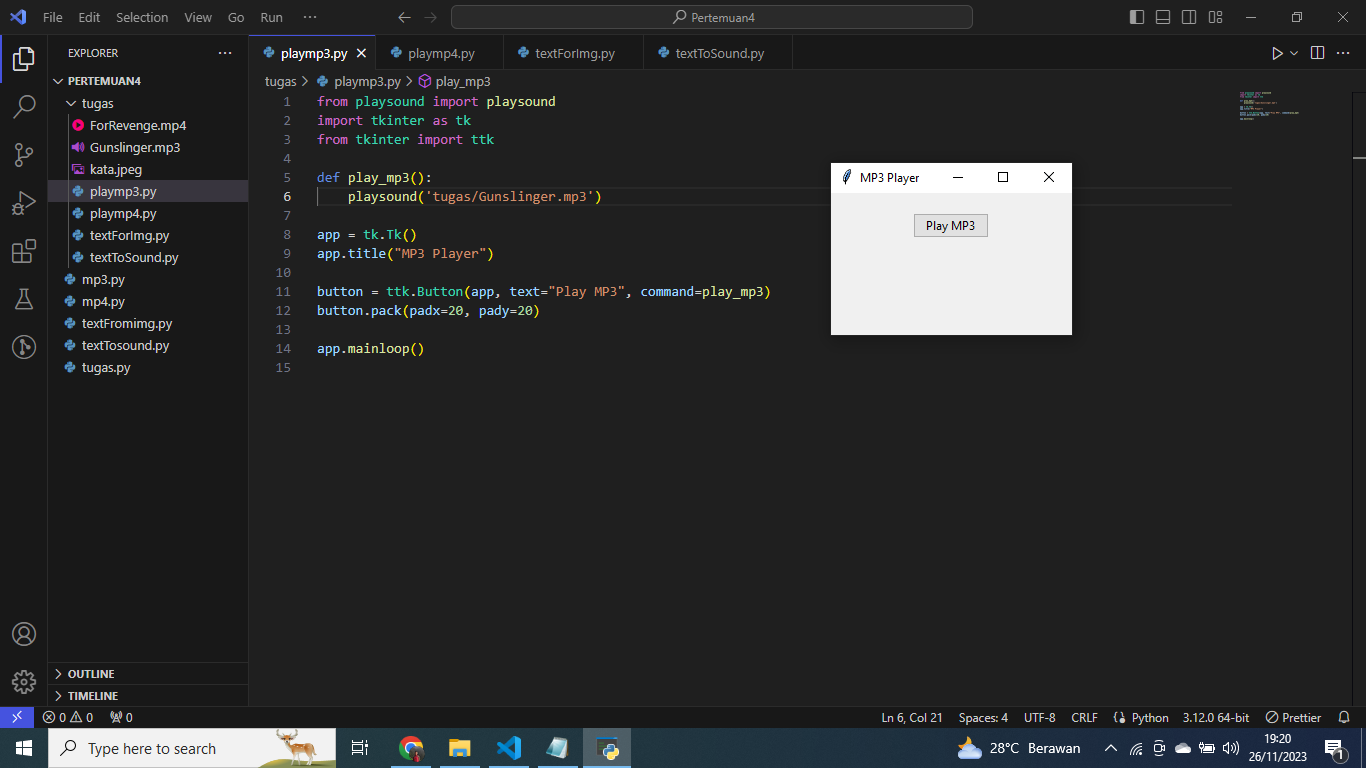
button.pack(padx=20, pady=20)

Membuat tombol Tkinter menggunakan modul ttk dengan teks "Play MP3" dan menetapkan fungsi play\_mp3 sebagai perintah yang akan dijalankan saat tombol ditekan.

1. Menjalankan Aplikasi Tkinter:

app.mainloop()

1. Output



Berikut tampilan sederhana untuk memutar Lagu dengan meng-klik “Play MP3” maka lagu yang telah diinputkan ke dalam Source code diatas akan otomatis berputar.

* Membuat Aplikasi MP4

1. Install library opencv, buka terminal atau command prompt dan jalankan perintah berikut:

pip install opencv-python

1. Siapkan file video yang akan dipakai
2. Import Modul:

import cv2

import tkinter as tk

from tkinter import ttk

Kode ini mengimpor modul cv2 untuk pengolahan gambar dan video dengan OpenCV, dan modul Tkinter untuk membuat antarmuka grafis.

1. Definisi Fungsi play\_video:

def play\_video():

    cap = cv2.VideoCapture('tugas6/ForRevenge.mp4')

    if not cap.isOpened():

        print("Error opening video file")

1. Definisi Fungsi update\_frame:

    def update\_frame():

        ret, frame = cap.read()

        if ret:

            \_, frame = cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB), cv2.resize(frame, (640, 480))

            photo = tk.PhotoImage(data=cv2.imencode('.ppm', frame)[1].tobytes())

            label.config(image=photo)

            label.image = photo

            label.after(10, update\_frame)

        else:

            cap.release()

    update\_frame()

Fungsi ini membaca setiap frame dari video, mengonversinya ke format yang dapat ditampilkan oleh Tkinter, dan menetapkan frame ke label di GUI. Fungsi ini kemudian dijadwalkan untuk dipanggil lagi setelah 10 milidetik menggunakan label.after(10, update\_frame).

1. Inisialisasi Tkinter:

app = tk.Tk()

app.title("OpenCV Video Player")

Membuat objek aplikasi Tkinter (app) dan menetapkan judul jendela aplikasi menjadi "OpenCV Video Player".

1. Membuat tombol play:

button = ttk.Button(app, text="Play Video", command=play\_video)

button.pack(padx=20, pady=20)

1. Membuat Label untuk Menampilkan Video:

label = ttk.Label(app)

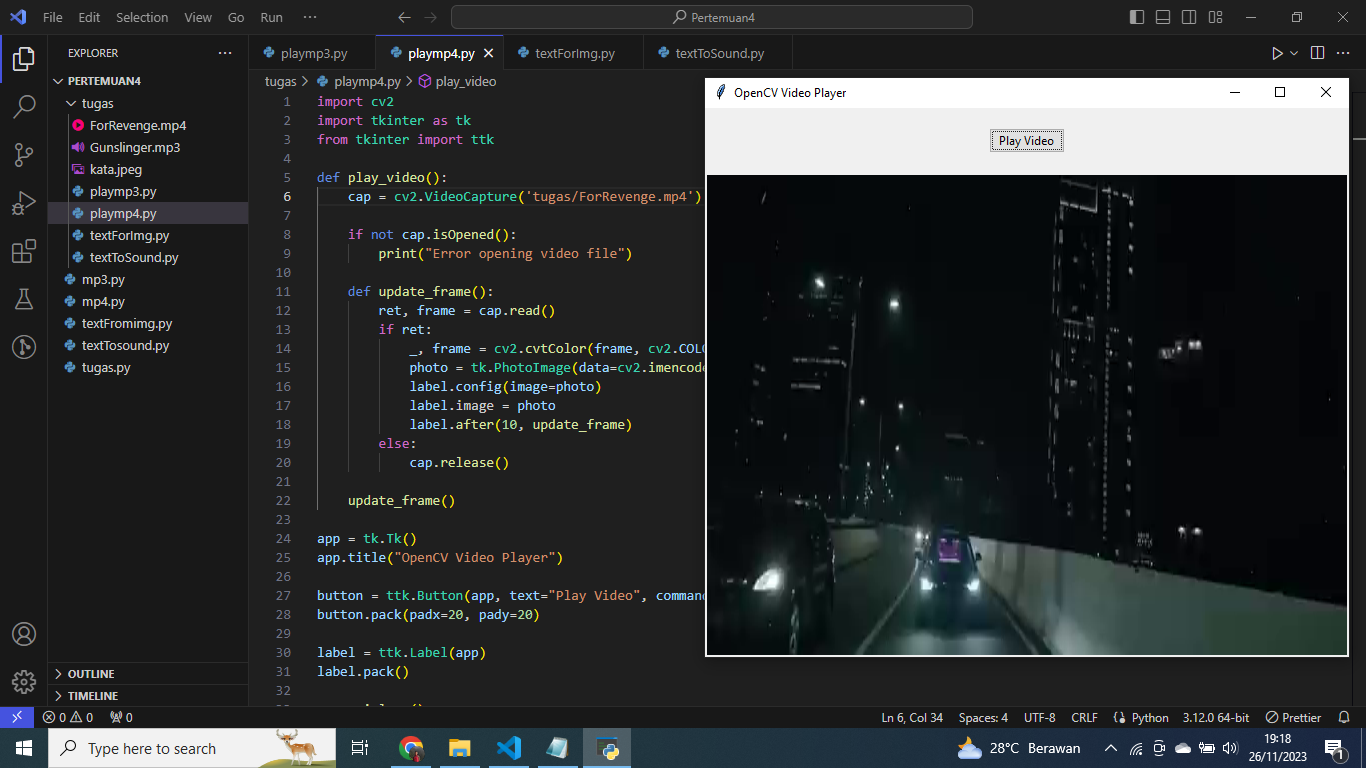
label.pack()

1. Menjalankan Aplikasi Tkinter:

app.mainloop()

Memanggil fungsi mainloop() untuk menjalankan loop utama Tkinter, sehingga menjaga aplikasi berjalan dan siap menerima interaksi pengguna.

1. Output



Setelah file dirunning maka akan muncul output sebagai berikut dengan menampilkan video dan sebuah tombol yang mana tombol “Play Video” untuk menjalankan videonya.

* Membuat Aplikasi mengubah gambar menjadi teks

1. Install library Pillow dan Pytesseract menggunakan pip pada terminal dengan mengetikan perintah:

pip install pillow

Perintah di atas akan menginstal modul Pillow yang digunakan untuk manipulasi gambar.

pip install pytesseract

1. Siapkan file gambar yang akan dikonversi
2. Import Modul

from PIL import Image, ImageTk

from pytesseract import pytesseract

import tkinter as tk

from tkinter import ttk

from tkinter import filedialog

Kode ini mengimpor modul Image dan ImageTk dari Pillow untuk manipulasi gambar, modul pytesseract untuk OCR, dan modul Tkinter untuk membuat antarmuka grafis.

1. Definisi Fungsi lakukan\_ocr:

def lakukan\_ocr():

    path\_to\_tesseract = r'C:\Program Files\Tesseract-OCR\tesseract.exe'

    pytesseract.tesseract\_cmd = path\_to\_tesseract

    path\_to\_image = 'tugas6/kata.jpeg'

    img = Image.open(path\_to\_image)

    text = pytesseract.image\_to\_string(img)

    result\_label.config(text=text)

* Fungsi ini dijalankan saat tombol "Play Program" ditekan.
* Menetapkan path ke file eksekusi Tesseract OCR dan menetapkan path gambar yang akan di-OCR.
* Membuka gambar menggunakan modul Pillow dan menggunakan pytesseract untuk mengubahnya menjadi teks.
* Menetapkan hasil teks ke label result\_label.

1. Inisialisasi Tkinter

app = tk.Tk()

app.title("Mengubah Gambar menjadi Teks")

Membuat objek aplikasi Tkinter (app) dan menetapkan judul jendela aplikasi.

1. Membuat tombol Play Program

button = ttk.Button(app, text="Play Program", command=lakukan\_ocr)

button.pack(padx=20, pady=20)

1. Membuat Label untuk Menampilkan Hasil OCR:

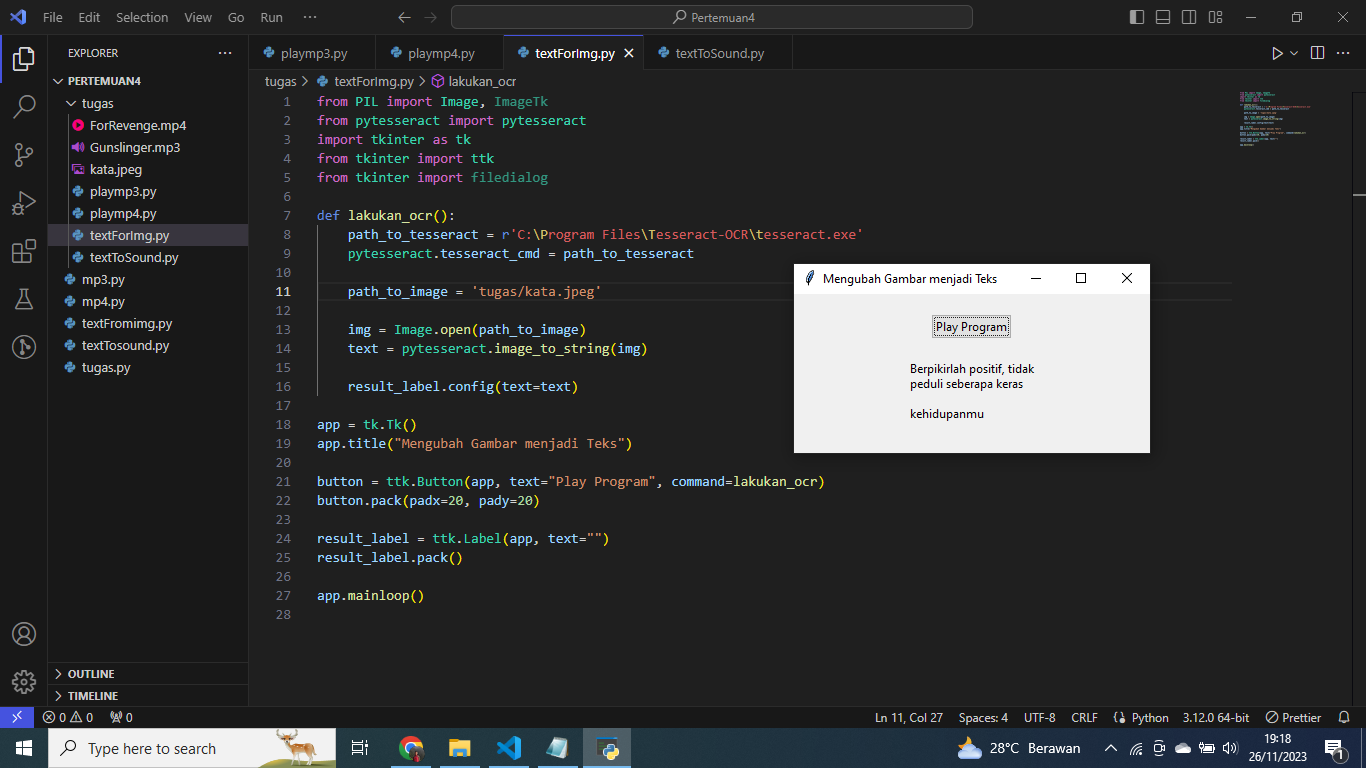
result\_label = ttk.Label(app, text="Sugianto Tegar Samudra 220511162")

result\_label.pack()

1. Menjalankan Aplikasi Tkinter:

app.mainloop()

1. Output



Hasil running file menampilkan aplikasi dengan judul “Mengubah Gambar menjadi Teks” serta terdapat tombol “Play Program” untuk memulai program sehingga program yang dihasilkan mengeluarkan semua tulisan yang terdapat pada gambar.

* Aplikasi mengkonversi Teks menjadi Suara

1. Install library gTTS (Google Text-to-Speech), menggunakan pip dengan buka terminal dan ketik perintah berikut:

pip install gtts

Perintah di atas akan menginstal library gTTS yang digunakan untuk mengonversi teks menjadi suara.

1. Import Modul

from gtts import gTTS

from playsound import playsound

import tkinter as tk

from tkinter import ttk

Kode ini mengimpor modul gTTS untuk Text-to-Speech dari Google dan playsound untuk memutar file audio.

1. Definisi Fungsi teks\_ke\_suara:

def teks\_ke\_suara():

    mytext = 'Sugianto Tegar Samudra'

    language = 'id'

    myobj = gTTS(text=mytext, lang=language, slow=False)

    myobj.save("sugianto.mp3")

    playsound("sugianto.mp3", True)

* Fungsi ini dijalankan saat tombol "Teks ke Suara dan Putar Audio" ditekan.
* Mendefinisikan teks yang akan diubah menjadi suara (mytext), bahasa (language), dan objek gTTS dengan teks dan bahasa tersebut.
* Menyimpan file suara hasil konversi ke "sugianto.mp3".
* Memutar file audio menggunakan playsound.

1. Inisialisasi Tkinter:

app = tk.Tk()

app.title("Text-to-Speech dan Pemutar Audio")

1. Membuat Tombol "Teks ke Suara dan Putar Audio":

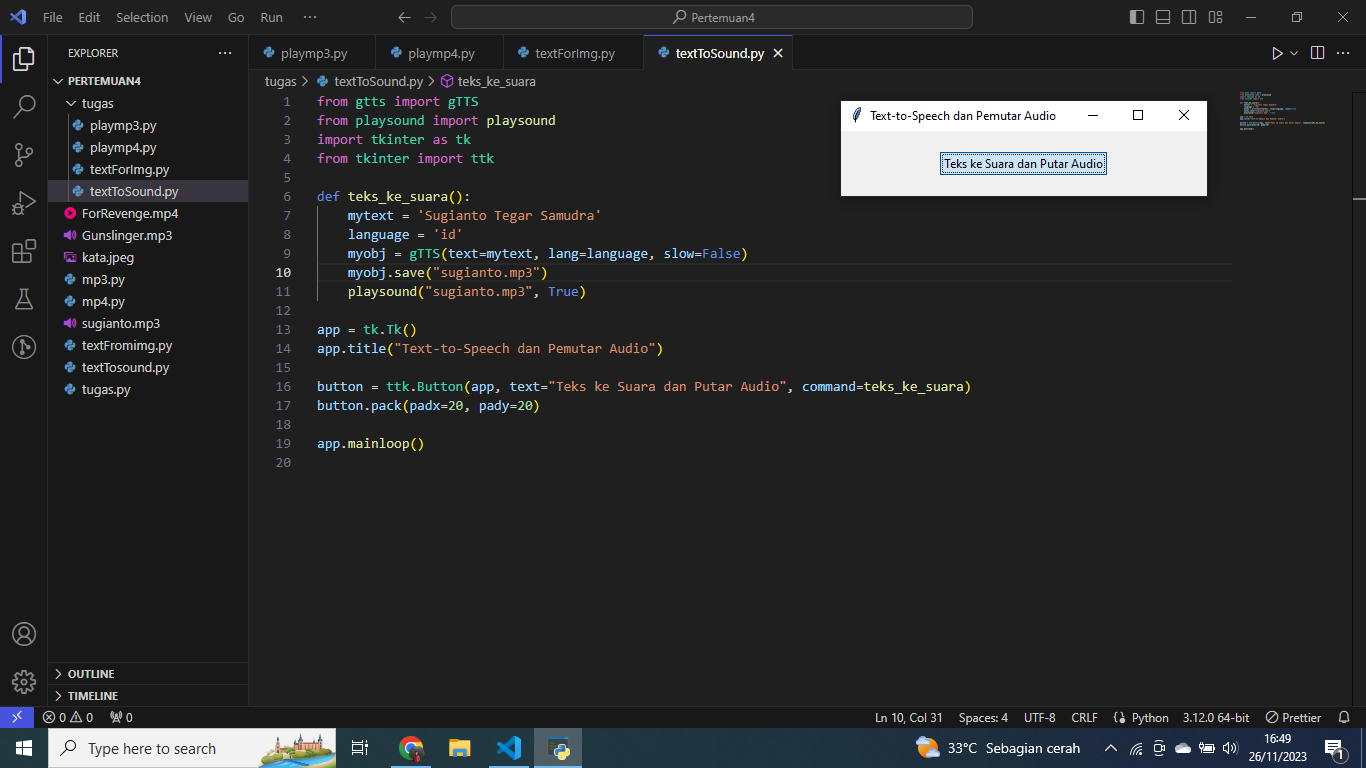
button = ttk.Button(app, text="Teks ke Suara dan Putar Audio", command=teks\_ke\_suara)

button.pack(padx=20, pady=20)

1. Menjalankan Aplikasi Tkinter:

app.mainloop()

1. Output



Pada aplikasi ini hanya menampilkan tombol sehingga Ketika tombol diklik maka otomatis teks yang tertulis dalam code akan bersuara.

# PERTEMUAN 7

**Membuat Aplikasi Konversi Suhu menggunakan Tkinter**

1. Membuat Window Tkinter

Langkah pertama dalam implementasi adalah membuat window Tkinter. Window ini akan menjadi wadah utama untuk elemen-elemen antarmuka pengguna.

# Membuat window

window = tk.Tk()

window.title("Aplikasi Konversi Suhu")

1. Menambahkan Komponen Antarmuka

Tambahkan komponen antarmuka seperti entry untuk suhu awal, dropdown untuk satuan suhu awal dan tujuan, tombol untuk konversi, dan label untuk menampilkan hasil.

#Label Nama

Nama =tk.Label(window, text="Sugianto Tegar Samudra (220511162)")

Nama.grid(row=0, column=0, sticky=W, padx=5, pady=5)

# Membuat label dan entry untuk suhu awal

label\_suhu\_awal = tk.Label(window, text="Suhu Awal:")

label\_suhu\_awal.grid(row=1, column=0, padx=10, pady=10)

entry\_suhu = tk.Entry(window)

entry\_suhu.grid(row=1, column=1, padx=10, pady=10)

# Membuat label dan combobox untuk satuan suhu awal

label\_suhu\_awal = tk.Label(window, text="Satuan Awal:")

label\_suhu\_awal.grid(row=2, column=0, padx=10, pady=10)

satuan\_suhu\_awal = ["Celsius", "Fahrenheit", "Kelvin"]

combo\_suhu\_awal = tk.StringVar()

combo\_suhu\_awal.set(satuan\_suhu\_awal[0])

dropdown\_suhu\_awal = tk.OptionMenu(window, combo\_suhu\_awal, \*satuan\_suhu\_awal)

dropdown\_suhu\_awal.grid(row=2, column=1, padx=10, pady=10)

# Membuat label dan combobox untuk satuan suhu tujuan

label\_suhu\_tujuan = tk.Label(window, text="Satuan Tujuan:")

label\_suhu\_tujuan.grid(row=3, column=0, padx=10, pady=10)

satuan\_suhu\_tujuan = ["Celsius", "Fahrenheit", "Kelvin"]

combo\_suhu\_tujuan = tk.StringVar()

combo\_suhu\_tujuan.set(satuan\_suhu\_tujuan[1])

dropdown\_suhu\_tujuan = tk.OptionMenu(window, combo\_suhu\_tujuan, \*satuan\_suhu\_tujuan)

dropdown\_suhu\_tujuan.grid(row=3, column=1, padx=10, pady=10)

# Membuat tombol konversi

tombol\_konversi = tk.Button(window, text="Konversi", command=konversi\_suhu)

tombol\_konversi.grid(row=4, column=0, columnspan=2, pady=10)

# Membuat label hasil konversi

label\_hasil = tk.Label(window, text="Hasil Konversi:")

label\_hasil.grid(row=5, column=0, columnspan=2, pady=10)

1. Logika Konversi Suhu

Implementasikan logika konversi suhu di dalam fungsi konversi\_suhu().

def konversi\_suhu():

    try:

        suhu\_input = float(entry\_suhu.get())

        satuan\_awal = combo\_suhu\_awal.get()

        satuan\_tujuan = combo\_suhu\_tujuan.get()

        if satuan\_awal == "Celsius":

            if satuan\_tujuan == "Fahrenheit":

                suhu\_output = (suhu\_input \* 9/5) + 32

            elif satuan\_tujuan == "Kelvin":

                suhu\_output = suhu\_input + 273.15

            else:

                suhu\_output = suhu\_input

        elif satuan\_awal == "Fahrenheit":

            if satuan\_tujuan == "Celsius":

                suhu\_output = (suhu\_input - 32) \* 5/9

            elif satuan\_tujuan == "Kelvin":

                suhu\_output = (suhu\_input - 32) \* 5/9 + 273.15

            else:

                suhu\_output = suhu\_input

        elif satuan\_awal == "Kelvin":

            if satuan\_tujuan == "Celsius":

                suhu\_output = suhu\_input - 273.15

            elif satuan\_tujuan == "Fahrenheit":

                suhu\_output = (suhu\_input - 273.15) \* 9/5 + 32

            else:

                suhu\_output = suhu\_input

        else:

            suhu\_output = suhu\_input

        label\_hasil.config(text=f"Hasil Konversi: {suhu\_output:.2f} {satuan\_tujuan}")

    except ValueError:

        label\_hasil.config(text="Masukkan suhu dengan benar!")

1. Menambahkan Fitur Tambahan

Pertimbangkan untuk menambahkan fitur tambahan agar aplikasi lebih fungsional. Misalnya, Anda dapat menambahkan tombol reset untuk mengosongkan input dan hasil konversi.

def reset():

    entry\_suhu.delete(0, tk.END)

    combo\_suhu\_awal.set(satuan\_suhu\_awal[0])

    combo\_suhu\_tujuan.set(satuan\_suhu\_tujuan[1])

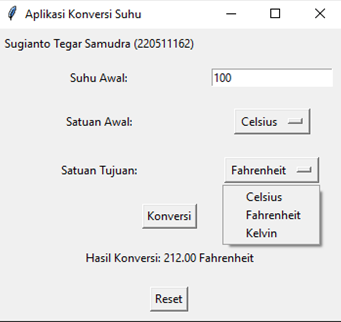
    label\_hasil.config(text="Hasil Konversi:")

# Menambahkan tombol reset

tombol\_reset = tk.Button(window, text="Reset", command=reset)

tombol\_reset.grid(row=5, column=0, columnspan=2, pady=10)

1. Output



Output yang dihasilkan Ketika file dirunning akan menampilkan aplikasi engan judul “Aplikasi Konversi Suhu”, dengan user menginput Suhu Awal kemudian pilih Satuan Awal dan Satuan Tujuan lalu klik Konversi maka akan muncul hasil dari Satuan Tujuannya.

# PERTEMUAN 8 : UTS

# PERTEMUAN 9

**Object dan Class**

Python telah menjadi bahasa berorientasi objek sejak bahasa Python sendiri dibuat. Untuk membuat dan menggunakan kelas dan objek pada Python benar-benar mudah.

Jika Anda tidak memiliki pengalaman sebelumnya dengan pemrograman berorientasi objek (OOP), Anda mempelajarinya terlebih dahulu agar Anda dapat memahami konsep dasarnya.

Jika memang sudah mengerti konsep dasar OOP berikut ini adalah pengenalan dari Object-Oriented Programming (OOP) untuk membantu Anda.

|  |  |
| --- | --- |
| ISTILAH | PENJELASAN |
| Class | Prototipe yang ditentukan pengguna untuk objek yang mendefinisikan seperangkat atribut yang menjadi ciri objek kelas apa pun. Atribut adalah data anggota (variabel kelas dan variabel contoh) dan metode, diakses melalui notasi titik. |
| Class Variable | Sebuah variabel yang dibagi oleh semua contoh kelas. Variabel kelas didefinisikan dalam kelas tapi di luar metode kelas manapun. Variabel kelas tidak digunakan sesering variabel contoh. |
| Data Member | Variabel kelas atau variabel contoh yang menyimpan data yang terkait dengan kelas dan objeknya. |
| Function Overloading | Penugasan lebih dari satu perilaku ke fungsi tertentu. Operasi yang dilakukan bervariasi menurut jenis objek atau argumen yang terlibat. |
| Instance Variable | Variabel yang didefinisikan di dalam sebuah metode dan hanya dimiliki oleh instance kelas saat ini. |
| Inheritance | Pengalihan karakteristik kelas ke kelas lain yang berasal darinya. |
| Instance | Objek individu dari kelas tertentu. Obyek obj yang termasuk dalam Lingkaran kelas, misalnya, adalah turunan dari Lingkaran kelas. |
| Instantiation | Penciptaan sebuah instance dari sebuah kelas. |
| Method | Jenis fungsi khusus yang didefinisikan dalam definisi kelas. |
| Object | Contoh unik dari struktur data yang didefinisikan oleh kelasnya. Objek terdiri dari kedua anggota data (variabel kelas dan variabel contoh) dan metode. |
| Operator Overloading | Penugasan lebih dari satu fungsi ke operator tertentu. |

1. Membuat Class Python

Statement class digunakan untuk membuat definisi kelas baru. Nama kelas segera mengikuti kelas kata kunci diikuti oleh titik dua sebagai berikut.

class AplikasiKonversiSuhu:

    def \_\_init\_\_(self, master):

        # ...

        # (Constructor) Inisialisasi objek AplikasiKonversiSuhu dengan master (Tk root window)

    def konversi\_suhu(self):

        # ...

        # Metode untuk melakukan konversi suhu

    def reset(self):

        # ...

        # Metode untuk mereset nilai pada entry, combo box, dan label hasil

1. Membuat instance objects

Untuk membuat instances kelas, Anda memanggil class menggunakan nama class dan meneruskan argumen apa pun yang metode init terima.

root = tk.Tk()  # Membuat objek Tkinter root window

app = AplikasiKonversiSuhu(root)  # Membuat objek AplikasiKonversiSuhu dengan root sebagai master

1. Mengakses atribut

Anda mengakses atribut objek menggunakan dot operator dengan objek. Variabel kelas akan diakses dengan menggunakan nama kelas sebagai berikut :

# Dalam constructor (\_\_init\_\_)

self.nama\_label  # Atribut Label Nama

self.entry\_suhu  # Atribut Entry untuk suhu

self.combo\_suhu\_awal  # Atribut StringVar untuk satuan suhu awal

self.dropdown\_suhu\_awal  # Atribut OptionMenu untuk combo box satuan suhu awal

self.combo\_suhu\_tujuan  # Atribut StringVar untuk satuan suhu tujuan

self.dropdown\_suhu\_tujuan  # Atribut OptionMenu untuk combo box satuan suhu tujuan

self.tombol\_konversi  # Atribut Button untuk tombol konversi

self.label\_hasil  # Atribut Label untuk menampilkan hasil konversi

self.tombol\_reset  # Atribut Button untuk tombol reset

# Dalam metode konversi\_suhu

self.entry\_suhu.get()  # Mengakses nilai yang dimasukkan ke dalam entry\_suhu

self.combo\_suhu\_awal.get()  # Mengakses nilai yang dipilih dalam combo\_suhu\_awal

self.combo\_suhu\_tujuan.get()  # Mengakses nilai yang dipilih dalam combo\_suhu\_tujuan

self.label\_hasil.config(text=f"Hasil Konversi: {suhu\_output:.2f} {satuan\_tujuan}")  # Mengakses dan mengubah teks pada label\_hasil

# Dalam metode reset

self.entry\_suhu.delete(0, tk.END)  # Menghapus nilai dalam entry\_suhu

self.combo\_suhu\_awal.set(self.satuan\_suhu\_awal[0])  # Mengatur nilai default combo\_suhu\_awal

self.combo\_suhu\_tujuan.set(self.satuan\_suhu\_tujuan[1])  # Mengatur nilai default combo\_suhu\_tujuan

self.label\_hasil.config(text="Hasil Konversi:")  # Mengatur teks pada label\_hasil

* contoh lengkapnya,silahkan lihat kode dibawah ini

import tkinter as tk

from tkinter import Label, Entry, Button, W, OptionMenu, StringVar

class AplikasiKonversiSuhu:

    def \_\_init\_\_(self, master):

        self.master = master

        master.title("Aplikasi Konversi Suhu")

        # Label Nama

        self.nama\_label = Label(master, text="Daffa Ramadani Putra (220511150)")

        self.nama\_label.grid(row=0, column=0, sticky=W, padx=5, pady=5)

        # Entry untuk suhu

        self.label\_suhu\_awal = Label(master, text="Suhu Awal:")

        self.label\_suhu\_awal.grid(row=1, column=0, padx=10, pady=10)

        self.entry\_suhu = Entry(master)

        self.entry\_suhu.grid(row=1, column=1, padx=10, pady=10)

        # Combobox untuk satuan suhu awal

        self.label\_suhu\_awal = Label(master, text="Satuan Awal:")

        self.label\_suhu\_awal.grid(row=2, column=0, padx=10, pady=10)

        self.satuan\_suhu\_awal = ["Celsius", "Fahrenheit", "Kelvin", "Reamur"]

        self.combo\_suhu\_awal = StringVar()

        self.combo\_suhu\_awal.set(self.satuan\_suhu\_awal[0])

        self.dropdown\_suhu\_awal = OptionMenu(master, self.combo\_suhu\_awal, \*self.satuan\_suhu\_awal)

        self.dropdown\_suhu\_awal.grid(row=2, column=1, padx=10, pady=10)

        # Combobox untuk satuan suhu tujuan

        self.label\_suhu\_tujuan = Label(master, text="Satuan Tujuan:")

        self.label\_suhu\_tujuan.grid(row=3, column=0, padx=10, pady=10)

        self.satuan\_suhu\_tujuan = ["Celsius", "Fahrenheit", "Kelvin", "Reamur"]

        self.combo\_suhu\_tujuan = StringVar()

        self.combo\_suhu\_tujuan.set(self.satuan\_suhu\_tujuan[1])

        self.dropdown\_suhu\_tujuan = OptionMenu(master, self.combo\_suhu\_tujuan, \*self.satuan\_suhu\_tujuan)

        self.dropdown\_suhu\_tujuan.grid(row=3, column=1, padx=10, pady=10)

        # Tombol Konversi

        self.tombol\_konversi = Button(master, text="Konversi", command=self.konversi\_suhu)

        self.tombol\_konversi.grid(row=4, column=0, columnspan=2, pady=10)

        # Label hasil

        self.label\_hasil = Label(master, text="Hasil Konversi:")

        self.label\_hasil.grid(row=5, column=0, columnspan=2, pady=10)

        # Tombol Reset

        self.tombol\_reset = Button(master, text="Reset", command=self.reset)

        self.tombol\_reset.grid(row=6, column=0, columnspan=2, pady=10)

    def konversi\_suhu(self):

        try:

            suhu\_input = float(self.entry\_suhu.get())

            satuan\_awal = self.combo\_suhu\_awal.get()

            satuan\_tujuan = self.combo\_suhu\_tujuan.get()

            if satuan\_awal == "Celsius":

                if satuan\_tujuan == "Fahrenheit":

                    suhu\_output = (suhu\_input \* 9/5) + 32

                elif satuan\_tujuan == "Kelvin":

                    suhu\_output = suhu\_input + 273.15

                elif satuan\_tujuan == "Reamur":

                    suhu\_output = suhu\_input \* 4/5

                else:

                    suhu\_output = suhu\_input

            elif satuan\_awal == "Fahrenheit":

                if satuan\_tujuan == "Celsius":

                    suhu\_output = (suhu\_input - 32) \* 5/9

                elif satuan\_tujuan == "Kelvin":

                    suhu\_output = (suhu\_input - 32) \* 5/9 + 273.15

                elif satuan\_tujuan == "Reamur":

                    suhu\_output = (suhu\_input - 32) \* 4/9

                else:

                    suhu\_output = suhu\_input

            elif satuan\_awal == "Kelvin":

                if satuan\_tujuan == "Celsius":

                    suhu\_output = suhu\_input - 273.15

                elif satuan\_tujuan == "Fahrenheit":

                    suhu\_output = (suhu\_input - 273.15) \* 9/5 + 32

                elif satuan\_tujuan == "Reamur":

                    suhu\_output = (suhu\_input - 273.15) \* 4/5

                else:

                    suhu\_output = suhu\_input

            elif satuan\_awal == "Reamur":

                if satuan\_tujuan == "Celsius":

                    suhu\_output = suhu\_input \* 5/4

                elif satuan\_tujuan == "Fahrenheit":

                    suhu\_output = suhu\_input \* 9/4 + 32

                elif satuan\_tujuan == "Kelvin":

                    suhu\_output = suhu\_input \* 5/4 + 273.15

                else:

                    suhu\_output = suhu\_input

            else:

                suhu\_output = suhu\_input

            self.label\_hasil.config(text=f"Hasil Konversi: {suhu\_output:.2f} {satuan\_tujuan}")

        except ValueError:

            self.label\_hasil.config(text="Masukkan suhu dengan benar!")

    def reset(self):

        self.entry\_suhu.delete(0, tk.END)

        self.combo\_suhu\_awal.set(self.satuan\_suhu\_awal[0])

        self.combo\_suhu\_tujuan.set(self.satuan\_suhu\_tujuan[1])

        self.label\_hasil.config(text="Hasil Konversi:")

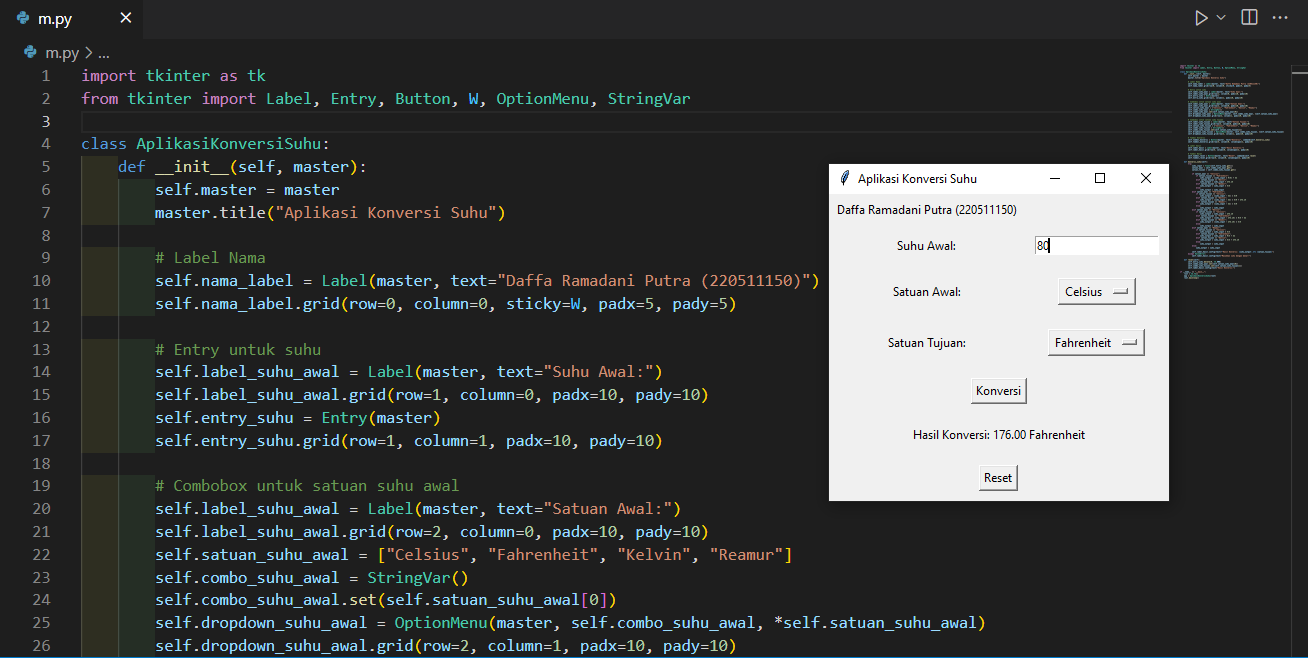
if \_\_name\_\_ == "\_\_main\_\_":

    root = tk.Tk()

    app = AplikasiKonversiSuhu(root)

    root.mainloop()

output :



# PERTEMUAN 10

**Pengenalan Data Base MYSQL**

Apa itu PyMySQL ?

PyMySQL adalah sebuah antarmuka untuk menghubungkan ke server database MySQL dari Python. Ini mengimplementasikan API Database Python v2.0 dan berisi perpustakaan klien MySQL murni-Python. Tujuan PyMySQL adalah penggantian drop-in untuk MySQLdb.

Cara Instal PyMySQL :

Sebelum melanjutka, pastikan Anda telah menginstal PyMySQL di komputer Anda. Cukup ketik berikut ini di skrip Python Anda dan jalankan.

“import pymysql.cursors”

Jika menghasilkan hasil berikut, berarti modul MySQLdb tidak terpasang:

Traceback (most recent call last): File "test.py", line 3, in Import PyMySQL ImportError: No module named PyMySQL

Untuk menginstal modul PyMySQL silahkan gunakan command/perintah berikut di command prompt:

“python -m pip install PyMySQL”

Database Connection :

Sebelum terhubung ke database MySQL, pastikan beberapa hal dibawah ini :

Anda telah membuat database TESTDB.

Anda telah membuat tabel EMPLOYEE di TESTDB.

Tabel ini memiliki bidang FIRST\_NAME, LAST\_NAME, AGE, SEX, dan INCOME.

User ID "testuser" dan password "test123" diatur untuk mengakses TESTDB.

Python modul PyMySQL terinstal dengan benar pada mesin Anda.

Anda telah melalui tutorial MySQL untuk memahami Dasar-Dasar MySQL

Berikut ini adalah contoh koneksi dengan database MySQL "TESTDB"

import pymysql.cursors

# Open database connection

db = pymysql.connect("localhost","testuser","test123","TESTDB" )

# prepare a cursor object using cursor() method

cursor = db.cursor()

# execute SQL query using execute() method.

cursor.execute("SELECT VERSION()")

# Fetch a single row using fetchone() method.

data = cursor.fetchone()

print ("Database version : %s " % data)

# disconnect from server

db.close()

Membuat table database :

import pymysql.cursors

# Open database connection

db = pymysql.connect("localhost","testuser","test123","TESTDB" )

# prepare a cursor object using cursor() method

cursor = db.cursor()

# Drop table if it already exist using execute() method.

cursor.execute("DROP TABLE IF EXISTS EMPLOYEE")

# Create table as per requirement

sql = """CREATE TABLE EMPLOYEE (

FIRST\_NAME  CHAR(20) NOT NULL,

LAST\_NAME  CHAR(20),

AGE INT,

SEX CHAR(1),

INCOME FLOAT )"""

cursor.execute(sql)

# disconnect from server

db.close()

* Operasi insert

Contoh berikut, mengeksekusi pernyataan SQL INSERT untuk membuat catatan di tabel EMPLOYEE

import pymysql.cursors

# Open database connection

db = pymysql.connect("localhost","testuser","test123","TESTDB" )

# prepare a cursor object using cursor() method

cursor = db.cursor()

# Prepare SQL query to INSERT a record into the database.

sql = """INSERT INTO EMPLOYEE(FIRST\_NAME,

LAST\_NAME, AGE, SEX, INCOME)

VALUES ('Mac', 'Mohan', 20, 'M', 2000)"""

try:

# Execute the SQL command

cursor.execute(sql)

# Commit your changes in the database

db.commit()

except:

# Rollback in case there is any error

db.rollback()

# disconnect from server

db.close()

Contoh di atas bisa dituliskan sebagai berikut untuk membuat query SQL secara dinamis

import pymysql.cursors

# Open database connection

db = pymysql.connect("localhost","testuser","test123","TESTDB" )

# prepare a cursor object using cursor() method

cursor = db.cursor()

# Prepare SQL query to INSERT a record into the database.

sql = "INSERT INTO EMPLOYEE(FIRST\_NAME, \

   LAST\_NAME, AGE, SEX, INCOME) \

   VALUES ('%s', '%s', '%d', '%c', '%d' )" % \

   ('Mac', 'Mohan', 20, 'M', 2000)

try:

   # Execute the SQL command

   cursor.execute(sql)

   # Commit your changes in the database

   db.commit()

except:

   # Rollback in case there is any error

   db.rollback()

# disconnect from server

db.close()

* Read Operation

READ Operation pada database apapun berarti mengambil beberapa informasi berguna dari database.

Setelah koneksi database terbentuk, Anda siap untuk membuat query ke dalam database ini. Anda bisa menggunakan metode fetchone() untuk mengambil satu record atau fetchall() metode untuk mengambil beberapa nilai dari tabel database.

Fetchone () - Ini mengambil baris berikut dari kumpulan hasil query. Set hasil adalah objek yang dikembalikan saat objek kursor digunakan untuk query tabel.

Fetchall () - Ini menjemput semua baris dalam kumpulan hasil. Jika beberapa baris telah diekstraksi dari himpunan hasil, maka akan diambil baris yang tersisa dari kumpulan hasil.

Rowcount - Ini adalah atribut read-only dan mengembalikan jumlah baris yang dipengaruhi oleh metode execute ().

import pymysql.cursors

# Open database connection

db = pymysql.connect("localhost","testuser","test123","TESTDB" )

# prepare a cursor object using cursor() method

cursor = db.cursor()

# Prepare SQL query to INSERT a record into the database.

sql = "SELECT \* FROM EMPLOYEE \

       WHERE INCOME > '%d'" % (1000)

try:

   # Execute the SQL command

   cursor.execute(sql)

   # Fetch all the rows in a list of lists.

   results = cursor.fetchall()

   for row in results:

      fname = row[0]

      lname = row[1]

      age = row[2]

      sex = row[3]

      income = row[4]

      # Now print fetched result

      print ("fname = %s,lname = %s,age = %d,sex = %s,income = %d" % \

             (fname, lname, age, sex, income ))

except:

   print ("Error: unable to fetch data")

# menutup koneksi ke server

db.close()

Setelah Anda eksekusi kode diatas, akan muncul hasil seperti dibawah ini :

fname = Mac, lname = Mohan, age = 20, sex = M, income = 2000

* Update Operation

Operasi UPDATE pada database apapun berarti mengupdate satu atau lebih catatan, yang sudah tersedia di database. Prosedur berikut memperbarui semua catatan yang memiliki SEX sebagai 'M'. Di sini, kita meningkatkan UMUR semua laki-laki satu tahun.

import pymysql.cursors

# Open database connection

db = pymysql.connect("localhost","testuser","test123","TESTDB" )

# prepare a cursor object using cursor() method

cursor = db.cursor()

# Prepare SQL query to UPDATE required records

sql = "UPDATE EMPLOYEE SET AGE = AGE + 1

                          WHERE SEX = '%c'" % ('M')

try:

   # Execute the SQL command

   cursor.execute(sql)

   # Commit your changes in the database

   db.commit()

except:

   # Rollback in case there is any error

   db.rollback()

# disconnect from server

db.close()

* Delete Operation

Operasi DELETE diperlukan bila Anda ingin menghapus beberapa catatan dari database Anda. Berikut ini adalah prosedur untuk menghapus semua catatan dari EMPLOYEE dimana AGE lebih dari 20

import pymysql.cursors

# Open database connection

db = pymysql.connect("localhost","testuser","test123","TESTDB" )

# prepare a cursor object using cursor() method

cursor = db.cursor()

# Prepare SQL query to DELETE required records

sql = "DELETE FROM EMPLOYEE WHERE AGE > '%d'" % (20)

try:

   # Execute the SQL command

   cursor.execute(sql)

   # Commit your changes in the database

   db.commit()

except:

   # Rollback in case there is any error

   db.rollback()

# disconnect from server

db.close()

# PERTEMUAN 11

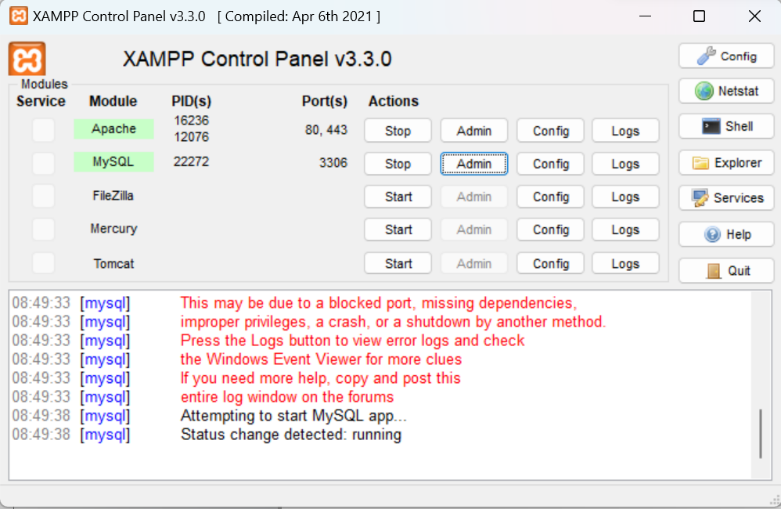
**Python database application**

Install xmapp :

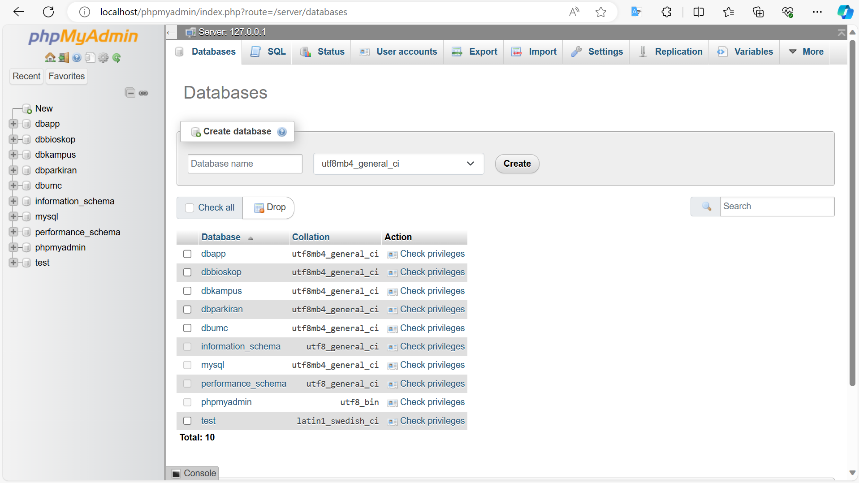
1. Download xmapp terlebih dahulu di sini [Download XAMPP (apachefriends.org)](https://www.apachefriends.org/download.html)
2. Pilih Versi di halaman unduh, nanti akan melihat pilihan versi XAMPP untuk berbagai sistem operasi. Pilih versi yang sesuai dengan sistem operasi Anda (Windows, macOS, atau Linux).
3. Pilih Installer, klik pada opsi "XAMPP for [nama sistem operasi]". Misalnya, "XAMPP for Windows".
4. Pilih bit yang sesuai dengan sistem operasi Anda (32-bit atau 64-bit).
5. Klik tombol "Download" untuk mulai mengunduh installer XAMPP.
6. Jika sudah di download lalu install aplikasi xmapp tersebut.

Menjalankan xmapp dan PhpMyAdmin :

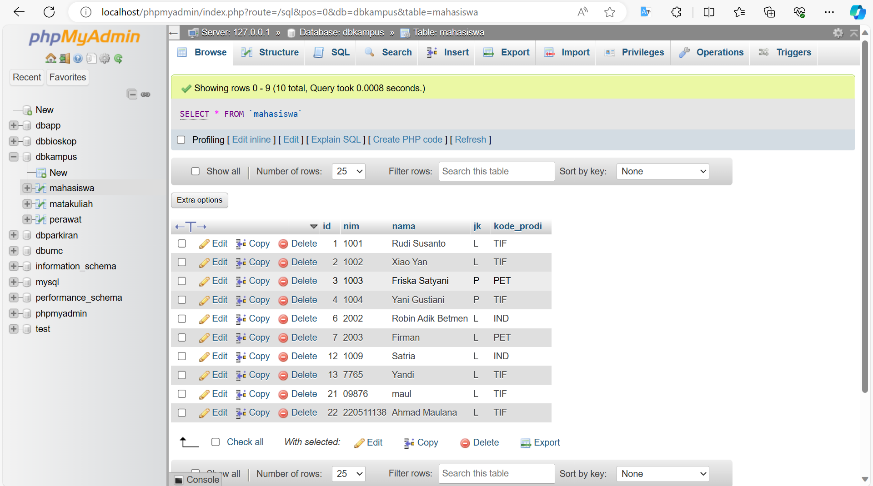
1. Buka aplikasi xmapp yang sudah di install
2. Klik “start” pada Apache dan MySQL. Pastikan berwarna hijau untuk Apache dan My SQL nya.



1. Klik “Admin” pada MySQL, jika sudah nanti akan masuk ke PhpMyAdmin di browser anda



1. Buat data base baru dengan nama “dbmahasiswa”, lalu import tabel mahasiswa.sql di menu import di PhpMyAdmin. Untuk file mahasiswa.sql bisa ambil di sini [AhmadMaulana220511138/PBO\_Semester3\_2023 (github.com)](https://github.com/AhmadMaulana220511138/PBO_Semester3_2023) pilih pertemuan 11, lalu pilih folder mahasiswa di dalam foldernya ada file mahasiswa.sql.
2. Jika sudah tampilanya akan seperti ini



Buka Visual Studio Code :

1. Buka Visual Studio Code, buat folder baru contoh : mahasiswa
2. Masuk ke github ini [AhmadMaulana220511138/PBO\_Semester3\_2023 (github.com)](https://github.com/AhmadMaulana220511138/PBO_Semester3_2023) pilih pertemuan 11, masuk ke folder mahasiswa lalu copy file yang Bernama db.py, frm.py dan mahasiswa.py. Bisa juga copy dari code dibawah ini.
3. Buat nama file nya sama persis seperti di atas , berikut adalan code untuk db.py, frm.py, mahasiswa.py
4. db.py

import mysql.connector as mc

class DBConnection:

    def \_\_init\_\_(*self*):

*self*.host = "localhost"

*self*.port = 3306

*self*.name = "dbkampus"

*self*.user = "root"

*self*.password = ""

*self*.conn = None

*self*.cursor = None

*self*.result = None

*self*.connected = False

*self*.affected = 0

*self*.connect()

    @property

    def connection\_status(*self*):

        return *self*.connected

    def connect(*self*):

        try:

*self*.conn = mc.connect(*host* = *self*.host,

*port* = *self*.port,

*database* = *self*.name,

*user* = *self*.user,

*password* = *self*.password)

*self*.connected = True

*self*.cursor=*self*.conn.cursor()

        except mc.Error as e:

*self*.connected = False

        return *self*.conn

    def disconnect(*self*):

        if(*self*.connected==True):

*self*.conn.close

        else:

*self*.conn = None

    def findOne(*self*, *sql*):

*self*.connect()

*self*.cursor.execute(*sql*)

*self*.result = *self*.cursor.fetchone()

        return *self*.result

    def findAll(*self*, *sql*):

*self*.connect()

*self*.result = *self*.cursor.execute(*sql*)

*self*.result = *self*.cursor.fetchall()

        return *self*.result

    def insert(*self*, *sql*):

*self*.connect()

*self*.cursor.execute(*sql*)

*self*.conn.commit()

*self*.affected = *self*.cursor.rowcount

        return *self*.affected

    def update(*self*, *sql*, *val*):

*self*.connect()

*self*.cursor.execute(*sql*, *val*)

*self*.conn.commit()

*self*.affected = *self*.cursor.rowcount

        return *self*.affected

    def delete(*self*, *sql*):

*self*.connect()

*self*.cursor.execute(*sql*)

*self*.conn.commit()

*self*.affected = *self*.cursor.rowcount

        return *self*.affected

    def show(*self*, *sql*):

*self*.connect()

*self*.cursor.execute(*sql*)

*self*.result = *self*.cursor.fetchone()

        return *self*.result

    @property

    def info(*self*):

        if(*self*.connected==True):

            return "Server is running on " + *self*.host + ' using port ' + str(*self*.port)

        else:

            return "Server is offline."

*# test koneksi databse*

A = DBConnection()

B = A.info

print(B)

1. frm.py

import tkinter as tk

from tkinter import Frame,Label,Entry,Button,Radiobutton,ttk,VERTICAL,YES,BOTH,END,Tk,W,StringVar,messagebox

from mahasiswa import Mahasiswa

class FormMahasiswa:

    def \_\_init\_\_(*self*, *parent*, *title*):

*self*.parent = *parent*

*self*.parent.geometry("450x450")

*self*.parent.title(*title*)

*self*.parent.protocol("WM\_DELETE\_WINDOW", *self*.onKeluar)

*self*.ditemukan = None

*self*.aturKomponen()

*self*.onReload()

    def aturKomponen(*self*):

        mainFrame = Frame(*self*.parent, *bd*=10)

        mainFrame.pack(*fill*=BOTH, *expand*=YES)

*# Label*

        Label(mainFrame, *text*='NIM:').grid(*row*=0, *column*=0, *sticky*=W, *padx*=5, *pady*=5)

*self*.txtNIM = Entry(mainFrame)

*self*.txtNIM.grid(*row*=0, *column*=1, *padx*=5, *pady*=5)

*self*.txtNIM.bind("<Return>",*self*.onCari) *# menambahkan event Enter key*

        Label(mainFrame, *text*='Nama:').grid(*row*=1, *column*=0, *sticky*=W, *padx*=5, *pady*=5)

*self*.txtNama = Entry(mainFrame)

*self*.txtNama.grid(*row*=1, *column*=1, *padx*=5, *pady*=5)

        Label(mainFrame, *text*='Jenis Kelamin:').grid(*row*=2, *column*=0, *sticky*=W, *padx*=5, *pady*=5)

*self*.txtJK = StringVar()

*self*.L = Radiobutton(mainFrame, *text*='Laki-laki', *value*='L', *variable*=*self*.txtJK)

*self*.L.grid(*row*=2, *column*=1, *padx*=5, *pady*=5, *sticky*=W)

*self*.L.select() *# set pilihan yg pertama*

*self*.P = Radiobutton(mainFrame, *text*='Perempuan', *value*='P', *variable*=*self*.txtJK)

*self*.P.grid(*row*=3, *column*=1, *padx*=5, *pady*=5, *sticky*=W)

        Label(mainFrame, *text*='Kode Prodi:').grid(*row*=4, *column*=0, *sticky*=W, *padx*=5, *pady*=5)

*self*.txtKodeProdi = StringVar()

        Cbo = ttk.Combobox(mainFrame, *width* = 27, *textvariable* = *self*.txtKodeProdi)

        Cbo.grid(*row*=4, *column*=1, *padx*=5, *pady*=5)

        Cbo['values'] = ('TIF','IND','PET')

        Cbo.current()

*# Button*

*self*.btnSimpan = Button(mainFrame, *text*='Simpan', *command*=*self*.onSimpan, *width*=10)

*self*.btnSimpan.grid(*row*=0, *column*=3, *padx*=5, *pady*=5)

*self*.btnClear = Button(mainFrame, *text*='Clear', *command*=*self*.onClear, *width*=10)

*self*.btnClear.grid(*row*=1, *column*=3, *padx*=5, *pady*=5)

*self*.btnHapus = Button(mainFrame, *text*='Hapus', *command*=*self*.onDelete, *width*=10)

*self*.btnHapus.grid(*row*=2, *column*=3, *padx*=5, *pady*=5)

*# define columns*

        columns = ('idmhs', 'nim', 'nama','jk','kode\_prodi')

*self*.tree = ttk.Treeview(mainFrame, *columns*=columns, *show*='headings')

*# define headings*

*self*.tree.heading('idmhs', *text*='ID')

*self*.tree.column('idmhs', *width*="30")

*self*.tree.heading('nim', *text*='NIM')

*self*.tree.column('nim', *width*="60")

*self*.tree.heading('nama', *text*='Nama')

*self*.tree.column('nama', *width*="200")

*self*.tree.heading('jk', *text*='JK')

*self*.tree.column('jk', *width*="30")

*self*.tree.heading('kode\_prodi', *text*='Kode Prodi')

*self*.tree.column('kode\_prodi', *width*="100")

*# set tree position*

*self*.tree.place(*x*=0, *y*=200)

*self*.onReload()

    def onClear(*self*, *event*=None):

*self*.txtNIM.delete(0,END)

*self*.txtNIM.insert(END,"")

*self*.txtNama.delete(0,END)

*self*.txtNama.insert(END,"")

*self*.txtKodeProdi.set("")

*self*.btnSimpan.config(*text*="Simpan")

*self*.L.select()

*self*.onReload()

*self*.ditemukan = False

    def onReload(*self*, *event*=None):

*# get data mahasiswa*

        mhs = Mahasiswa()

        result = mhs.getAllData()

        for item in *self*.tree.get\_children():

*self*.tree.delete(item)

        students=[]

        for row\_data in result:

            students.append(row\_data)

        for student in students:

*self*.tree.insert('',END, *values*=student)

    def onCari(*self*, *event*=None):

        nim = *self*.txtNIM.get()

        mhs = Mahasiswa()

        res = mhs.getByNIM(nim)

        rec = mhs.affected

        if(rec>0):

            messagebox.showinfo("showinfo", "Data Ditemukan")

*self*.TampilkanData()

*self*.ditemukan = True

        else:

            messagebox.showwarning("showwarning", "Data Tidak Ditemukan")

*self*.ditemukan = False

*self*.txtNama.focus()

        return res

    def TampilkanData(*self*, *event*=None):

        nim = *self*.txtNIM.get()

        mhs = Mahasiswa()

        res = mhs.getByNIM(nim)

*self*.txtNama.delete(0,END)

*self*.txtNama.insert(END,mhs.nama)

        jk = mhs.jk

        if(jk=="P"):

*self*.P.select()

        else:

*self*.L.select()

*self*.txtKodeProdi.set(mhs.kode\_prodi)

*self*.btnSimpan.config(*text*="Update")

    def onSimpan(*self*, *event*=None):

        nim = *self*.txtNIM.get()

        nama = *self*.txtNama.get()

        jk = *self*.txtJK.get()

        prodi = *self*.txtKodeProdi.get()

        mhs = Mahasiswa()

        mhs.nim = nim

        mhs.nama = nama

        mhs.jk = jk

        mhs.kode\_prodi = prodi

        if(*self*.ditemukan==True):

            res = mhs.updateByNIM(nim)

            ket = 'Diperbarui'

        else:

            res = mhs.simpan()

            ket = 'Disimpan'

        rec = mhs.affected

        if(rec>0):

            messagebox.showinfo("showinfo", "Data Berhasil "+ket)

        else:

            messagebox.showwarning("showwarning", "Data Gagal "+ket)

*self*.onClear()

        return rec

    def onDelete(*self*, *event*=None):

        nim = *self*.txtNIM.get()

        mhs = Mahasiswa()

        mhs.nim = nim

        if(*self*.ditemukan==True):

            res = mhs.deleteByNIM(nim)

            rec = mhs.affected

        else:

            messagebox.showinfo("showinfo", "Data harus ditemukan dulu sebelum dihapus")

            rec = 0

        if(rec>0):

            messagebox.showinfo("showinfo", "Data Berhasil dihapus")

*self*.onClear()

    def onKeluar(*self*, *event*=None):

*# memberikan perintah menutup aplikasi*

*self*.parent.destroy()

if \_\_name\_\_ == '\_\_main\_\_':

    root = tk.Tk()

    aplikasi = FormMahasiswa(root, "Aplikasi Data Mahasiswa")

    root.mainloop()

1. mahasiswa.py

from db import DBConnection as mydb

class Mahasiswa:

    def \_\_init\_\_(*self*):

*self*.\_\_id=None

*self*.\_\_nim=None

*self*.\_\_nama=None

*self*.\_\_jk=None

*self*.\_\_kode\_prodi=None

*self*.conn = None

*self*.affected = None

*self*.result = None

    @property

    def id(*self*):

        return *self*.\_\_id

    @property

    def nim(*self*):

        return *self*.\_\_nim

    @nim.setter

    def nim(*self*, *value*):

*self*.\_\_nim = *value*

    @property

    def nama(*self*):

        return *self*.\_\_nama

    @nama.setter

    def nama(*self*, *value*):

*self*.\_\_nama = *value*

    @property

    def jk(*self*):

        return *self*.\_\_jk

    @jk.setter

    def jk(*self*, *value*):

*self*.\_\_jk = *value*

    @property

    def kode\_prodi(*self*):

        return *self*.\_\_kode\_prodi

    @kode\_prodi.setter

    def kode\_prodi(*self*, *value*):

*self*.\_\_kode\_prodi = *value*

    def simpan(*self*):

*self*.conn = mydb()

        val = (*self*.\_\_nim, *self*.\_\_nama, *self*.\_\_jk, *self*.\_\_kode\_prodi)

        sql="INSERT INTO mahasiswa (nim, nama, jk, kode\_prodi) VALUES " + str(val)

*self*.affected = *self*.conn.insert(sql)

*self*.conn.disconnect

        return *self*.affected

    def update(*self*, *id*):

*self*.conn = mydb()

        val = (*self*.\_\_nim, *self*.\_\_nama, *self*.\_\_jk, *self*.\_\_kode\_prodi, *id*)

        sql="UPDATE mahasiswa SET nim = %s, nama = %s, jk=%s, kode\_prodi=%s WHERE id=%s"

*self*.affected = *self*.conn.update(sql, val)

*self*.conn.disconnect

        return *self*.affected

    def updateByNIM(*self*, *nim*):

*self*.conn = mydb()

        val = (*self*.\_\_nama, *self*.\_\_jk, *self*.\_\_kode\_prodi, *nim*)

        sql="UPDATE mahasiswa SET nama = %s, jk=%s, kode\_prodi=%s WHERE nim=%s"

*self*.affected = *self*.conn.update(sql, val)

*self*.conn.disconnect

        return *self*.affected

    def delete(*self*, *id*):

*self*.conn = mydb()

        sql="DELETE FROM mahasiswa WHERE id='" + str(*id*) + "'"

*self*.affected = *self*.conn.delete(sql)

*self*.conn.disconnect

        return *self*.affected

    def deleteByNIM(*self*, *nim*):

*self*.conn = mydb()

        sql="DELETE FROM mahasiswa WHERE nim='" + str(*nim*) + "'"

*self*.affected = *self*.conn.delete(sql)

*self*.conn.disconnect

        return *self*.affected

    def getByID(*self*, *id*):

*self*.conn = mydb()

        sql="SELECT \* FROM mahasiswa WHERE id='" + str(*id*) + "'"

*self*.result = *self*.conn.findOne(sql)

*self*.\_\_nim = *self*.result[1]

*self*.\_\_nama = *self*.result[2]

*self*.\_\_jk = *self*.result[3]

*self*.\_\_kode\_prodi = *self*.result[4]

*self*.conn.disconnect

        return *self*.result

    def getByNIM(*self*, *nim*):

        a=str(*nim*)

        b=a.strip()

*self*.conn = mydb()

        sql="SELECT \* FROM mahasiswa WHERE nim='" + b + "'"

*self*.result = *self*.conn.findOne(sql)

        if(*self*.result!=None):

*self*.\_\_nim = *self*.result[1]

*self*.\_\_nama = *self*.result[2]

*self*.\_\_jk = *self*.result[3]

*self*.\_\_kode\_prodi = *self*.result[4]

*self*.affected = *self*.conn.cursor.rowcount

        else:

*self*.\_\_nim = ''

*self*.\_\_nama = ''

*self*.\_\_jk = ''

*self*.\_\_kode\_prodi = ''

*self*.affected = 0

*self*.conn.disconnect

        return *self*.result

    def getAllData(*self*):

*self*.conn = mydb()

        sql="SELECT \* FROM mahasiswa"

*self*.result = *self*.conn.findAll(sql)

        return *self*.result

*# Tampilkan Data*

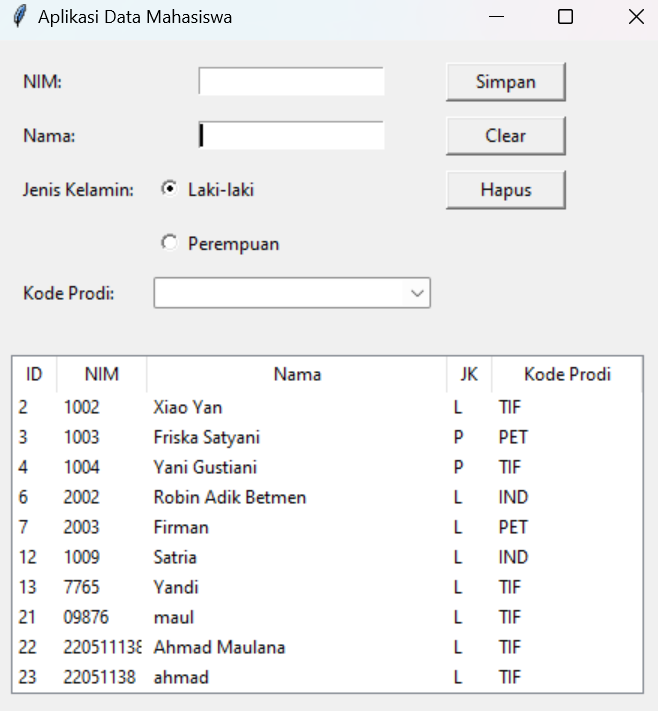
A = Mahasiswa()

B = A.getAllData()

print(B)

1. Output frm.py, setelah di run :

Inputkan data terlebih dahulu, jika sudah klik tombol simpan otomatis data akan tersimpan di data base “dbmahasiswa” dan di tabel mahasiswa di PhpMyAdmin.



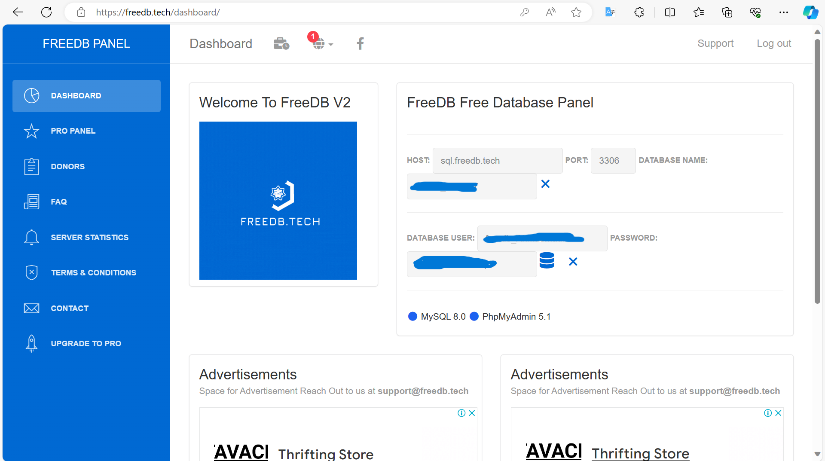
1. Selanjutkan ada file tabel perawat.sql. Masuk ke github ini [AhmadMaulana220511138/PBO\_Semester3\_2023 (github.com)](https://github.com/AhmadMaulana220511138/PBO_Semester3_2023) pilih pertemuan 11, masuk ke folder perawat lalu copy file yang Bernama db.py, frm.py dan Perawat.py. buat nama file nya sama di visual studio code nya. Untuk cara pengerjaaannya sama seperti di atas yang sudah saya jelaskan.

# PERTEMUAN 12

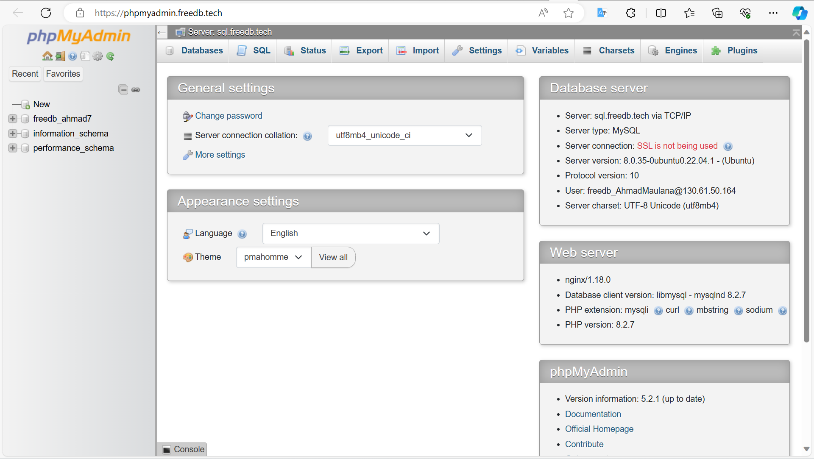
**python database application using remote mysql server on internet**

Daftar akun di Freedb :

1. Buka website ini di browser [Free Remote MySQL Hosting (freedb.tech)](https://freedb.tech/)
2. Klik “Get Started”. Isi bagian from register dengan benar, lalu klik register.
3. Buka isi kotak masuk di email yang pengirim dari FreeDB.Tech. Klik link “Verify Your Account”.
4. Buatlah database baru dengan sesuai kebutuhan kamu, lalu klik Create DB.
5. Buatlah nama pengguna baru dengan sesuai kebutuhan kamu, lalu klik Create User. Jika sudah nanti tampilannya akan seperti ini.

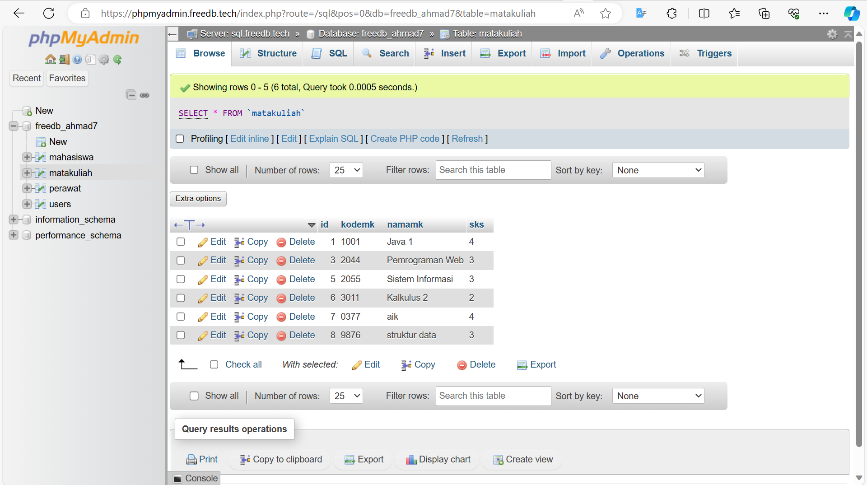


1. Jika anda ingin mengakses PHPMyAdmin, bisa klik icon database yang terdapat pada gambar di bawah ini. Masukkan pengisian data nama server (host), nama pengguna, beserta kata sandi yang didapatkan dari dashboard FreeDB.Tech tadi, lalu klik Kirim. Jika sudah nanti tampilannya seperti ini.



Masuk ke PhpMyAdmin :

1. Buka PhpMyAdmin pada website [Free Remote MySQL Hosting (freedb.tech)](https://freedb.tech/) untuk cara nya sudah di jelaskan pada poin no 6 di atas.
2. Klik pada data base yang sudah di buat, lalu import tabel matakuliah.sql di menu import di PhpMyAdmin. Untuk file matakuliah.sql bisa ambil di sini [AhmadMaulana220511138/PBO\_Semester3\_2023 (github.com)](https://github.com/AhmadMaulana220511138/PBO_Semester3_2023) pilih pertemuan 12, lalu pilih folder matakuliah di dalam foldernya ada file matakuliah.sql.
3. Jika sudah di import nanti tampilaannya akan seperti ini



Buka Visual Studio Code :

1. Buka Visual Studio Code, buat folder baru contoh : matakuliah
2. Masuk ke github ini [AhmadMaulana220511138/PBO\_Semester3\_2023 (github.com)](https://github.com/AhmadMaulana220511138/PBO_Semester3_2023) pilih pertemuan 12, masuk ke folder matakuliah lalu copy file yang Bernama db.py, frm.py dan matakuliah.py. Bisa juga copy dari code dibawah ini.
3. Buat nama file nya sama persis seperti poin no 2 di atas , berikut adalan code untuk db.py, frm.py, matakuliah.py
4. db.py

pastikan pada bagian def \_\_init\_\_(self) sudah di ganti dengan data anda sendiri

import mysql.connector as mc

class DBConnection:

    def \_\_init\_\_(*self*):

*self*.host = "sql.freedb.tech"

*self*.port = 3306

*self*.name = "isi dengan nama data base anda sendiri"

*self*.user = "isi dengan user anda sendiri"

*self*.password = "isi dengan password anda sendiri"

*self*.conn = None

*self*.cursor = None

*self*.result = None

*self*.connected = False

*self*.affected = 0

*self*.connect()

    @property

    def connection\_status(*self*):

        return *self*.connected

    def connect(*self*):

        try:

*self*.conn = mc.connect(*host* = *self*.host,

*port* = *self*.port,

*database* = *self*.name,

*user* = *self*.user,

*password* = *self*.password)

*self*.connected = True

*self*.cursor=*self*.conn.cursor()

        except mc.Error as e:

*self*.connected = False

        return *self*.conn

    def disconnect(*self*):

        if(*self*.connected==True):

*self*.conn.close

        else:

*self*.conn = None

    def findOne(*self*, *sql*):

*self*.connect()

*self*.cursor.execute(*sql*)

*self*.result = *self*.cursor.fetchone()

        return *self*.result

    def findAll(*self*, *sql*):

*self*.connect()

*self*.result = *self*.cursor.execute(*sql*)

*self*.result = *self*.cursor.fetchall()

        return *self*.result

    def insert(*self*, *sql*):

*self*.connect()

*self*.cursor.execute(*sql*)

*self*.conn.commit()

*self*.affected = *self*.cursor.rowcount

        return *self*.affected

    def update(*self*, *sql*, *val*):

*self*.connect()

*self*.cursor.execute(*sql*, *val*)

*self*.conn.commit()

*self*.affected = *self*.cursor.rowcount

        return *self*.affected

    def delete(*self*, *sql*):

*self*.connect()

*self*.cursor.execute(*sql*)

*self*.conn.commit()

*self*.affected = *self*.cursor.rowcount

        return *self*.affected

    def show(*self*, *sql*):

*self*.connect()

*self*.cursor.execute(*sql*)

*self*.result = *self*.cursor.fetchone()

        return *self*.result

    @property

    def info(*self*):

        if(*self*.connected==True):

            return "Server is running on " + *self*.host + ' using port ' + str(*self*.port)

        else:

            return "Server is offline."

*# test koneksi databse*

A = DBConnection()

B = A.info

print(B)

1. frm.py

import tkinter as tk

from tkinter import Frame,Label,Entry,Button,Radiobutton,ttk,VERTICAL,YES,BOTH,END,Tk,W,StringVar,messagebox

from matakuliah import Matakuliah

class FormMatakuliah:

    def \_\_init\_\_(*self*, *parent*, *title*):

*self*.parent = *parent*

*self*.parent.geometry("450x450")

*self*.parent.title(*title*)

*self*.parent.protocol("WM\_DELETE\_WINDOW", *self*.onKeluar)

*self*.ditemukan = None

*self*.aturKomponen()

*self*.onReload()

    def aturKomponen(*self*):

        mainFrame = Frame(*self*.parent, *bd*=10)

        mainFrame.pack(*fill*=BOTH, *expand*=YES)

*# Label*

        Label(mainFrame, *text*='Kode MK:').grid(*row*=0, *column*=0, *sticky*=W, *padx*=5, *pady*=5)

*self*.txtKodeMK = Entry(mainFrame)

*self*.txtKodeMK.grid(*row*=0, *column*=1, *padx*=5, *pady*=5)

*self*.txtKodeMK.bind("<Return>",*self*.onCari) *# menambahkan event Enter key*

        Label(mainFrame, *text*='Nama MK:').grid(*row*=1, *column*=0, *sticky*=W, *padx*=5, *pady*=5)

*self*.txtNamamk = Entry(mainFrame)

*self*.txtNamamk.grid(*row*=1, *column*=1, *padx*=5, *pady*=5)

        Label(mainFrame, *text*='Sks:').grid(*row*=2, *column*=0, *sticky*=W, *padx*=5, *pady*=5)

*self*.txtSKS = StringVar()

*self*.Java = Radiobutton(mainFrame, *text*='4', *value*='4', *variable*=*self*.txtSKS)

*self*.Java.grid(*row*=2, *column*=1, *padx*=5, *pady*=5, *sticky*=W)

*self*.Java.select() *# set pilihan yg pertama*

*self*.PemogramanWeb = Radiobutton(mainFrame, *text*='3', *value*='3', *variable*=*self*.txtSKS)

*self*.PemogramanWeb.grid(*row*=3, *column*=1, *padx*=5, *pady*=5, *sticky*=W)

*# self.txtKodeProdi = StringVar()*

*self*.SistemInformasi = Radiobutton(mainFrame, *text*='2', *value*='2', *variable*=*self*.txtSKS)

*self*.SistemInformasi.grid(*row*=4, *column*=1, *padx*=5, *pady*=5, *sticky*=W)

*# self.txtKodeProdi = StringVar()*

*# Button*

*self*.btnSimpan = Button(mainFrame, *text*='Simpan', *command*=*self*.onSimpan, *width*=10)

*self*.btnSimpan.grid(*row*=0, *column*=3, *padx*=5, *pady*=5)

*self*.btnClear = Button(mainFrame, *text*='Clear', *command*=*self*.onClear, *width*=10)

*self*.btnClear.grid(*row*=1, *column*=3, *padx*=5, *pady*=5)

*self*.btnHapus = Button(mainFrame, *text*='Hapus', *command*=*self*.onDelete, *width*=10)

*self*.btnHapus.grid(*row*=2, *column*=3, *padx*=5, *pady*=5)

*# define columns*

        columns = ('idmk', 'kodemk', 'namamk','sks')

*self*.tree = ttk.Treeview(mainFrame, *columns*=columns, *show*='headings')

*# define headings*

*self*.tree.heading('idmk', *text*='ID')

*self*.tree.column('idmk', *width*="30")

*self*.tree.heading('kodemk', *text*='Kode')

*self*.tree.column('kodemk', *width*="60")

*self*.tree.heading('namamk', *text*='Nama MK')

*self*.tree.column('namamk', *width*="200")

*self*.tree.heading('sks', *text*='SKS')

*self*.tree.column('sks', *width*="30")

*# set tree position*

*self*.tree.place(*x*=0, *y*=200)

*self*.onReload()

    def onClear(*self*, *event*=None):

*self*.txtKodeMK.delete(0,END)

*self*.txtKodeMK.insert(END,"")

*self*.txtNamamk.delete(0,END)

*self*.txtNamamk.insert(END,"")

*self*.btnSimpan.config(*text*="Simpan")

*self*.Java.select()

*self*.onReload()

*self*.ditemukan = False

    def onReload(*self*, *event*=None):

*# get data mahasiswa*

        mk = Matakuliah()

        result = mk.getAllData()

        for item in *self*.tree.get\_children():

*self*.tree.delete(item)

        students=[]

        for row\_data in result:

            students.append(row\_data)

        for student in students:

*self*.tree.insert('',END, *values*=student)

    def onCari(*self*, *event*=None):

        kodemk = *self*.txtKodeMK.get()

        mk = Matakuliah()

        res = mk.getByKodeMK(kodemk)

        rec = mk.affected

        if(rec>0):

            messagebox.showinfo("showinfo", "Data Ditemukan")

*self*.TampilkanData()

*self*.ditemukan = True

        else:

            messagebox.showwarning("showwarning", "Data Tidak Ditemukan")

*self*.ditemukan = False

*self*.txtNamamk.focus()

        return res

    def TampilkanData(*self*, *event*=None):

        kodemk = *self*.txtKodeMK.get()

        mk = Matakuliah()

        res = mk.getByKodeMK(kodemk)

*self*.txtNamamk.delete(0,END)

*self*.txtNamamk.insert(END,mk.namamk)

        sks = mk.sks

        if(sks=="4"):

*self*.Java.select()

        else:

*self*.PemogramanWeb.select()

*# self.txtKodeProdi.set(mk.kode\_prodi)*

*self*.btnSimpan.config(*text*="Update")

    def onSimpan(*self*, *event*=None):

        kodemk = *self*.txtKodeMK.get()

        namamk = *self*.txtNamamk.get()

        sks = *self*.txtSKS.get()

        mk = Matakuliah()

        mk.kodemk = kodemk

        mk.namamk = namamk

        mk.sks = sks

        if(*self*.ditemukan==True):

            res = mk.updateByKodeMK(kodemk)

            ket = 'Diperbarui'

        else:

            res = mk.simpan()

            ket = 'Disimpan'

        rec = mk.affected

        if(rec>0):

            messagebox.showinfo("showinfo", "Data Berhasil "+ket)

        else:

            messagebox.showwarning("showwarning", "Data Gagal "+ket)

*self*.onClear()

        return rec

    def onDelete(*self*, *event*=None):

        kodemk = *self*.txtKodeMK.get()

        mk = Matakuliah()

        mk.kodemk = kodemk

        if(*self*.ditemukan==True):

            res = mk.deleteByKodeMK(kodemk)

            rec = mk.affected

        else:

            messagebox.showinfo("showinfo", "Data harus ditemukan dulu sebelum dihapus")

            rec = 0

        if(rec>0):

            messagebox.showinfo("showinfo", "Data Berhasil dihapus")

*self*.onClear()

    def onKeluar(*self*, *event*=None):

*# memberikan perintah menutup aplikasi*

*self*.parent.destroy()

if \_\_name\_\_ == '\_\_main\_\_':

    root = tk.Tk()

    aplikasi = FormMatakuliah(root, "Aplikasi Data Mtakuliah")

    root.mainloop()

1. matakuliah.py

from db import DBConnection as mydb

class Matakuliah:

    def \_\_init\_\_(*self*):

*self*.\_\_id=None

*self*.\_\_kodemk=None

*self*.\_\_namamk=None

*self*.\_\_sks=None

*self*.conn = None

*self*.affected = None

*self*.result = None

    @property

    def id(*self*):

        return *self*.\_\_id

    @property

    def kodemk(*self*):

        return *self*.\_\_kodemk

    @kodemk.setter

    def kodemk(*self*, *value*):

*self*.\_\_kodemk = *value*

    @property

    def namamk(*self*):

        return *self*.\_\_namamk

    @namamk.setter

    def namamk(*self*, *value*):

*self*.\_\_namamk = *value*

    @property

    def sks(*self*):

        return *self*.\_\_sks

    @sks.setter

    def sks(*self*, *value*):

*self*.\_\_sks = *value*

    def simpan(*self*):

*self*.conn = mydb()

        val = (*self*.\_\_kodemk, *self*.\_\_namamk, *self*.\_\_sks)

        sql="INSERT INTO matakuliah (kodemk, namamk, sks) VALUES " + str(val)

*self*.affected = *self*.conn.insert(sql)

*self*.conn.disconnect

        return *self*.affected

    def update(*self*, *id*):

*self*.conn = mydb()

        val = (*self*.\_\_kodemk, *self*.\_\_namamk, *self*.\_\_sks, *id*)

        sql="UPDATE matakuliah SET kodemk = %s, namamk = %s, sks=%s, WHERE id=%s"

*self*.affected = *self*.conn.update(sql, val)

*self*.conn.disconnect

        return *self*.affected

    def updateByNIM(*self*, *kodemk*):

*self*.conn = mydb()

        val = (*self*.\_\_namamk, *self*.\_\_sks,*self*, *kodemk*)

        sql="UPDATE matakuliah SET namamk = %s, sks=%s, WHERE kodemk=%s"

*self*.affected = *self*.conn.update(sql, val)

*self*.conn.disconnect

        return *self*.affected

    def delete(*self*, *id*):

*self*.conn = mydb()

        sql="DELETE FROM matakuliah WHERE id='" + str(*id*) + "'"

*self*.affected = *self*.conn.delete(sql)

*self*.conn.disconnect

        return *self*.affected

    def deleteByNIM(*self*, *kodemk*):

*self*.conn = mydb()

        sql="DELETE FROM matakuliah WHERE nim='" + str(*kodemk*) + "'"

*self*.affected = *self*.conn.delete(sql)

*self*.conn.disconnect

        return *self*.affected

    def getByID(*self*, *id*):

*self*.conn = mydb()

        sql="SELECT \* FROM matakuliah WHERE id='" + str(*id*) + "'"

*self*.result = *self*.conn.findOne(sql)

*self*.\_\_kodemk = *self*.result[1]

*self*.\_\_namamk = *self*.result[2]

*self*.\_\_sks = *self*.result[3]

*self*.conn.disconnect

        return *self*.result

    def getByNIM(*self*, *kodemk*):

        a=str(*kodemk*)

        b=a.strip()

*self*.conn = mydb()

        sql="SELECT \* FROM mahasiswa WHERE nim='" + b + "'"

*self*.result = *self*.conn.findOne(sql)

        if(*self*.result!=None):

*self*.\_\_kodemk = *self*.result[1]

*self*.\_\_namamk = *self*.result[2]

*self*.\_\_sks = *self*.result[3]

*self*.affected = *self*.conn.cursor.rowcount

        else:

*self*.\_\_kodemk = ''

*self*.\_\_namamk = ''

*self*.\_\_sks = ''

*self*.affected = 0

*self*.conn.disconnect

        return *self*.result

    def getAllData(*self*):

*self*.conn = mydb()

        sql="SELECT \* FROM matakuliah"

*self*.result = *self*.conn.findAll(sql)

        return *self*.result

*# Tampilkan Data*

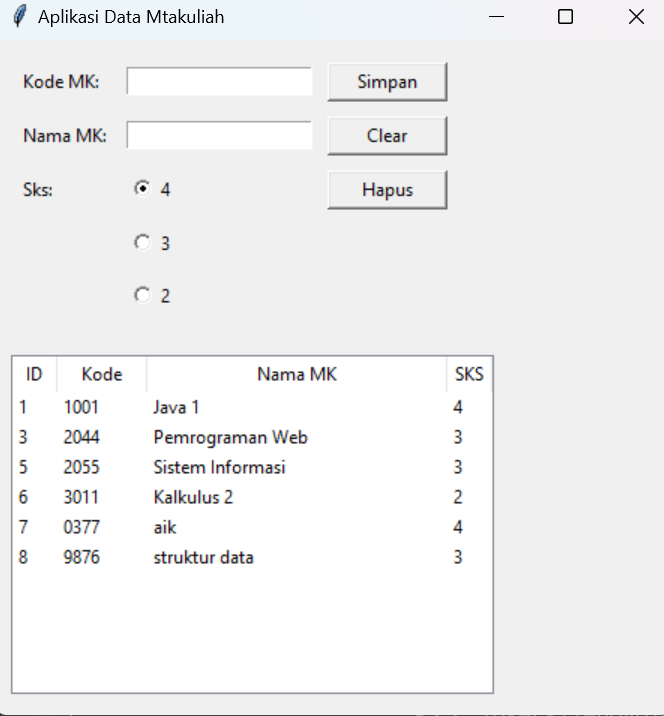
A = Matakuliah()

B = A.getAllData()

print(B)

1. Output frm.py, setelah di run :

Inputkan data terlebih dahulu, jika sudah klik tombol simpan otomatis data akan tersimpan di data base “yang anda buat” dan di tabel matakuliah di PhpMyAdmin.



1. Selanjutkan ada file tabel mahasiswa.sql dan perawat.sql. Masuk ke github ini [AhmadMaulana220511138/PBO\_Semester3\_2023 (github.com)](https://github.com/AhmadMaulana220511138/PBO_Semester3_2023) pilih pertemuan 12, masuk ke folder mahasiswa dan perawat, lalu copy file yang Bernama db.py, frm.py dan mahasiswa.py di folder mahasiswa.

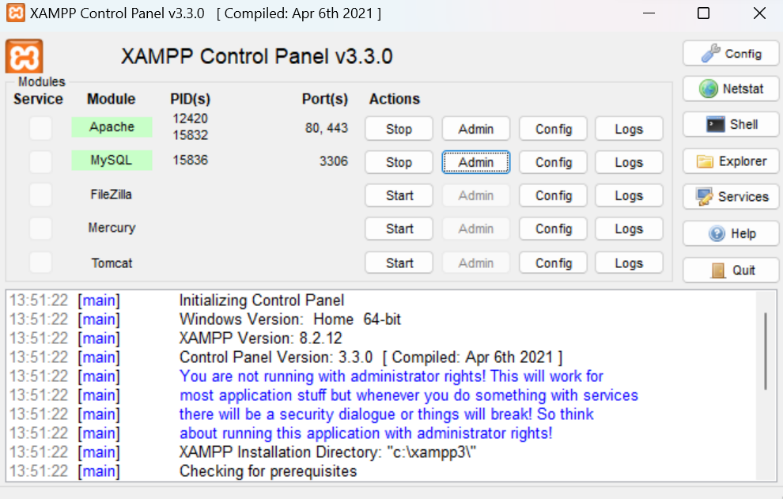
Copy juga file yang Bernama db.py, frm.py dan Perawat.py di folder perawat. buat nama file nya sama di visual studio code nya. Untuk cara pengerjaaannya sama seperti di atas yang sudah saya jelaskan.

# PERTEMUAN 13

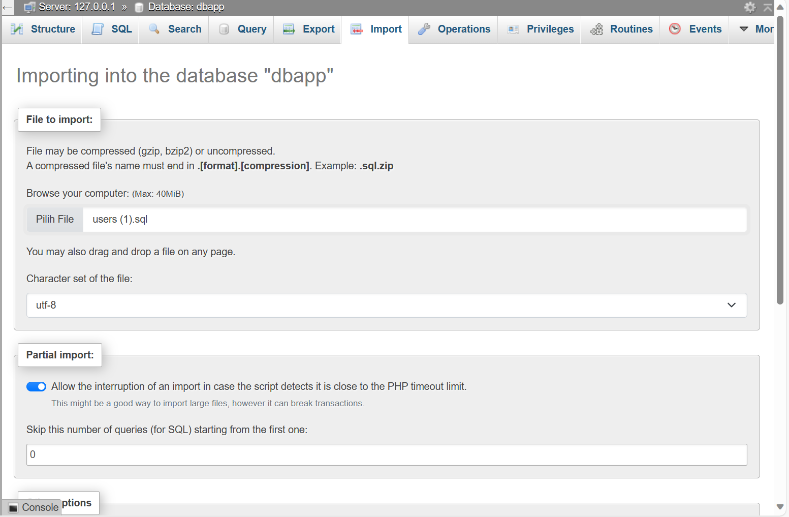
**aplikasi login multi user python-mysql**

Masuk ke aplikasi xmapp :

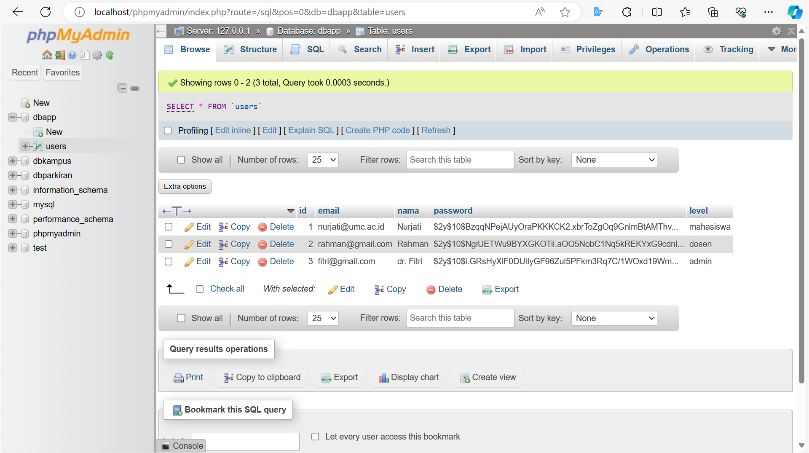
1. Buka aplikasi xmapp kemudian klik “start” pada Apache dan MySQL. Pastikan berwarna hijau untuk Apache dan MySQL nya.



1. Klik tombol Admin pada MySQL. Jika sudah di klik nanti akan di arahkan ke PhpMyAdmin.
2. Buat data base baru dengan nama “dbapp”. lalu import tabel Users.sql di menu import di PhpMyAdmin. Untuk file User.sql bisa ambil di sini [AhmadMaulana220511138/PBO\_Semester3\_2023 (github.com)](https://github.com/AhmadMaulana220511138/PBO_Semester3_2023) pilih pertemuan 13, lalu pilih folder tugas di dalam foldernya ada file User.sql.



1. Jika sudah di import file nya, nanti tampilannya akan seperti ini



Buka Visual Studio Code :

1. Buka Visual Studio Code, buat folder baru contoh : tugas
2. Masuk ke github ini [AhmadMaulana220511138/PBO\_Semester3\_2023 (github.com)](https://github.com/AhmadMaulana220511138/PBO_Semester3_2023) pilih pertemuan 13, masuk ke folder tugas lalu copy file yang Bernama Dashboard.py, FrmLingkaran.py, FrmLogin.py, FrmPersegi.py, FrmSegitiga.py, Users.py, db.py. Bisa juga copy dari code dibawah ini.
3. Buat nama file nya sama persis seperti di atas , berikut adalan code untuk Dashboard.py, FrmLingkaran.py, FrmLogin.py, FrmPersegi.py, FrmSegitiga.py, Users.py, db.py,
4. Dashboard.py

import tkinter as tk

from tkinter import Menu, messagebox

from FrmLogin import \*

from FrmPersegi import \*

from FrmSegitiga import \*

from FrmLingkaran import \*

class Dashboard:

    def \_\_init\_\_(*self*):

*# root window*

*self*.root = tk.Tk()

*self*.root.title('Menu Demo')

*# self.root.attributes('-fullscreen', True)*

*self*.root.geometry("900x400")

*self*.\_\_data = None

*self*.\_\_level = None

*# create a menubar*

*self*.menubar = Menu(*self*.root)

*self*.root.config(*menu*=*self*.menubar)

*# create menus*

*self*.file\_menu = Menu(*self*.menubar)

*self*.guest\_menu = Menu(*self*.menubar)

*self*.admin\_menu = Menu(*self*.menubar)

*self*.mahasiswa\_menu = Menu(*self*.menubar)

*self*.dosen\_menu = Menu(*self*.menubar)

*# add menu items to File menu*

*self*.file\_menu.add\_command(*label*='Login', *command*=lambda: *self*.new\_window("Log Me In", FormLogin))

*self*.file\_menu.add\_command(*label*='Exit', *command*=*self*.root.destroy)

*# add menu items to menu Admin*

*self*.admin\_menu.add\_command(*label*='Admin-1', *command*=lambda: *self*.new\_window("Luas Persegi", FrmPersegi))

*self*.admin\_menu.add\_command(*label*='Admin-2', *command*=lambda: *self*.new\_window("Luas Segitiga", FrmSegitiga))

*self*.admin\_menu.add\_command(*label*='Admin-3', *command*=lambda: *self*.new\_window("Luas Lingkaran", FrmLingkaran))

*# add menu items to menu Mahasiswa*

*self*.mahasiswa\_menu.add\_command(*label*='Mahasiswa-1', *command*=lambda: *self*.new\_window("Luas Persegi", FrmPersegi))

*self*.mahasiswa\_menu.add\_command(*label*='Mahasiswa-2', *command*=lambda: *self*.new\_window("Luas Segitiga", FrmSegitiga))

*self*.mahasiswa\_menu.add\_command(*label*='Mahasiswa-3', *command*=lambda: *self*.new\_window("Luas Lingkaran", FrmLingkaran))

*# add menu items to menu Dosen*

*self*.dosen\_menu.add\_command(*label*='Dosen-1', *command*=lambda: *self*.new\_window("Luas Persegi", FrmPersegi))

*self*.dosen\_menu.add\_command(*label*='Dosen-2', *command*=lambda: *self*.new\_window("Luas Segitiga", FrmSegitiga))

*self*.dosen\_menu.add\_command(*label*='Dosen-3', *command*=lambda: *self*.new\_window("Luas Lingkaran", FrmLingkaran))

*# add menus to the menubar*

*self*.menubar.add\_cascade(*label*="File", *menu*=*self*.file\_menu)

    def new\_window(*self*, *number*, *\_class*):

        new = tk.Toplevel(*self*.root)

        new.transient()

        new.grab\_set()

*\_class*(new, *number*, *self*.update\_main\_window)

    def update\_main\_window(*self*, *data*):

*# Method to receive data from child windows*

*self*.\_\_data = *data*

        level = *self*.\_\_data[0]

        loginvalid = *self*.\_\_data[1]

        if(loginvalid==True):

            index = *self*.file\_menu.index('Login')

*# hapus menu login*

*self*.file\_menu.delete(index)

*self*.file\_menu.add\_command(*label*='Logout', *command*=*self*.Logout)

*# tambahkan menu sesuai level*

            if(level=='admin'):

*self*.menubar.add\_cascade(*label*="Admin", *menu*=*self*.admin\_menu)

*self*.\_\_level = 'Admin'

            elif(level=='mahasiswa'):

*self*.menubar.add\_cascade(*label*="Mahasiswa", *menu*=*self*.mahasiswa\_menu)

*self*.\_\_level = 'Mahasiswa'

            elif(level=='dosen'):

*self*.menubar.add\_cascade(*label*="Dosen", *menu*=*self*.dosen\_menu)

*self*.\_\_level = 'Dosen'

            else:

                pass

    def Logout(*self*):

        index = *self*.file\_menu.index('Logout')

*self*.file\_menu.delete(index)

*self*.file\_menu.add\_command(*label*='Login', *command*=lambda: *self*.new\_window("Log Me In", FormLogin))

*self*.remove\_all\_menus()

    def remove\_all\_menus(*self*):

        index = *self*.menubar.index(*self*.\_\_level)

        if index is not None:

*self*.menubar.delete(index)

    def run(*self*):

*self*.root.mainloop()

if \_\_name\_\_ == "\_\_main\_\_":

    menu\_app = Dashboard()

    menu\_app.run()

1. FrmLingkaran.py

from tkinter import Frame,Label,Entry,Button,YES,BOTH,END,Tk,W

class FrmLingkaran:

    def \_\_init\_\_(*self*, *parent*, *title*, *update\_main\_window*):

*self*.parent = *parent*

*self*.update\_main\_window = *update\_main\_window*

*#self.parent.geometry("400x200")*

*self*.parent.title(*title*)

*self*.parent.protocol("WM\_DELETE\_WINDOW", *self*.onKeluar)

*self*.aturKomponen()

    def aturKomponen(*self*):

        mainFrame = Frame(*self*.parent, *bd*=10)

        mainFrame.pack(*fill*=BOTH, *expand*=YES)

*# pasang Label*

        Label(mainFrame, *text*='Jari-jari:').grid(*row*=0, *column*=0,

*sticky*=W, *padx*=5, *pady*=5)

        Label(mainFrame, *text*="Luas:").grid(*row*=3, *column*=0,

*sticky*=W, *padx*=5, *pady*=5)

*# pasang textbox*

*self*.txtJarijari = Entry(mainFrame)

*self*.txtJarijari.grid(*row*=0, *column*=1, *padx*=5, *pady*=5)

*self*.txtLuas = Entry(mainFrame)

*self*.txtLuas.grid(*row*=3, *column*=1, *padx*=5, *pady*=5)

*# Pasang Button*

*self*.btnHitung = Button(mainFrame, *text*='Hitung',

*command*=*self*.onHitung)

*self*.btnHitung.grid(*row*=2, *column*=1, *padx*=5, *pady*=5)

*# fungsi "onHitung" berfungsi untuk menghitung luas persegi panjang*

    def onHitung(*self*, *event*=None):

        jarijari = int(*self*.txtJarijari.get())

        luas = 22/7 \* jarijari\*\*2

*self*.txtLuas.delete(0,END)

*self*.txtLuas.insert(END,str(luas))

    def onKeluar(*self*, *event*=None):

*# memberikan perintah menutup aplikasi*

*self*.parent.destroy()

if \_\_name\_\_ == '\_\_main\_\_':

    def update\_main\_window(*result*):

        print(*result*)

    root = Tk()

    aplikasi = FrmLingkaran(root, "Program Luas Lingkaran")

    root.mainloop()

1. FrmLogin.py

import tkinter as tk

from tkinter import Frame,Label,Entry,Button,Radiobutton,ttk,VERTICAL,YES,BOTH,END,Tk,W,StringVar,messagebox

from tkinter import ttk

from tkinter import messagebox

from Users import \*

class FormLogin:

    def \_\_init\_\_(*self*, *parent*, *title*, *update\_main\_window*):

*self*.parent = *parent*

*self*.update\_main\_window = *update\_main\_window*

*#self.parent.geometry("250x150")*

*self*.parent.title(*title*)

*self*.parent.protocol("WM\_DELETE\_WINDOW", *self*.onKeluar)

*self*.aturKomponen()

    def aturKomponen(*self*):

        mainFrame = Frame(*self*.parent, *bd*=10)

        mainFrame.pack(*fill*=BOTH, *expand*=YES)

*# diatur agar tampil di tengah layar*

        screen\_width = mainFrame.winfo\_screenwidth()

        screen\_height = mainFrame.winfo\_screenheight()

        x = (screen\_width - 250) // 2

        y = (screen\_height - 150) // 2

        width = 250

        height = 150

*self*.parent.geometry(f"{width}x{height}+{x}+{y}")

*# Label*

        Label(mainFrame, *text*='Email:').grid(*row*=0, *column*=0, *sticky*=W, *padx*=5, *pady*=5)

*self*.txtEmail = Entry(mainFrame)

*self*.txtEmail.grid(*row*=0, *column*=1, *padx*=5, *pady*=5)

        Label(mainFrame, *text*='Password:').grid(*row*=1, *column*=0, *sticky*=W, *padx*=5, *pady*=5)

*self*.txtPassword = Entry(mainFrame)

*self*.txtPassword.grid(*row*=1, *column*=1, *padx*=5, *pady*=5)

*# Button*

*self*.btnSubmit = Button(mainFrame, *text*='Submit', *command*=*self*.onSubmit, *width*=10)

*self*.btnSubmit.grid(*row*=2, *column*=0, *padx*=5, *pady*=5)

*self*.btnCancel = Button(mainFrame, *text*='Cancel', *command*=*self*.onKeluar, *width*=10)

*self*.btnCancel.grid(*row*=2, *column*=1, *padx*=5, *pady*=5)

    def onSubmit(*self*, *event*=None):

        email = *self*.txtEmail.get()

        password = *self*.txtPassword.get()

        obj = Users()

        val = obj.Validasi(email, password)

        C = val[1]

        if C:

*self*.update\_main\_window(val)

*self*.parent.destroy()

*# Menampilkan pop-up login berhasil dengan warna hijau*

            messagebox.showinfo("Berhasil", "Login Berhasil", *icon*='info')

        else:

*# Menampilkan pop-up login gagal dengan warna merah*

            messagebox.showwarning("Login Gagal", "Email atau Password Salah")

        return val

    def onKeluar(*self*, *event*=None):

*# memberikan perintah menutup aplikasi*

*self*.parent.destroy()

if \_\_name\_\_ == '\_\_main\_\_':

    def update\_main\_window(*result*):

        print(*result*)

    root = tk.Tk()

    aplikasi = FormLogin(root, "Aplikasi Data Login",update\_main\_window)

    root.mainloop()

1. FrmPersegi.py

from tkinter import Frame,Label,Entry,Button,YES,BOTH,END,Tk,W

class FrmPersegi:

    def \_\_init\_\_(*self*, *parent*, *title*, *update\_main\_window*):

*self*.parent = *parent*

*self*.update\_main\_window = *update\_main\_window*

*#self.parent.geometry("400x200")*

*self*.parent.title(*title*)

*self*.parent.protocol("WM\_DELETE\_WINDOW", *self*.onKeluar)

*self*.aturKomponen()

    def aturKomponen(*self*):

        mainFrame = Frame(*self*.parent, *bd*=10)

        mainFrame.pack(*fill*=BOTH, *expand*=YES)

*# pasang Label*

        Label(mainFrame, *text*='Panjang:').grid(*row*=0, *column*=0,

*sticky*=W, *padx*=5, *pady*=5)

        Label(mainFrame, *text*="Lebar:").grid(*row*=1, *column*=0,

*sticky*=W, *padx*=5, *pady*=5)

        Label(mainFrame, *text*="Luas:").grid(*row*=3, *column*=0,

*sticky*=W, *padx*=5, *pady*=5)

*# pasang textbox*

*self*.txtPanjang = Entry(mainFrame)

*self*.txtPanjang.grid(*row*=0, *column*=1, *padx*=5, *pady*=5)

*self*.txtLebar = Entry(mainFrame)

*self*.txtLebar.grid(*row*=1, *column*=1, *padx*=5, *pady*=5)

*self*.txtLuas = Entry(mainFrame)

*self*.txtLuas.grid(*row*=3, *column*=1, *padx*=5, *pady*=5)

*# Pasang Button*

*self*.btnHitung = Button(mainFrame, *text*='Hitung',

*command*=*self*.onHitung)

*self*.btnHitung.grid(*row*=2, *column*=1, *padx*=5, *pady*=5)

*# fungsi "onHitung" berfungsi untuk menghitung luas persegi panjang*

    def onHitung(*self*, *event*=None):

        panjang = int(*self*.txtPanjang.get())

        lebar = int(*self*.txtLebar.get())

        luas = panjang \* lebar

*self*.txtLuas.delete(0,END)

*self*.txtLuas.insert(END,str(luas))

*self*.update\_main\_window(str(luas))

    def onKeluar(*self*, *event*=None):

*# memberikan perintah menutup aplikasi*

*self*.parent.destroy()

if \_\_name\_\_ == '\_\_main\_\_':

    def update\_main\_window(*result*):

        print(*result*)

    root = Tk()

    aplikasi = FrmPersegi(root, "Program Luas Persegi Panjang")

    root.mainloop()

1. FrmSegitiga.py

from tkinter import Frame,Label,Entry,Button,YES,BOTH,END,Tk,W

class FrmSegitiga:

    def \_\_init\_\_(*self*, *parent*, *title*, *update\_main\_window*):

*self*.parent = *parent*

*self*.update\_main\_window = *update\_main\_window*

*#self.parent.geometry("400x200")*

*self*.parent.title(*title*)

*self*.parent.protocol("WM\_DELETE\_WINDOW", *self*.onKeluar)

*self*.aturKomponen()

    def aturKomponen(*self*):

        mainFrame = Frame(*self*.parent, *bd*=10)

        mainFrame.pack(*fill*=BOTH, *expand*=YES)

*# pasang Label*

        Label(mainFrame, *text*='Lebar Alas:').grid(*row*=0, *column*=0,

*sticky*=W, *padx*=5, *pady*=5)

        Label(mainFrame, *text*="Tinggi:").grid(*row*=1, *column*=0,

*sticky*=W, *padx*=5, *pady*=5)

        Label(mainFrame, *text*="Luas Segitiga:").grid(*row*=3, *column*=0,

*sticky*=W, *padx*=5, *pady*=5)

*# pasang textbox*

*self*.txtAlas = Entry(mainFrame)

*self*.txtAlas.grid(*row*=0, *column*=1, *padx*=5, *pady*=5)

*self*.txtTinggi = Entry(mainFrame)

*self*.txtTinggi.grid(*row*=1, *column*=1, *padx*=5, *pady*=5)

*self*.txtLuas = Entry(mainFrame)

*self*.txtLuas.grid(*row*=3, *column*=1, *padx*=5, *pady*=5)

*# Pasang Button*

*self*.btnHitung = Button(mainFrame, *text*='Hitung',

*command*=*self*.onHitung)

*self*.btnHitung.grid(*row*=2, *column*=1, *padx*=5, *pady*=5)

*# fungsi "onHitung" berfungsi untuk menghitung luas persegi panjang*

    def onHitung(*self*, *event*=None):

        alas = int(*self*.txtAlas.get())

        tinggi = int(*self*.txtTinggi.get())

        luas = 0.5 \* alas \* tinggi

*self*.txtLuas.delete(0,END)

*self*.txtLuas.insert(END,str(luas))

    def onKeluar(*self*, *event*=None):

*# memberikan perintah menutup aplikasi*

*self*.parent.destroy()

if \_\_name\_\_ == '\_\_main\_\_':

    def update\_main\_window(*result*):

        print(*result*)

    root = Tk()

    aplikasi = FrmSegitiga(root, "Program Luas Persegi Panjang")

    root.mainloop()

1. Users.py

import bcrypt

from db import DBConnection as mydb

class Users:

    def \_\_init\_\_(*self*):

*self*.\_\_id= None

*self*.\_\_email= None

*self*.\_\_nama= None

*self*.\_\_password= None

*self*.\_\_level= None

*self*.\_\_uservalid = None

*self*.\_\_passwordvalid = None

*self*.\_\_loginvalid = None*#*

*self*.conn = None

*self*.affected = None

*self*.result = None

    @property

    def id(*self*):

        return *self*.\_\_id

    @property

    def email(*self*):

        return *self*.\_\_email

    @email.setter

    def email(*self*, *value*):

*self*.\_\_email = *value*

    @property

    def nama(*self*):

        return *self*.\_\_nama

    @nama.setter

    def nama(*self*, *value*):

*self*.\_\_nama = *value*

    @property

    def password(*self*):

        return *self*.\_\_password

    @password.setter

    def password(*self*, *value*):

*self*.\_\_password = *value*

    @property

    def level(*self*):

        return *self*.\_\_level

    @level.setter

    def level(*self*, *value*):

*self*.\_\_level = *value*

    @property

    def loginvalid(*self*):

        return *self*.\_\_loginvalid

    @loginvalid.setter

    def loginvalid(*self*, *value*):

*self*.\_\_loginvalid = *value*

    def cekUsername(*self*, *email*):

*self*.conn = mydb()

        sql="SELECT \* FROM users WHERE email='" + *email* + "'"

*self*.result = *self*.conn.findOne(sql)

        if(*self*.result!=None):

*self*.\_\_email = *self*.result[1]

*self*.\_\_nama = *self*.result[2]

*self*.\_\_password = *self*.result[3]

*self*.\_\_level = *self*.result[4]

*self*.affected = *self*.conn.cursor.rowcount

*self*.\_\_uservalid = True

        else:

*self*.\_\_email = ''

*self*.\_\_nama = ''

*self*.\_\_password = ''

*self*.\_\_level = ''

*self*.affected = 0

*self*.\_\_uservalid = False

        return *self*.\_\_uservalid

    def cekPassword(*self*, *password*):

        hashedpass=*self*.\_\_password.encode('utf-8')

        c = *password*.encode('utf-8')

        d = bcrypt.checkpw(c, hashedpass)

        if(d):

*self*.\_\_passwordvalid=True

        else:

*self*.\_\_passwordvalid=False

        return *self*.\_\_passwordvalid

    def Validasi(*self*, *email*, *password*):

        a = *self*.cekUsername(*email*)

        if(a==True):

            b = *self*.cekPassword(*password*)

            if(b==True):

*self*.\_\_loginvalid=True

            else:

*self*.\_\_loginvalid=False

        else:

*self*.\_\_loginvalid=False

        val = []

        val = [*self*.\_\_level, *self*.\_\_loginvalid]

        return val

*# A = Users()*

*# print("\n\n")*

*# print("Username BENAR, dan password BENAR")*

*# B = A.Validasi('nurjati@umc.ac.id','123')*

*# print(B)*

*# print("\n\nUsername BENAR, dan password SALAH")*

*# C = A.Validasi('nurjati@umc.ac.id','1234')*

*# print(C)*

*# print("\n\nUsername SALAH, dan password BENAR")*

*# D = A.Validasi('baim@umc.ac.id','123')*

*# print(D)*

*# print("\n\nUsername SALAH, dan password SALAH")*

*# E = A.Validasi('baim@umc.ac.id','1234')*

*# print(E)*

*# print("\n\n")*

1. db.py

import mysql.connector as mc

class DBConnection:

    def \_\_init\_\_(*self*):

*self*.host = "localhost"

*self*.port = 3306

*self*.name = "dbapp"

*self*.user = "root"

*self*.password = ""

*self*.conn = None

*self*.cursor = None

*self*.result = None

*self*.connected = False

*self*.affected = 0

*self*.connect()

    @property

    def connection\_status(*self*):

        return *self*.connected

    def connect(*self*):

        try:

*self*.conn = mc.connect(*host* = *self*.host,

*port* = *self*.port,

*database* = *self*.name,

*user* = *self*.user,

*password* = *self*.password)

*self*.connected = True

*self*.cursor=*self*.conn.cursor()

        except mc.Error as e:

*self*.connected = False

        return *self*.conn

    def disconnect(*self*):

        if(*self*.connected==True):

*self*.conn.close

        else:

*self*.conn = None

    def findOne(*self*, *sql*):

*self*.connect()

*self*.cursor.execute(*sql*)

*self*.result = *self*.cursor.fetchone()

        return *self*.result

    def findAll(*self*, *sql*):

*self*.connect()

*self*.result = *self*.cursor.execute(*sql*)

*self*.result = *self*.cursor.fetchall()

        return *self*.result

    def insert(*self*, *sql*):

*self*.connect()

*self*.cursor.execute(*sql*)

*self*.conn.commit()

*self*.affected = *self*.cursor.rowcount

        return *self*.affected

    def update(*self*, *sql*, *val*):

*self*.connect()

*self*.cursor.execute(*sql*, *val*)

*self*.conn.commit()

*self*.affected = *self*.cursor.rowcount

        return *self*.affected

    def delete(*self*, *sql*):

*self*.connect()

*self*.cursor.execute(*sql*)

*self*.conn.commit()

*self*.affected = *self*.cursor.rowcount

        return *self*.affected

    def show(*self*, *sql*):

*self*.connect()

*self*.cursor.execute(*sql*)

*self*.result = *self*.cursor.fetchone()

        return *self*.result

    @property

    def info(*self*):

        if(*self*.connected==True):

            return "Server is running on " + *self*.host + ' using port ' + str(*self*.port)

        else:

            return "Server is offline."

*# test koneksi databse*

A = DBConnection()

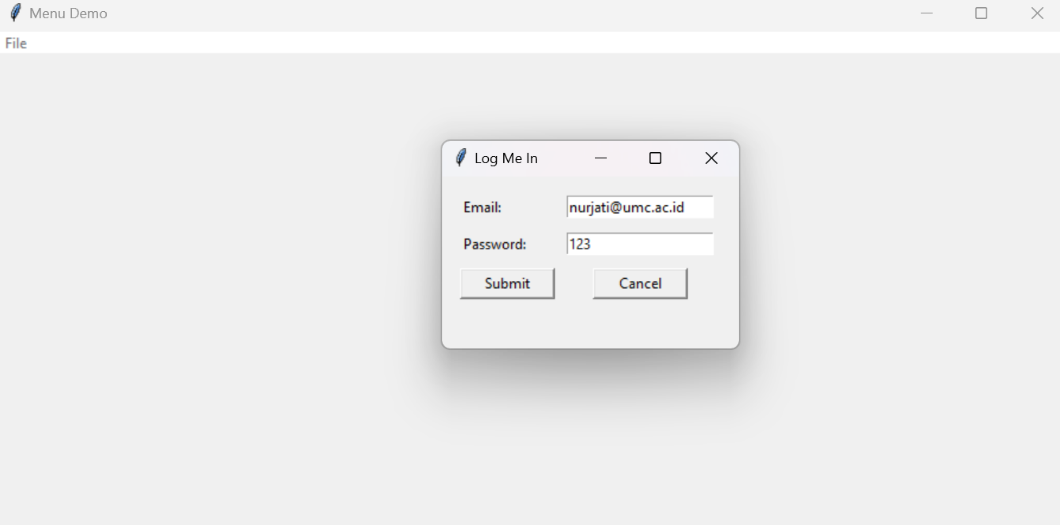
B = A.info

print(B)

1. Output Dashboard.py setelah di run :

Catatan :

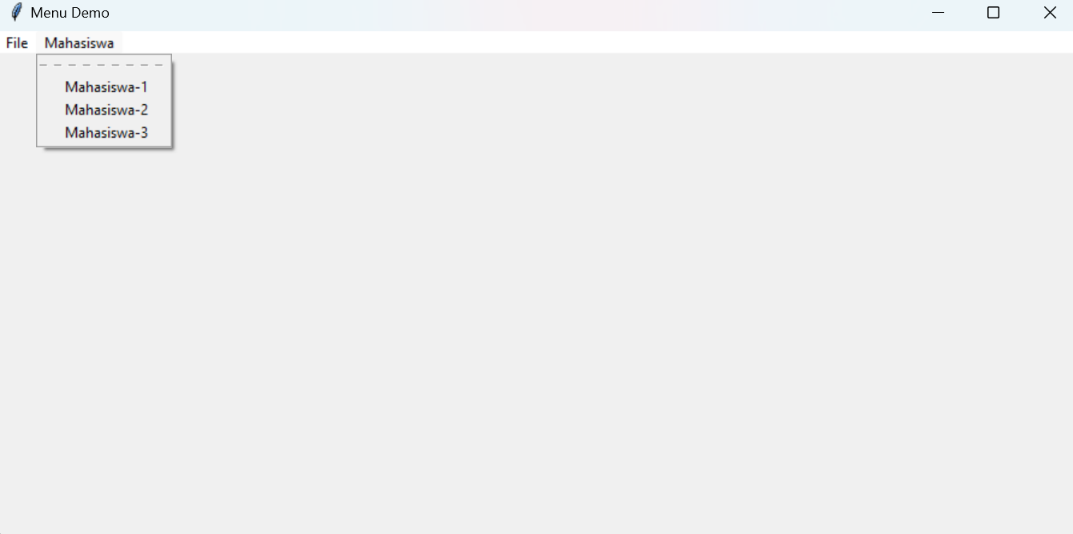
* Untuk email bisa ambil/copy di data base “dbapp” dengan nama tabel Users di PhpMyAdmin.
* Untuk password yaitu “123” untuk semua email di tabel Users.

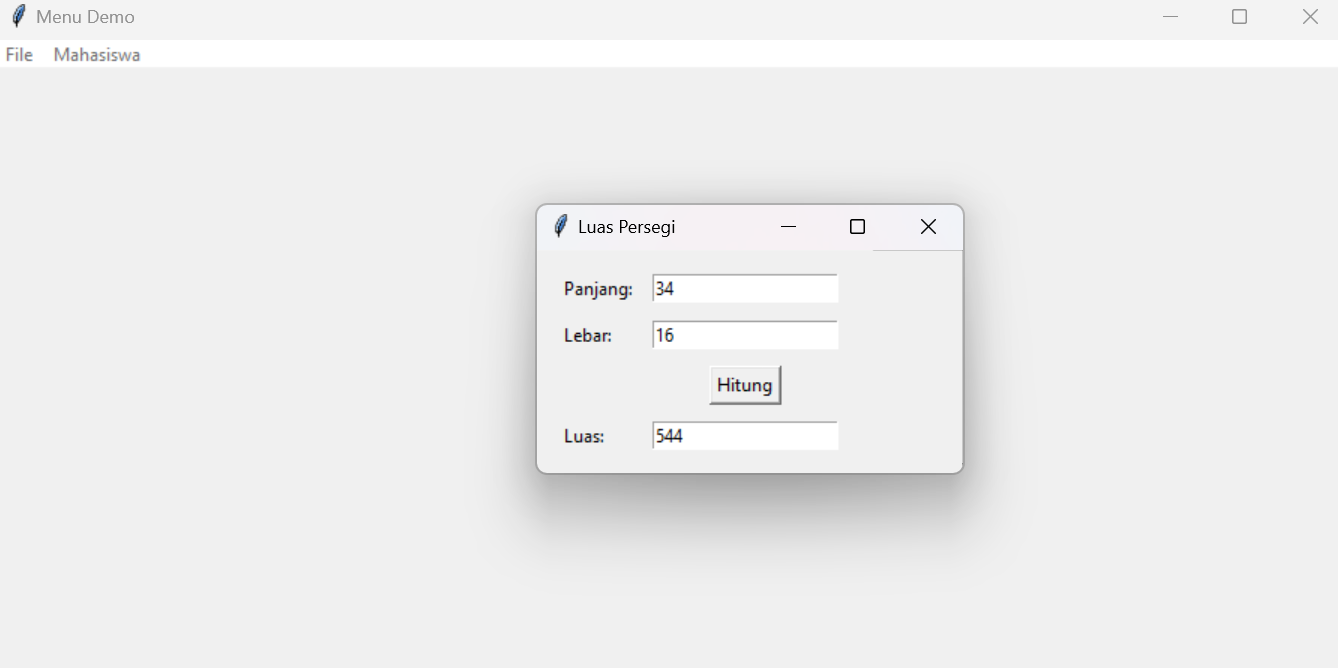


Jika berhasil Login tampilannya seperti ini. Muncul menu Mahasiswa yang di dalamnya ada 3 aplikasi yang berbeda beda .

Catatan :

* Setiap email memiliki menu yang berbeda beda. Tetapi untuk aplikasi yang di dalamnya itu sama .





# PERTEMUAN 14 Membuat modul praktikum PBO

Yaitu ini Modul nya