****

# DAFTAR ISI

[DAFTAR ISI ii](#_Toc157298621)

[Langkah Awal 1](#_Toc157298622)

[Pertemuan 1 Pengenalan Python 2](#_Toc157298623)

[Pertemuan 2 If, Loop dan Array 4](#_Toc157298624)

[Pertemuan 3 Function 8](#_Toc157298625)

[Pertemuan 4 Module From dan Import 12](#_Toc157298626)

[Pertemuan 5 Quis 18](#_Toc157298627)

[Pertemuan 6 Aplikasi Bantu (Tools) 20](#_Toc157298628)

[Pertemuan 7 Konversi Suhu GUI 34](#_Toc157298629)

[Pertemuan 8 UTS 39](#_Toc157298630)

[Pertemuan 9 Class dan Object (OOP) 41](#_Toc157298631)

[Pertemuan 10 Pengenalan Python Database 45](#_Toc157298632)

[Pertemuan 11 Python Database 48](#_Toc157298633)

[Pertemuan 12 Database Using Remote MySQL Server on freedb.tech 57](#_Toc157298634)

[Petemuan 13 Aplikasi Login Multi User Python-MySQL 60](#_Toc157298635)

[Pertemuan 14 Tugas Modul 76](#_Toc157298636)

# Langkah Awal

Sebelum Anda menggunakan Python, Anda harus menginstalnya terlebih dahulu di sistem operasi komputer Anda. Disini kita akan belajar bahasa pemrograman Python menggunakan versi terbaru 3.11.5.

Cara menginstal python pada Windows:

1. Buka browser, kunjungi <http://www.python.org/downloads/windows/>
2. ATAU, klik direct link <https://www.python.org/ftp/python/3.11.5/python-3.11.5-amd64.exe>
3. Buka (klik 2x) file installer python yang baru saja di download
4. Ikuti langkah instalasi sampai selesai
5. Jika sudah terinstall, install juga extension Python pada Visual Studio Code

Langkah mengecek Python:

1. Buka CMD
2. Ketikan python –version
3. Maka akan muncul versi Pythonnya yang sudah terinstal

# Pertemuan 1 Pengenalan Python

1. Syntax Dasar dalam Aplikas Menghitung Luas dan Volume Balok

Dibawah ini adalah contoh fungsi Python yang digunakan untuk mencetak. Di Python untuk mencetak cukup gunakan fungsi *print()*, contohnya seperti *print("Aplikasi Menghitung Luas dan Volume Balok")*.

1. Input

Contoh pada program Aplikasi Menghitung dan Volume Balok, terdapat *Variabel* p yang berarti panjang, l yang berarti lebar dan t yang berarti tinggi. Bertipe data Integer untuk inputnya. Aturan penulisan variabel:

* Karakter pertama harus berupa huruf atau garis bawah/underscore \_
* Karakter selanjutnya dapat berupa huruf, garis bawah/underscore \_ atau angka
* 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.

Contoh penulisannya dalam kasus diatas:

p = int(input("Panjang: "))

l = int(input("Lebar: "))

t = int(input("Tinggi: "))

1. Proses

Dalam prosesnya kita mencari luas dan volume Balok, didalamnya terdapat variabel luas dan volume, terdapat juga operator yang digunakan yaitu kali (\*) dan tambah (+).

Contoh penulisannya yaitu:

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

volume = (p \* l \* t)

Dalam proses tersbut memanggil variabel p, l dan t.

1. Output

Untuk mencetak hasil dari pemrosesan di atas harus mengetik print() terlebih dahulu kemudian didalam kurung tersebut harus mencantukan atau memanggil variabel luas dan volume. Contohnya:

print("Hasil Perhitungan")

print("Luas:", luas)

print("Volume:", volume)

Hasil outputnya ketika nilai p = 10, l = 5 dan t = 4 yaitu:

  
Syntax lengkapnya:

print("Aplikasi Menghitung Luas dan Volume Balok")

# Atur Nilai

print("Masukkan Nilai")

p = int(input("Panjang: "))

l = int(input("Lebar: "))

t = int(input("Tinggi: "))

# Rumus

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

volume = (p \* l \* t)

# Output

print("Hasil Perhitungan")

print("Luas:", luas)

print("Volume:", volume

# Pertemuan 2 If, Loop dan Array

1. If

* If Statement

If statement digunakan untuk menjalankan blok kode jika suatu kondisi terpenuhi. Contoh:

bilangan = int(input("Masukkan bilangan: "))

if bilangan > 0:

print(f"{bilangan} adalah bilangan positif.")

Penjelasan:

* Program meminta pengguna memasukkan bilangan.
* Jika bilangan tersebut lebih besar dari 0, blok kode di dalam `if` dieksekusi, mencetak bahwa bilangan tersebut positif.
* If Else Statement (Pernyataan If Else)

If else statement digunakan untuk menjalankan dua blok kode berbeda tergantung pada kondisi. Contoh:

bilangan = int(input("Masukkan bilangan: "))

if bilangan > 0:

print(f"{bilangan} adalah bilangan positif.")

else:

print(f"{bilangan} adalah bilangan non-positif.")

Penjelasan:

* Program meminta pengguna memasukkan bilangan.
* Jika bilangan lebih besar dari 0, blok kode dalam `if` dieksekusi; jika tidak, blok kode dalam `else` dieksekusi.
* Elif Statement (Pernyataan Elif)

Elif statement digunakan untuk menangani beberapa kondisi secara berurutan. Contoh:

nilai = int(input("Masukkan nilai: "))

if nilai > 80:

print("Nilai A")

elif nilai > 60:

print("Nilai B")

elif nilai > 40:

print("Nilai C")

else:

print("Nilai D")

Penjelasan:

* Program meminta pengguna memasukkan nilai.
* Bergantung pada rentang nilai, program akan mencetak grade yang sesuai.

1. Loop

Loop digunakan untuk melakukan eksekusi berulang terhadap satu atau beberapa blok kode. Python memiliki dua jenis utama loop, yaitu *`for*` dan *`while`*.

* For Loop

For loop digunakan untuk mengulangi blok kode sejumlah elemen dalam suatu urutan. Contoh:

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

jumlah = 0

for elemen in angka:

jumlah += elemen

print(f"Jumlah elemen dalam array: {jumlah}")

* While Loop

While loop digunakan untuk mengulangi blok kode selama suatu kondisi terpenuhi. Contoh:

counter = 0

while counter < 5:

print(f"Iterasi ke-{counter+1}")

counter += 1

1. Array

Array di Python biasanya diimplementasikan menggunakan daftar (`list`). List adalah struktur data yang dapat menyimpan elemen-elemen berbagai tipe data. Contoh:

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

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

campuran = [1, "dua", 3.0, True]

# Mengakses elemen dalam list

print(angka[2]) # Output: 3

# Menambahkan elemen ke dalam list

buah.append("anggur")

print(buah) # Output: ['apel', 'jeruk', 'pisang', 'anggur']

Penjelasan:

* Penggunaan `if`, `for`, dan `while` harus memperhatikan indentasi yang benar untuk menandai blok kode yang terkait.
* List di Python dimulai dari indeks 0.

1. Contoh pengaplikasian If, Loop dan Array dalam 1 studi kasus

# Program Menentukan Grade Mahasiswa

# Inisialisasi list mahasiswa beserta nilai

mahasiswa = [

{"nama": "Ibnu", "nilai": 85},

{"nama": "Haris", "nilai": 72},

{"nama": "Hafizt", "nilai": 60},

{"nama": "Pedal", "nilai": 92},

]

# Loop untuk mengecek grade setiap mahasiswa

for data in mahasiswa:

nama = data["nama"]

nilai = data["nilai"]

if nilai > 90:

grade = "A"

elif nilai > 80:

grade = "B"

elif nilai > 70:

grade = "C"

elif nilai > 60:

grade = "D"

else:

grade = "E"

print(f"{nama} mendapatkan nilai {nilai}, sehingga mendapat grade {grade}")

Penjelasan:

* + Program memiliki list `mahasiswa` yang berisi dictionary dengan informasi nama dan nilai setiap mahasiswa.
  + Menggunakan loop `for` untuk iterasi setiap mahasiswa dalam list.
  + Dalam loop, menggunakan `if`, `elif`, dan `else` untuk menentukan grade berdasarkan nilai mahasiswa.
  + Program mencetak nama, nilai, dan grade setiap mahasiswa ke layar.

Output:

A black background with white text

Description automatically generated

# Pertemuan 3 Function

1. Fungsi dan GUI

Function atau Fungsi adalah blok kode terorganisir dan dapat digunakan kembali yang digunakan untuk melakukan sebuah tindakan/action. Fungsi memberikan modularitas yang lebih baik untuk aplikasi Anda dan tingkat penggunaan kode yang tinggi.

Berikut adalah aturan sederhana untuk mendefinisikan fungsi dengan Python.

1. Mulai blok fungsi dengan kata kunci `def`, diikuti oleh nama fungsi dan tanda kurung `(())`.
2. Tempatkan parameter masukan atau argumen di dalam tanda kurung tersebut. Parameter dapat dijelaskan di dalam tanda kurung.
3. Opsional: Gunakan pernyataan pertama sebagai docstring, yang merupakan string dokumentasi untuk fungsi.
4. Mulai blok kode fungsi dengan titik dua (`:`) dan indentasi.
5. Gunakan pernyataan `return [ekspresi]` untuk mengembalikan nilai dari fungsi. Pernyataan ini bersifat opsional jika tidak digunakan fungsi akan mengembalikan`None`

GUI adalah singkatan dari "Graphical User Interface" (Antarmuka Pengguna Grafis). Ini merujuk pada cara pengguna berinteraksi dengan perangkat lunak melalui elemen-elemen grafis seperti ikon, tombol, jendela, dan menu, daripada menggunakan perintah teks. Dalam GUI, pengguna dapat berinteraksi dengan aplikasi menggunakan perangkat keras input seperti mouse, keyboard, dan layar sentuh. Salah satu contohnya yaitu Tkinter, Tkinter adalah library standar Python yang digunakan untuk membuat antarmuka pengguna grafis (GUI). Ini menyediakan berbagai widget dan tools untuk membangun aplikasi GUI dengan mudah.

Contoh kasus yang didalam programnya terdapat function dan GUI yaitu Aplikasi Penghitung Luas dan Volume Balok :

* Import Library

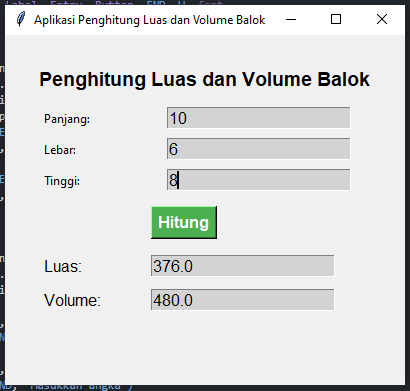
Pertama yang harus dilakukan yaitu mengimport library Tk inter dengan cara mengetik:

import tkinter as tk

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

Pada tkinter ini yang kami pakai yaitu ada Frame, Label, Entry, Button, END, W serta font.

* Fungsi
  + Fungsi `hitung\_luas()`:
* Fungsi ini digunakan untuk menghitung luas balok berdasarkan panjang (`p`), lebar (`l`), dan tinggi (`t`) yang dimasukkan oleh pengguna.
* Jika input valid, luas dihitung menggunakan rumus `2 \* (p \* l + p \* t + l \* t)`.
* Hasil perhitungan luas dibulatkan ke dua desimal dan ditampilkan di dalam widget `txtLuas`.
  + Fungsi `hitung\_volume()`:
* Fungsi ini digunakan untuk menghitung volume balok berdasarkan panjang (`p`), lebar (`l`), dan tinggi (`t`) yang dimasukkan oleh pengguna.
* Jika input valid, volume dihitung menggunakan rumus `p \* l \* t`.
* Hasil perhitungan volume dibulatkan ke dua desimal dan ditampilkan di dalam widget `txtVolume`.
  + Fungsi `hitung(event=None)`:
* Fungsi ini memanggil kedua fungsi `hitung\_luas()` dan `hitung\_volume()` untuk menghitung luas dan volume secara bersamaan.
* Fungsi ini juga dipanggil saat tombol "Hitung" ditekan atau saat tombol Enter ditekan.
* Setting GUI
  + Tkinter Window (`app`):
    - Menginisialisasi main window aplikasi menggunakan `Tk()`.
  + Frame (`frame`):
    - Digunakan untuk mengelompokkan elemen-elemen GUI agar tata letaknya rapi.
    - Memiliki latar belakang berwarna abu-abu muda (`bg="#f0f0f0"`).
  + Label (`judul`, `panjang`, `lebar`, `tinggi`, `luas`, `volume`):
    - Menampilkan teks sebagai label untuk memberikan petunjuk atau nama input/output.
    - Menggunakan font Arial dengan berbagai ukuran dan gaya.
  + Entry (`txtpanjang`, `txtlebar`, `txttinggi`, `txtLuas`, `txtVolume`):
    - Menyediakan input teks untuk pengguna memasukkan nilai panjang, lebar, tinggi, luas, dan volume.
    - Mempunyai latar belakang berwarna abu-abu muda (`bg="lightgray"`).
  + Button (`hitung\_button`):
    - Tombol "Hitung" yang digunakan untuk memicu perhitungan luas dan volume.
    - Menggunakan warna hijau (`bg="#4CAF50"`) dan teks putih.
  + Binding (\`app.bind('<Return>', hitung) ):
    - Menambahkan binding untuk tombol Enter, sehingga fungsi `hitung` akan dipanggil saat tombol Enter ditekan.
  + Main Loop (\`app.mainloop() ):
    - Membuat aplikasi tetap berjalan dan menunggu input pengguna atau interaksi GUI.
* Output Hasil



# Pertemuan 4 Module From dan Import

Module From dan Import digunakan untuk memisah antara main (GUI) dan fungsi.

1. From

Menggunakan pernyataan from diikuti dengan modul atau bagian modul yang ingin diimpor. Contoh untuk program aplikasinya yaitu Aplikasi Menghitung Luas dan Volume Balok. Pada main.py terdapat program GUI atau tampilan windowsnya, terdapat juga berbagai fungsi atau def hitung luas dan hitung volume pada file fungsi.py.

Contoh program pada fungsi.py:

def hitung\_luas(p, l, t):

    try:

        p = float(p)

        l = float(l)

        t = float(t)

        L = round(((2 \* p \* l) + (2 \* p \* t) + (2 \* l \* t)), 2)

        return L

    except ValueError:

        return "Masukkan angka"

def hitung\_volume(p, l, t):

    try:

        p = float(p)

        l = float(l)

        t = float(t)

        V = round((p \* l \* t), 2)

        return V

    except ValueError:

        return "Masukkan angka"

Jika sudah membuat fungsi pada fungsi.py maka kita masukkan ke dalam main.py dengan mengetik syntax:

from fungsi import hitung\_luas, hitung\_volume

Full program pada main.py:

import tkinter as tk

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

from fungsi import hitung\_luas, hitung\_volume

# Create tkinter object

app = Tk()

# Tambahkan Judul

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

# Windows

frame = Frame(app)

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

# Label Panjang

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

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

# Textbox Panjang

txtp = Entry(frame)

txtp.grid(row=0, column=1)

# Label Lebar

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

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

# Textbox Lebar

txtl = Entry(frame)

txtl.grid(row=1, column=1)

# Label Tinggi

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

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

# Textbox Tinggi

txtt = Entry(frame)

txtt.grid(row=2, column=1)

# Button

def hitung():

    hasil\_luas = hitung\_luas(txtp.get(), txtl.get(), txtt.get())

    hasil\_volume = hitung\_volume(txtp.get(), txtl.get(), txtt.get())

    txtluas.delete(0, END)

    txtvolume.delete(0, END)

    txtluas.insert(END, hasil\_luas)

    txtvolume.insert(END, hasil\_volume)

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

hitung\_button.grid(row=3, column=1, sticky=W, padx=5, pady=5)

# Output Label Luas

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

luas.grid(row=4, column=0, sticky=W, padx=5, pady=5)

# Output Textbox Luas

txtluas = Entry(frame)

txtluas.grid(row=4, column=1)

# Output Label Volume

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

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

# Output Textbox Volume

txtvolume = Entry(frame)

txtvolume.grid(row=5, column=1)

app.mainloop()

1. Import

Menggunakan pernyataan import tanpa from. Ini mengimpor seluruh modul ke dalam program. Contoh untuk program aplikasinya yaitu Aplikasi Menghitung Luas dan Volume Limas Segitiga. Pada main.py terdapat program GUI atau tampilan windowsnya, terdapat juga berbagai fungsi atau def hitung luas dan hitung volume namun hanya terdapat syntax pemrosesan atau rumus saja pada file fungsi.py.

Contoh program pada fungsi.py:

def hitung\_luas\_segitiga(LS\_1, LS\_2, LS\_3, LS\_4):

    return round((LS\_1 + LS\_2 + LS\_3 + LS\_4), 2)

def hitung\_volume\_limas(a, t, T):

    return round((1/6 \* a \* t \* T), 2)

Jika sudah membuat fungsi pada fungsi.py maka kita masukkan ke dalam main.py dengan mengetik syntax:

import fungsi

Full program pada main.py:

import tkinter as tk

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

import fungsi

def hitung\_luas():

    LS\_1 = float(txtluas\_segitiga1.get())

    LS\_2 = float(txtluas\_segitiga2.get())

    LS\_3 = float(txtluas\_segitiga3.get())

    LS\_4 = float(txtluas\_segitiga4.get())

    L = fungsi.hitung\_luas\_segitiga(LS\_1, LS\_2, LS\_3, LS\_4)

    txtluas.delete(0,END)

    txtluas.insert(END, L)

def hitung\_volume():

    LS\_1 = float(txtluas\_segitiga1.get())

    LS\_2 = float(txtluas\_segitiga2.get())

    LS\_3 = float(txtluas\_segitiga3.get())

    LS\_4 = float(txtluas\_segitiga4.get())

    a = float(txttinggi.get())

    t = float(txttinggi.get())

    T = float(txttinggi\_limas.get())

    V = fungsi.hitung\_volume\_limas(a, t, T)

    txtvolume.delete(0,END)

    txtvolume.insert(END, V)

def hitung():

    hitung\_luas()

    hitung\_volume()

# Create tkinter object

app = tk.Tk()

# Tambahkan Judul

app.title("Kalkulator Luas dan Volume Limas Segitiga")

# Windows

frame = Frame(app)

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

# Label Luas Segitiga 1

luas\_segitiga1 = Label(frame, text="Luas Segitiga 1:")

luas\_segitiga1.grid(row=0, column=0, sticky=W, padx=5, pady=5)

# Textbox Luas Segitiga 1

txtluas\_segitiga1 = Entry(frame)

txtluas\_segitiga1.grid(row=0, column=1)

# Label Luas Segitiga 2

luas\_segitiga2 = Label(frame, text="Luas Segitiga 2:")

luas\_segitiga2.grid(row=1, column=0, sticky=W, padx=5, pady=5)

# Textbox Luas Segitiga 2

txtluas\_segitiga2 = Entry(frame)

txtluas\_segitiga2.grid(row=1, column=1)

# Label Luas Segitiga 3

luas\_segitiga3 = Label(frame, text="Luas Segitiga 3:")

luas\_segitiga3.grid(row=2, column=0, sticky=W, padx=5, pady=5)

# Textbox Luas Segitiga 2

txtluas\_segitiga3 = Entry(frame)

txtluas\_segitiga3.grid(row=2, column=1)

# Label Luas Segitiga 4

luas\_segitiga4 = Label(frame, text="Luas Segitiga 4:")

luas\_segitiga4.grid(row=3, column=0, sticky=W, padx=5, pady=5)

# Textbox Luas Segitiga 4

txtluas\_segitiga4 = Entry(frame)

txtluas\_segitiga4.grid(row=3, column=1)

# Label Luas Alas

luas\_alas = Label(frame, text="Luas Alas:")

luas\_alas.grid(row=4, column=0, sticky=W, padx=5, pady=5)

# Textbox Luas Alas

txtluas\_alas = Entry(frame)

txtluas\_alas.grid(row=4, column=1)

# Label Tinggi

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

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

# Textbox Tinggi

txttinggi = Entry(frame)

txttinggi.grid(row=5, column=1)

# Label Tinggi Limas

tinggi\_limas = Label(frame, text="Tinggi Limas:")

tinggi\_limas.grid(row=6, column=0, sticky=W, padx=5, pady=5)

# Textbox Tinggi Limas

txttinggi\_limas = Entry(frame)

txttinggi\_limas.grid(row=6, column=1)

# Button

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

hitung\_button.grid(row=7, column=1, sticky=W, padx=5, pady=5)

# Output Luas

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

luas.grid(row=8, column=0, sticky=W, padx=5, pady=5)

# Textbox Luas

txtluas = Entry(frame)

txtluas.grid(row=8, column=1)

# Output Volume

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

volume.grid(row=9, column=0, sticky=W, padx=5, pady=5)

# Textbox Luas

txtvolume = Entry(frame)

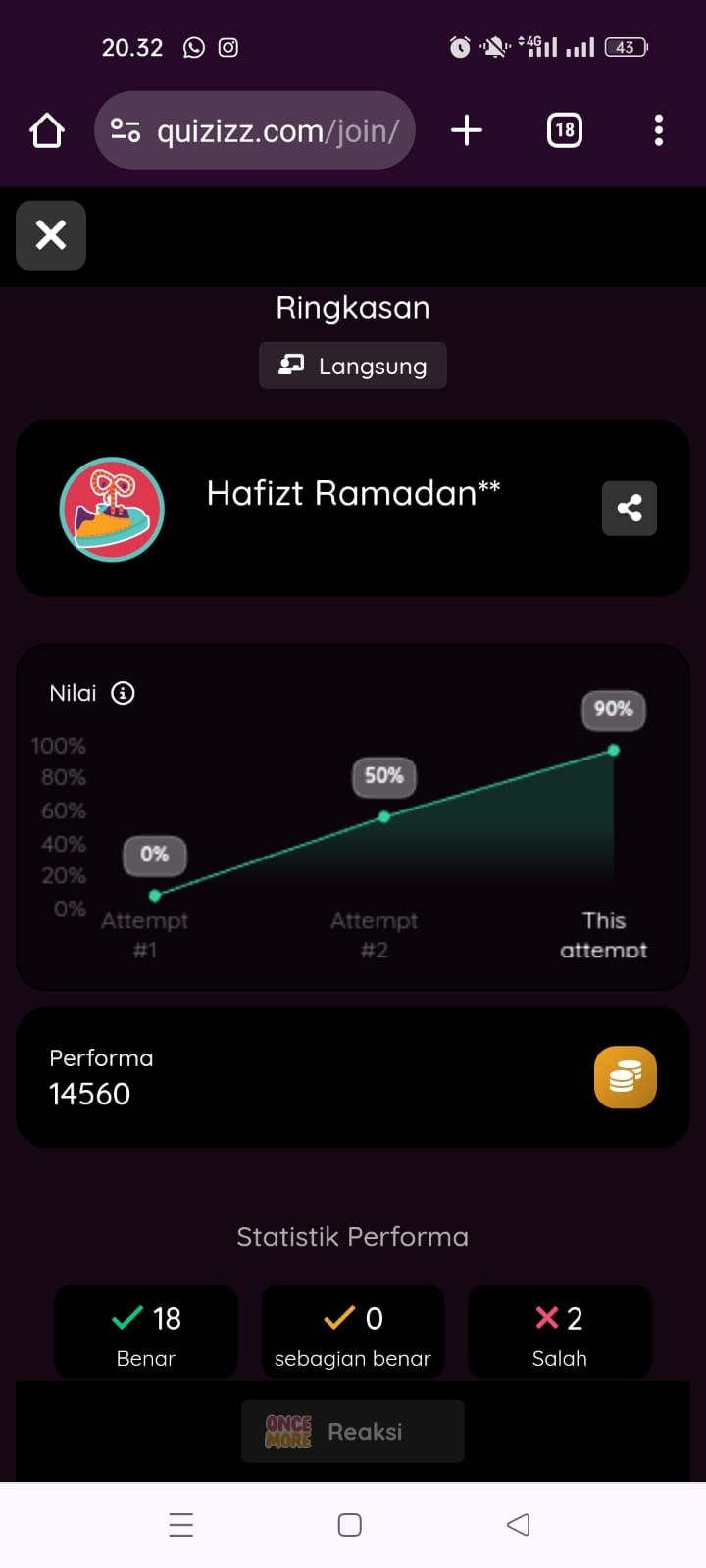
txtvolume.grid(row=9, column=1)

app.mainloop()

# Pertemuan 5 Quis

Dipertemuan 5 ini melakukan quis pada quizizz. Berikut nilai – nilai tiap anggota dari kelompok:

1. Hafizt Ramadann (220511159)



1. Haris Kurniawan (220511085)

A screenshot of a phone

Description automatically generated

1. Ibnu Alwan (220511014)

A screenshot of a phone

Description automatically generated

# Pertemuan 6 Aplikasi Bantu (Tools)

1. MP3

Untuk mengoprasikan tools mp3 bisa menggunakan library pygame atau playsound. Sebelum itu perlu menginstall librarynya terlebih dahulu, untuk penginstalan melalui cmd dengan mengetikkan pip install playsound untuk library playsound dan pip install pygame untuk library pygame. Disini kami programnnya menggunakan library pygame. Untuk langkahnya sebagai berikut:

* Import Library

import pygame.mixer

Setelah mengimport library GUI atau semacam tkinter kemudian dilanjutkan mengimport library pygame.

* Fungsi
  + Def putar\_mp3

def putar\_mp3():

    global lagu\_sedang\_diputar

    jalur\_file = filedialog.askopenfilename(filetypes=[("File MP3", "\*.mp3")])

    if jalur\_file:

        pygame.mixer.init()

        pygame.mixer.music.load(jalur\_file)

        pygame.mixer.music.play()

        lagu\_sedang\_diputar = jalur\_file

        perbarui\_keterangan\_lagu()

* Fungsi ini meminta pengguna untuk memilih file MP3 menggunakan dialog pemilihan file (filedialog.askopenfilename).
* Jika pengguna memilih file, program akan menginisialisasi mixer Pygame (pygame.mixer.init()), memuat file MP3 yang dipilih (pygame.mixer.music.load(jalur\_file)), dan memutar musik (pygame.mixer.music.play()).
* Informasi tentang lagu yang sedang diputar disimpan dalam variabel global lagu\_sedang\_diputar dan kemudian fungsi perbarui\_keterangan\_lagu() dipanggil (walaupun implementasinya tidak diberikan dalam potongan kode ini).
  + Def pause\_mp3

def pause\_mp3():

    pygame.mixer.music.pause()

* Fungsi ini menghentikan pemutaran musik sementara dengan menggunakan pygame.mixer.music.pause().
  + Def play\_mp3

def play\_mp3():

    pygame.mixer.music.unpause()

* Fungsi ini menghentikan pemutaran musik sementara dengan menggunakan pygame.mixer.music.pause().
  + Def berhenti\_mp3

def berhenti\_mp3():

    pygame.mixer.music.stop()

    pygame.mixer.quit()

    root.quit()

* Fungsi ini menghentikan pemutaran musik dan menutup mixer Pygame menggunakan pygame.mixer.music.stop() dan pygame.mixer.quit().
* Selanjutnya, fungsi ini juga memanggil root.quit(). root mungkin adalah objek Tkinter utama (jendela utama) dan quit() digunakan untuk keluar dari program.
* Program lengkap

import tkinter as tk

from tkinter import filedialog

import pygame.mixer #Menggunakan library pygame

def putar\_mp3():

    global lagu\_sedang\_diputar

    jalur\_file = filedialog.askopenfilename(filetypes=[("File MP3", "\*.mp3")])

    if jalur\_file:

        pygame.mixer.init()

        pygame.mixer.music.load(jalur\_file)

        pygame.mixer.music.play()

        lagu\_sedang\_diputar = jalur\_file

        perbarui\_keterangan\_lagu()

def pause\_mp3():

    pygame.mixer.music.pause()

def play\_mp3():

    pygame.mixer.music.unpause()

def berhenti\_mp3():

    pygame.mixer.music.stop()

    pygame.mixer.quit()

    root.quit()

def perbarui\_keterangan\_lagu():

    keterangan\_lagu.config(text="Lagu: " + lagu\_sedang\_diputar)

# Variabel global

lagu\_sedang\_diputar = ""

# Buat jendela tkinter

root = tk.Tk()

root.title("Pemutar MP3")

root.minsize(width=400, height=300)

# Buat dan letakkan label judul

label\_judul = tk.Label(root, text="Pemutar MP3", font=("Helvetica", 16))

label\_judul.pack(pady=10)

# Buat dan letakkan tombol putar

tombol\_putar = tk.Button(root, text="Putar MP3", command=putar\_mp3)

tombol\_putar.pack(pady=10)

# Buat dan letakkan tombol jeda

tombol\_jeda = tk.Button(root, text="Pause", command=pause\_mp3)

tombol\_jeda.pack(pady=10)

# Buat dan letakkan tombol lanjutkan

tombol\_lanjutkan = tk.Button(root, text="Play", command=play\_mp3)

tombol\_lanjutkan.pack(pady=10)

# Buat dan letakkan tombol keluar

tombol\_keluar = tk.Button(root, text="Keluar", command=berhenti\_mp3)

tombol\_keluar.pack(pady=10)

# Buat dan letakkan label keterangan lagu

keterangan\_lagu = tk.Label(root, text="Lagu: -", font=("Helvetica", 12))

keterangan\_lagu.pack(pady=5)

# Mulai loop tkinter

root.mainloop()

* Tampilan output

A screenshot of a computer

Description automatically generated

1. MP4

Untuk mengoprasikan tools mp4 bisa menggunakan library moviepy. Sebelum itu perlu menginstall librarynya terlebih dahulu, untuk penginstalan melalui cmd dengan mengetikkan pip install moviepy untuk library moviepy. Disini kami programnnya menggunakan library moviepy. Untuk langkahnya sebagai berikut:

* Import library

from moviepy.editor import VideoFileClip

Setelah mengimport library GUI atau semacam tkinter kemudian dilanjutkan mengimport library moviepy.

* Fungsi
  + Def browse\_file

    def browse\_file(self):

        # Membuka dialog untuk memilih file video

        file\_path = filedialog.askopenfilename(filetypes=[("Video files", "\*.mp4;\*.avi;\*.mkv")])

        # Menyimpan path file video ke dalam variable

        self.video\_path.set(file\_path)

* Fungsi ini digunakan untuk membuka dialog pemilihan file video (filedialog.askopenfilename).
* Pengguna dapat memilih file video dengan ekstensi .mp4, .avi, atau .mkv.
* Path dari file video yang dipilih disimpan dalam variabel file\_path.
* Path file video tersebut kemudian disimpan dalam variabel self.video\_path yang mungkin merupakan variabel Tkinter StringVar yang terhubung dengan antarmuka pengguna untuk menampilkan path file video yang dipilih.
  + Def play\_video

    def play\_video(self):

        # Mengambil path file video dari variable

        video\_file = self.video\_path.get()

        # Memutar video menggunakan moviepy

        video\_clip = VideoFileClip(video\_file)

        video\_clip.preview()

* Fungsi ini mengambil path file video dari variabel self.video\_path yang kemungkinan besar telah diisi oleh fungsi browse\_file.
* Membuat objek VideoFileClip menggunakan moviepy dengan path file video yang dipilih.
* Memutar video menggunakan metode preview() dari objek VideoFileClip.
  + Def stop\_video

    def stop\_video(self):

        # Do nothing for now

        pass

* Fungsi ini tidak melakukan apa-apa untuk saat ini (pass). Mungkin rencananya untuk diimplementasikan lebih lanjut jika ada kebutuhan untuk menghentikan pemutaran video.
* Program lengkap

import tkinter as tk

from tkinter import filedialog

from tkinter import ttk

from moviepy.editor import VideoFileClip

class VideoPlayerApp:

    def \_\_init\_\_(self, root):

        self.root = root

        self.root.title("Pemutar Video")

        # Variabel untuk menyimpan path file video

        self.video\_path = tk.StringVar()

        # Frame untuk menempatkan elemen-elemen GUI

        self.frame = ttk.Frame(root, padding="10")

        self.frame.grid(column=0, row=0, sticky=(tk.W, tk.E, tk.N, tk.S))

        # Button untuk memilih video

        self.browse\_button = ttk.Button(self.frame, text="Pilih Video", command=self.browse\_file)

        self.browse\_button.grid(column=0, row=0, columnspan=2, pady=10)

        # Label untuk menampilkan path file video

        self.path\_label = ttk.Label(self.frame, textvariable=self.video\_path)

        self.path\_label.grid(column=0, row=1, columnspan=2)

        # Button untuk memutar video

        self.play\_button = ttk.Button(self.frame, text="Putar", command=self.play\_video)

        self.play\_button.grid(column=0, row=2, pady=10)

        # Button untuk menghentikan pemutaran video

        self.stop\_button = ttk.Button(self.frame, text="Stop", command=self.stop\_video)

        self.stop\_button.grid(column=1, row=2, pady=10)

    def browse\_file(self):

        # Membuka dialog untuk memilih file video

        file\_path = filedialog.askopenfilename(filetypes=[("Video files", "\*.mp4;\*.avi;\*.mkv")])

        # Menyimpan path file video ke dalam variable

        self.video\_path.set(file\_path)

    def play\_video(self):

        # Mengambil path file video dari variable

        video\_file = self.video\_path.get()

        # Memutar video menggunakan moviepy

        video\_clip = VideoFileClip(video\_file)

        video\_clip.preview()

    def stop\_video(self):

        # Do nothing for now

        pass

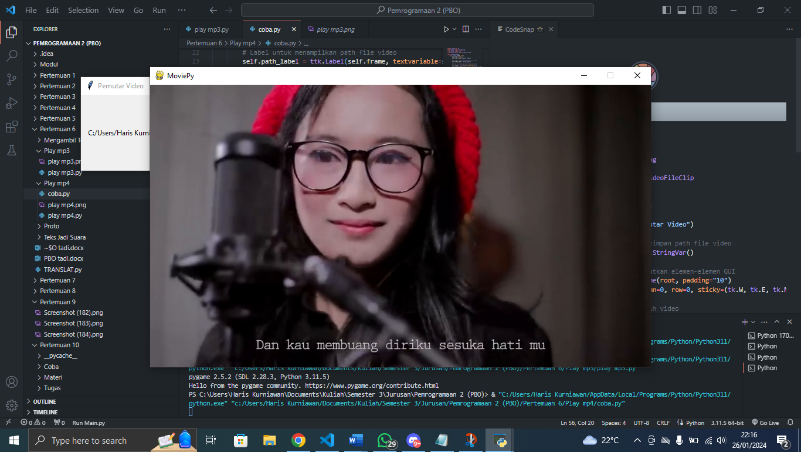
if \_\_name\_\_ == "\_\_main\_\_":

    root = tk.Tk()

    app = VideoPlayerApp(root)

    root.mainloop()

* Tampilan output



1. Teks Jadi Suara

Untuk mengoprasikan tools teks jadi suara bisa menggunakan library gtts (Google Text-to-Speech). Sebelum itu perlu menginstall librarynya terlebih dahulu, untuk penginstalan melalui cmd dengan mengetikkan pip install gtts untuk library gtts. Disini kami programnnya menggunakan library gtts. Untuk langkahnya sebagai berikut:

* Import library

from gtts import gTTS

import os

Setelah mengimport library GUI atau semacam tkinter kemudian dilanjutkan mengimport library gtts.

* Fungsi
  + Def convert\_to\_speech

def convert\_to\_speech():

    text = text\_entry.get()

    if text:

        language = language\_var.get()

        speech = gTTS(text=text, lang=language, slow=False)

        speech.save("output.mp3")

        os.system("start output.mp3")

    else:

        result\_label.config(text="Masukkan teks terlebih dahulu.")

* Fungsi kemudian mengambil nilai dari suatu variabel yang mungkin berisi kode bahasa (misalnya 'en' untuk bahasa Inggris) dari elemen antarmuka pengguna (language\_var).
* Selanjutnya, fungsi membuat objek gTTS (Google Text-to-Speech) dengan parameter teks, kode bahasa, dan opsi "slow" (kecepatan pembacaan) yang diatur ke False.
* Audio yang dihasilkan kemudian disimpan dalam file MP3 dengan nama "output.mp3" menggunakan metode save().
* Terakhir, fungsi menggunakan perintah sistem operasi (os.system) untuk memulai pemutaran file audio "output.mp3". Perintah start digunakan untuk memulai file dengan aplikasi default yang terkait dengan ekstensi file tersebut.
* Else statement: Jika teks yang diambil kosong, maka program akan mengatur teks pada elemen label tertentu (result\_label) untuk memberi tahu pengguna bahwa mereka harus memasukkan teks terlebih dahulu.
* Program lengkap

import tkinter as tk

from tkinter import ttk

from gtts import gTTS

import os

def convert\_to\_speech():

    text = text\_entry.get()

    if text:

        language = language\_var.get()

        speech = gTTS(text=text, lang=language, slow=False)

        speech.save("output.mp3")

        os.system("start output.mp3")

    else:

        result\_label.config(text="Masukkan teks terlebih dahulu.")

# Membuat GUI menggunakan Tkinter

root = tk.Tk()

root.title("Text to Speech Converter")

# Label dan Entry untuk memasukkan teks

text\_label = tk.Label(root, text="Masukkan Teks:")

text\_label.pack(pady=10)

text\_entry = tk.Entry(root, width=40)

text\_entry.pack(pady=10)

# Label untuk memilih bahasa

language\_label = tk.Label(root, text="Pilih Bahasa:")

language\_label.pack(pady=5)

languages = ['id', 'en']  # Contoh bahasa (dapat disesuaikan)

language\_var = ttk.Combobox(root, values=languages, state="readonly")

language\_var.set('id')  # Bahasa default

language\_var.pack(pady=5)

# Tombol untuk mengonversi teks menjadi suara

convert\_button = tk.Button(root, text="Konversi ke Suara", command=convert\_to\_speech)

convert\_button.pack(pady=20)

# Label untuk menampilkan pesan hasil

result\_label = tk.Label(root, text="")

result\_label.pack()

# Menjalankan aplikasi

root.mainloop()

* Tampilan output

A screenshot of a computer

Description automatically generated

1. Mengambil Teks dari Gambar

Untuk mengoprasikan tools mengambil teks dari gambar perlu install tesseract <https://github.com/UB-Mannheim/tesseract/wiki> dan install juga library gtts (Google Text-to-Speech). Sebelum itu perlu menginstall librarynya terlebih dahulu, untuk penginstalan melalui cmd dengan mengetikkan pip install gtts untuk library gtts. Disini kami programnnya menggunakan library gtts. Untuk langkahnya sebagai berikut:

* Import library

from PIL import Image, ImageTk

import pytesseract

Setelah mengimport library GUI atau semacam tkinter kemudian dilanjutkan mengimport library pytesseract dan library pillow.

* Fungsi
  + Def buka\_gambar\_dialog

    def buka\_gambar\_dialog(self):

        file\_path = filedialog.askopenfilename(filetypes=[("Image files", "\*.png;\*.jpg;\*.jpeg;\*.gif;\*.bmp")])

        if file\_path:

            self.image\_path = file\_path

            self.display\_image(file\_path)

* Fungsi ini membuka dialog pemilihan file gambar (filedialog.askopenfilename).
* Pengguna dapat memilih file gambar dengan ekstensi .png, .jpg, .jpeg, .gif, atau .bmp.
* Jika pengguna memilih file gambar, path dari file tersebut disimpan dalam variabel file\_path.
* Variabel self.image\_path kemudian diatur ke nilai file\_path, dan fungsi display\_image dipanggil untuk menampilkan gambar pada antarmuka pengguna.
  + Def ekstrak\_teks

    def ekstrak\_teks(self):

        if self.image\_path:

            image = Image.open(self.image\_path)

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

            self.text\_display.delete(1.0, tk.END)

            self.text\_display.insert(tk.END, text)

* Fungsi ini memeriksa apakah self.image\_path sudah diatur (artinya, apakah ada gambar yang dipilih sebelumnya).
* Jika self.image\_path telah diatur, gambar dibuka menggunakan Pillow (Image.open(self.image\_path)), dan teks diekstrak dari gambar menggunakan Tesseract (pytesseract.image\_to\_string(image)).
* Setelah itu, teks pada antarmuka pengguna dihapus (self.text\_display.delete(1.0, tk.END)) dan teks yang diekstrak dimasukkan ke dalam widget teks (self.text\_display.insert(tk.END, text)).
  + Def display\_image

    def display\_image(self, image\_path):

        img = Image.open(image\_path)

        img.thumbnail((300, 300))

        photo = ImageTk.PhotoImage(img)

        self.label.configure(image=photo)

        self.label.image = photo

* Fungsi ini membuka gambar dengan menggunakan Pillow (Image.open(image\_path)).
* Gambar kemudian diubah ukurannya menjadi thumbnail dengan ukuran maksimum 300x300 piksel (img.thumbnail((300, 300))).
* Objek ImageTk.PhotoImage dibuat dari gambar yang diubah ukurannya.
* Label pada antarmuka pengguna (self.label) kemudian diatur untuk menampilkan gambar menggunakan atribut image dengan nilai objek photo.
* Perhatikan bahwa objek photo harus disimpan sebagai atribut objek untuk mencegah garbage collection, yang dapat menyebabkan gambar tidak ditampilkan. Ini dilakukan dengan mengatur self.label.image ke nilai photo.
* Program lengkap

import tkinter as tk

from tkinter import ttk, filedialog

from PIL import Image, ImageTk

import pytesseract

class TextFromImageApp:

    def \_\_init\_\_(self):

        self.root = tk.Tk()

        self.root.title("Text from Image App")

        self.image\_path = None

        self.create\_widgets()

    def create\_widgets(self):

        # Frame utama

        main\_frame = ttk.Frame(self.root, padding=(20, 10))

        main\_frame.grid(column=0, row=0, sticky=(tk.W, tk.E, tk.N, tk.S))

        open\_button = ttk.Button(main\_frame, text="Buka Gambar", command=self.buka\_gambar\_dialog)

        open\_button.grid(column=0, row=0, pady=10)

        extract\_button = ttk.Button(main\_frame, text="Ekstrak Teks", command=self.ekstrak\_teks)

        extract\_button.grid(column=0, row=1, pady=5)

        # Frame untuk menampilkan gambar

        image\_frame = ttk.Frame(main\_frame, padding=(0, 10))

        image\_frame.grid(column=0, row=2)

        self.label = ttk.Label(image\_frame)

        self.label.grid(column=0, row=0)

        # Frame untuk menampilkan teks

        text\_frame = ttk.Frame(main\_frame, padding=(0, 10))

        text\_frame.grid(column=0, row=3)

        self.text\_display = tk.Text(text\_frame, height=10, width=50, wrap="word")

        self.text\_display.grid(column=0, row=0)

    def buka\_gambar\_dialog(self):

        file\_path = filedialog.askopenfilename(filetypes=[("Image files", "\*.png;\*.jpg;\*.jpeg;\*.gif;\*.bmp")])

        if file\_path:

            self.image\_path = file\_path

            self.display\_image(file\_path)

    def ekstrak\_teks(self):

        if self.image\_path:

            image = Image.open(self.image\_path)

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

            self.text\_display.delete(1.0, tk.END)

            self.text\_display.insert(tk.END, text)

    def display\_image(self, image\_path):

        img = Image.open(image\_path)

        img.thumbnail((300, 300))

        photo = ImageTk.PhotoImage(img)

        self.label.configure(image=photo)

        self.label.image = photo

    def run(self):

        self.root.mainloop()

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

    app = TextFromImageApp()

    app.run()

* Tampilan output

A screenshot of a computer

Description automatically generated

# Pertemuan 7 Konversi Suhu GUI

1. Konversi Suhu

A math equations on a white background

Description automatically generated with medium confidence

Program konversi suhu GUI:

* Import Library

Mengimport library Tkinter nya

import tkinter as tk

from tkinter import ttk

* Fungsi
* konversi\_suhu():
* Mengambil nilai suhu awal dari input pengguna.
* Mengonversi suhu awal ke dalam satuan Celcius terlebih dahulu.
* Mengonversi suhu Celcius ke unit keluaran yang dipilih.
* Menampilkan hasil konversi di label hasil\_label.
* clear\_input():
* Membersihkan nilai input suhu dan mereset label hasil\_label.
* Setting GUI
* Membua GUI:
* Membuat objek utama Tkinter (`app`).
* Menentukan judul dan ukuran jendela.
* Entry:
* Label "Suhu Awal" dan widget Entry untuk memasukkan nilai suhu awal.
* Dropdown:
* Label "Unit Suhu Awal" dan Combobox (dropdown) untuk memilih unit suhu awal.
* Label Suhu:
* Label "Suhu Konversi" dan Combobox untuk memilih unit keluaran
* Tombol:
* Tombol "Konversi" yang, saat ditekan, memanggil fungsi `konversi\_suhu`.
* Tombol "Clear" yang, saat ditekan, memanggil fungsi `clear\_input`.
* Label Hasil:
* Label "Hasil Konversi" untuk menampilkan hasil konversi.
* Menjalankan Aplikasi:
* Memanggil `app.mainloop()` untuk menjalankan loop utama aplikasi Tkinter.
* Cara Penggunaan

1. Pengguna memasukkan nilai suhu awal.
2. Memilih unit suhu awal dan unit konversi.
3. Menekan tombol "Konversi" untuk melihat hasil konversi.
4. Jika diperlukan, pengguna dapat menekan tombol "Clear" untuk mengosongkan input dan hasil konversi.

* Program dan Output
* Program

import tkinter as tk

from tkinter import ttk

def konversi\_suhu():

    try:

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

    except ValueError:

        hasil\_label.config(text="Masukkan nilai.")

        return

    unit\_input = input\_unit\_var.get()

    unit\_output = output\_unit\_var.get()

    if unit\_input == unit\_output:

        hasil\_label.config(text="Pilih unit berbeda.")

        return

    if unit\_input == "Celcius":

        suhu\_celcius = suhu\_awal

    elif unit\_input == "Fahrenheit":

        suhu\_celcius = (suhu\_awal - 32) \* 5/9

    elif unit\_input == "Kelvin":

        suhu\_celcius = suhu\_awal - 273.15

    elif unit\_input == "Reamur":

        suhu\_celcius = suhu\_awal \* 5/4

    elif unit\_input == "Rankine":

        suhu\_celcius = (suhu\_awal - 491.67) \* 5/9

    if unit\_output == "Celcius":

        hasil = suhu\_celcius

    elif unit\_output == "Fahrenheit":

        hasil = (suhu\_celcius \* 9/5) + 32

    elif unit\_output == "Kelvin":

        hasil = suhu\_celcius + 273.15

    elif unit\_output == "Reamur":

        hasil = suhu\_celcius \* 4/5

    elif unit\_output == "Rankine":

        hasil = (suhu\_celcius + 273.15) \* 9/5

    hasil\_label.config(text=f"{hasil:.2f} {unit\_output}")

def clear\_input():

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

    hasil\_label.config(text="Hasil Konversi:")

# Membuat GUI

app = tk.Tk()

app.title("Konversi Suhu")

app.geometry("270x250")

# Entry untuk nilai suhu awal

entry\_suhu\_awal\_label = tk.Label(app, text="Suhu Awal:")

entry\_suhu\_awal\_label.grid(row=0, column=0, padx=10, pady=5, sticky="W")

entry\_suhu\_awal = tk.Entry(app)

entry\_suhu\_awal.grid(row=0, column=1, padx=10, pady=5, sticky="W")

# Label Unit Awal

entry\_suhu\_awal\_label = tk.Label(app, text="Unit Suhu Awal:")

entry\_suhu\_awal\_label.grid(row=1, column=0, padx=10, pady=5, sticky="W")

# Dropdown untuk unit input

input\_unit\_var = tk.StringVar()

input\_unit\_dropdown = ttk.Combobox(app, textvariable=input\_unit\_var, values=["Celcius", "Fahrenheit", "Kelvin", "Reamur", "Rankine"])

input\_unit\_dropdown.grid(row=1, column=1, padx=10, pady=5, sticky="W")

input\_unit\_dropdown.current(0)

input\_unit\_dropdown.configure(width=10)

# Label Suhu Konversi

entry\_suhu\_konversi\_label = tk.Label(app, text="Suhu Konversi:")

entry\_suhu\_konversi\_label.grid(row=2, column=0, padx=10, pady=5, sticky="W")

# Dropdown untuk unit output

output\_unit\_var = tk.StringVar()

output\_unit\_dropdown = ttk.Combobox(app, textvariable=output\_unit\_var, values=["Celcius", "Fahrenheit", "Kelvin", "Reamur", "Rankine"])

output\_unit\_dropdown.grid(row=2, column=1, padx=10, pady=5, sticky="W")

output\_unit\_dropdown.current(1)

output\_unit\_dropdown.configure(width=10)

# Tombol untuk konversi dan reset

konversi\_button = tk.Button(app, text="Konversi", command=konversi\_suhu)

konversi\_button.grid(row=3, column=0, pady=10)

clear\_button = tk.Button(app, text="Clear", command=clear\_input)

clear\_button.grid(row=3, column=1, pady=10)

# Label untuk hasil konversi

hasil\_label = tk.Label(app, text="Hasil Konversi:")

hasil\_label.grid(row=4, column=0, pady=5)

# Menjalankan aplikasi

app.mainloop()

* Output

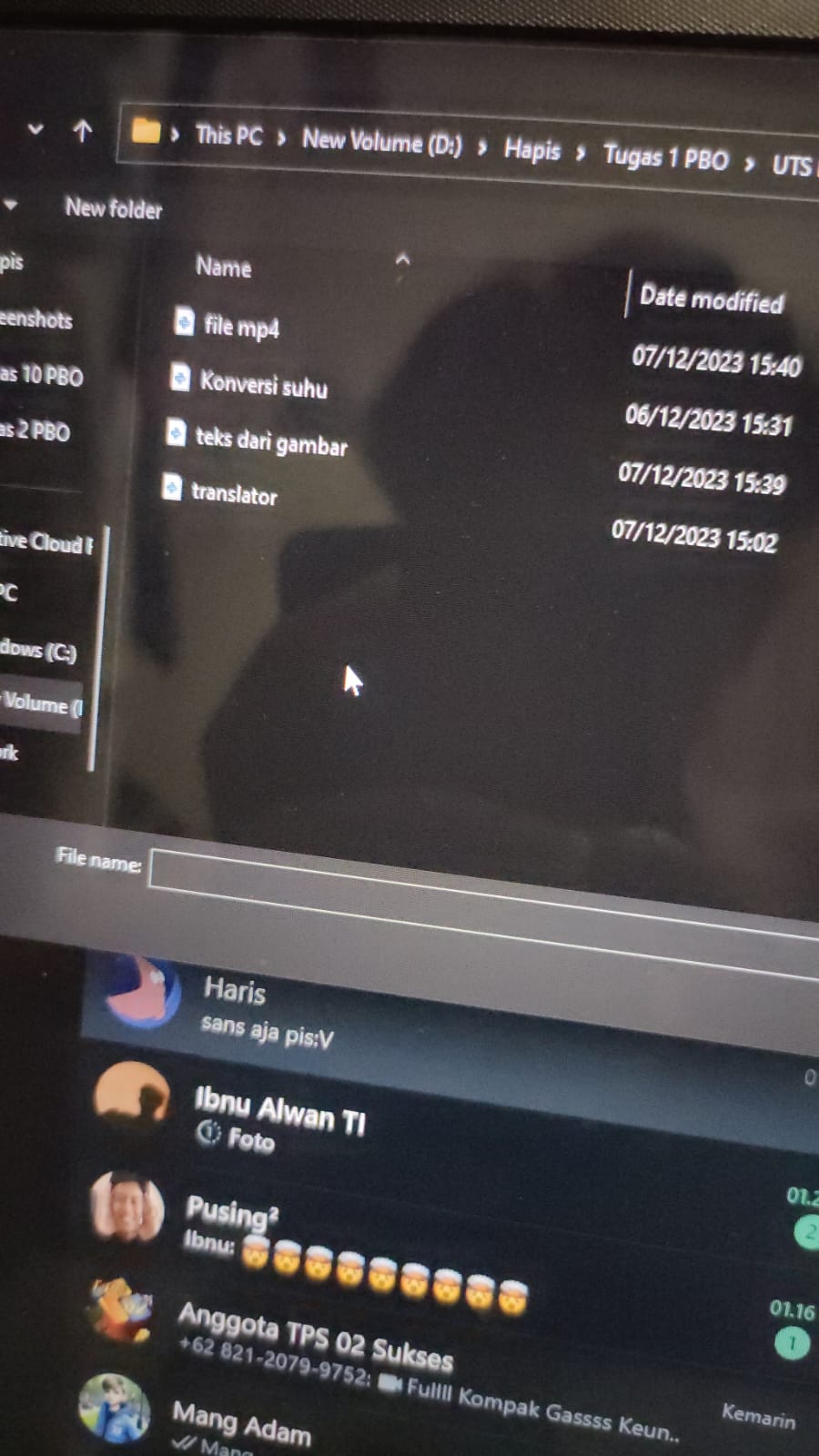
A screenshot of a computer

Description automatically generated

# Pertemuan 8 UTS

Melakukan Ujian Tengah Semester yang berisi materi dari pertemuan 1 sampai pertemuan 7.

* Hafizt Ramadan



* Haris Kurniawan

A computer screen with a paper on it

Description automatically generated

* Ibnu Alwan

A computer screen with a paper on it

Description automatically generated

# Pertemuan 9 Class dan Object (OOP)

1. Paradigma Pemrograman Berorientasi Objek

Paradigma pemrograman berorientasi objek (OOP) adalah suatu pendekatan atau cara untuk merancang dan mengorganisasi kode program berdasarkan konsep *"objek"*. Objek dalam konteks ini dapat merepresentasikan suatu entitas yang memiliki data (atribut) dan perilaku (metode). OOP membantu dalam memodelkan dunia nyata dan memungkinkan pengorganisasian kode yang lebih modular, mudah dimengerti, dan mudah dikelola.

Konsep utama dalam paradigma OOP:

1. Kelas (Class): Sebuah "kelas" adalah blueprint atau cetak biru untuk menciptakan objek. Kelas mendefinisikan atribut dan metode yang akan dimiliki oleh objek yang akan diinstansiasi dari kelas tersebut.
2. Objek (Object): Objek adalah instansi konkret dari suatu kelas. Objek memiliki atribut yang mewakili data dan metode yang mewakili perilaku.
3. Enkapsulasi (Encapsulation): Konsep ini memungkinkan penyembunyian detail implementasi suatu objek dan hanya mengekspos fungsionalitas publik. Enkapsulasi membantu dalam mencegah akses langsung ke data internal suatu objek dari luar.
4. Warisan (Inheritance): Warisan memungkinkan suatu kelas untuk "mewarisi" sifat-sifat dan perilaku dari kelas lain. Ini memungkinkan penggunaan kembali kode dan membangun hierarki kelas.
5. Polimorfisme (Polymorphism): Polimorfisme memungkinkan objek untuk dapat memiliki bentuk yang berbeda (misalnya, menggunakan metode yang sama) berdasarkan tipe atau kelas yang berbeda.

OOP membantu dalam meningkatkan keterbacaan, pemeliharaan, dan pengembangan perangkat lunak dengan memisahkan fungsionalitas ke dalam unit-unit yang lebih kecil dan terorganisir. Banyak bahasa pemrograman, termasuk Python (seperti yang digunakan dalam contoh program yang diberikan), mendukung paradigma OOP.

1. Program Konversi Suhu Celcius OOP

Implementasi Object-Oriented Programming menggunakan bahasa pemrograman Python dengan library Tkinter untuk membuat antarmuka grafis. Berikut langkah-langkah programnya:

* Membuat Class `FrmSuhu`:
* Class ini bertanggung jawab untuk menangani antarmuka grafis dan logika aplikasi konversi suhu.
* Constructor (`\_\_init\_\_`): Digunakan untuk inisialisasi objek saat objek pertama kali dibuat. Menerima dua parameter, `parent` (frame utama) dan `title` (judul jendela).
* Metode `aturKomponen`: Menyiapkan dan menata komponen-komponen GUI seperti label, entry (textbox), dan tombol.
* Metode `onHitung`: Menangani perhitungan konversi suhu berdasarkan nilai Celcius yang dimasukkan pengguna.
* Metode `onKeluar`: Menangani peristiwa penutupan aplikasi (tutup jendela).
* Membuat Objek `aplikasi`:
* Membuat objek dari class `FrmSuhu` dengan menyediakan instance dari `Tk` sebagai frame utama (`root`) dan judul jendela ("Program Konversi Suhu").
* Objek ini mempresentasikan aplikasi konversi suhu yang dapat dijalankan.
* Membuat Metode `\_\_main\_\_`:
* Menjalankan aplikasi utama.
* Membuat instance dari kelas `Tk` sebagai frame utama (`root`).
* Membuat objek `aplikasi` dari kelas `FrmSuhu` dengan menggunakan `root` sebagai frame utama dan "Program Konversi Suhu" sebagai judul.
* Memulai loop utama Tkinter dengan `root.mainloop()`.

1. Full Program dan Output/Hasil

* Program

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

class FrmSuhu:

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

        self.parent = parent

        #self.parent.geometry("400x400")

        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='Celcius:').grid(row=0, column=0,

            sticky=W, padx=5, pady=5)

        Label(mainFrame, text="Fahrenheit:").grid(row=2, column=0,

            sticky=W, padx=5, pady=5)

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

            sticky=W, padx=5, pady=5)

        Label(mainFrame, text="Kelvin:").grid(row=4, column=0,

            sticky=W, padx=5, pady=5)

        # pasang textbox

        self.txtCelcius = Entry(mainFrame)

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

        self.txtFahrenheit = Entry(mainFrame)

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

        self.txtReamur = Entry(mainFrame)

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

        self.txtKelvin = Entry(mainFrame)

        self.txtKelvin .grid(row=4, column=1, padx=5, pady=5)

        # Pasang Button

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

            command=self.onHitung)

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

    # fungsi "onHitung" berfungsi untuk menghitung luas persegi panjang

    def onHitung(self):

        C = int(self.txtCelcius.get())

        valuef = (9/5 \* C)+32

        self.txtFahrenheit.delete(0,END)

        self.txtFahrenheit.insert(END,str(valuef))

        valuer = (4/5 \* C)

        self.txtReamur.delete(0,END)

        self.txtReamur.insert(END,str(valuer))

        valuek = C + 273

        self.txtKelvin.delete(0,END)

        self.txtKelvin.insert(END,str(valuek))

    def onKeluar(self, event=None):

        # memberikan perintah menutup aplikasi

        self.parent.destroy()

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

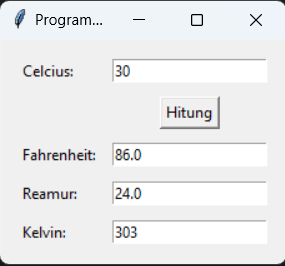
    root = Tk()

    aplikasi = FrmSuhu(root, "Program Konversi Suhu")

    root.mainloop()

* Output/Hasil

Ketika menginputkan nilai 30 derajat celcius maka output yang dihasilkan sebagai berikut:



# Pertemuan 10 Pengenalan Python Database

1. Database

Standar Interface Python untuk database adalah Python DB-API. Kebanyakan Interface database Python mematuhi standar ini.

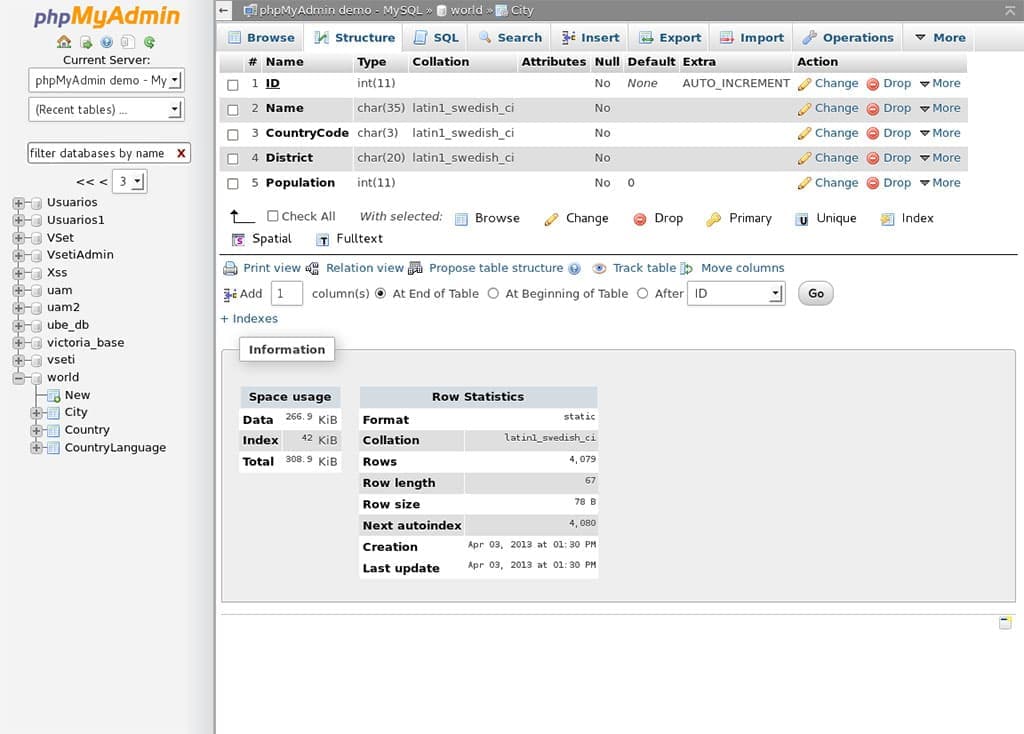
Kita bisa memilih database yang tepat. API Database Python mendukung berbagai macam server database seperti. - GadFly - mSQL - MySQL - PostgreSQL - Microsoft SQL Server 2000 - Informix - Interbase - Oracle - Sybase – SQLite. Yang kami gunakan yaitu MySQL

API DB menyediakan standar minimal untuk bekerja dengan database menggunakan struktur dan sintaks Python sedapat mungkin. API ini meliputi:

* Mengimpor modul API.
* Mendapatkan koneksi dengan database.
* Menerbitkan pernyataan SQL dan prosedur tersimpan.
* Menutup koneksi

1. Bahan dan Proses

Bahan yang diperlukan yaitu xampp. Cara install xampp cukup mengunduh pada link ini <https://www.apachefriends.org/download.html>, jika sudah mengunduh kemudian instaal dan ikuti langkah – langkah hingga selesai. Setelah terinsstal, buka xampp kemudian tekan start pada Apache dan MySQL sampai berwarna hijau. Selanjutnya untuk membuka databasenya menggunakan phpmyadmin, untuk mengaksesnya pada link ini <http://localhost/phpmyadmin> disitu bebas menambahkan. Berikut tampilan phpmyadmin:



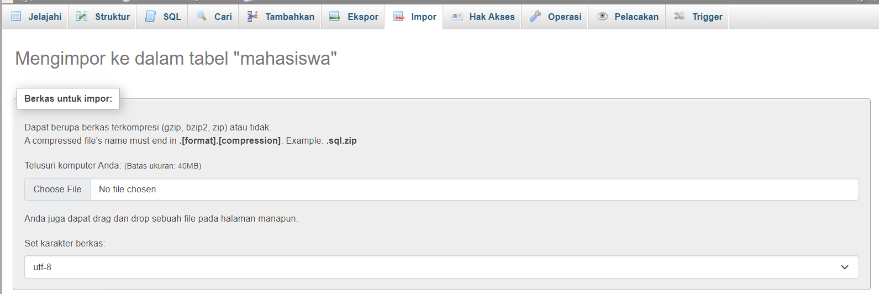
1. Cara Penggunaan Database

* Membuat Database pada phpMyAdmin
  + Buat Basis Data



Untuk nama basis data beserta nama tabel disarankan agar tidak menggunakan spasi atau bisa diganti dengan (\_). Kemudian dilanjutkan dengan mengisi struktur, sesuaikan saja dengan kebutuhannya.

* + Impor Basis Data



Klik import pada bagian menunya kemudian klik choose file untuk memilih Basis Data apa yang ingin diimport atau dimasukkan kedalam phpmyadmin, setelah itu geser kebawah tekan tombol import.

* + Ekpor Basis Data

Langkah awal yaitu pilih dahulu basis data mana yang ingin kita ekspor. Setelah itu pilih menu ekspor kemudian geser paling bawah tekan tombol ekspor, file basis data akan secara otomatis terunduh.



* Menampilkan Database Melalui CMD
  + Masuk ke directory xampp

Caranya yaitu cari lokasi terinstallnya xampp kemdian arahkan sampai ke directory bin pada folder xamppnya. Biasanya terdapat di C:\xampp\mysql\bin, agar cmd masuk kedalam directory tersebut carannya ialah cd C:\xampp\mysql\bin

* + Setting masuk phpmyadmin via cmd

Setelah cmd tersetting pada bin, kemudian ketikkan mysql -u root -p. Akan dimuncul password maka langsung klik enter saja karena tidak ada passwordnya. Setelah itu ketik *“use (sesuaikan dengan nama basis data yang ingin ditampilkan)”* sebagai contoh *‘use pbo”*, kemudian ketikkan *‘show tables;’*. Akan muncul Tabel dari basis data yang tadi diinputkan.

* + Menampilkan Tabel Basis Data

Untuk menampilkan tabel pada basis datanya yaitu dengan mengetikkan *“select \* from (nama tabel basis datanya);”*, sebagai contoh *‘select \* from matakuliah;’*. Setelah itu tabel akan muncul.

(note: ketika hanya ingin menampilkan tabel atau kategori yang diiginkan dengan cara *select \* from (nama tabel basis datanya) where (kategori)=(pilihan kategori);*)

# Pertemuan 11 Python Database

Ketika sudah memeliki database pada phpmyadmin untuk mengkoneksikan dengan python yang pertama perlu install library mysql connector dengan cara mengetikkan pip install mysql-connector-python pada CMD. Jika sudah terinstall maka perlu membuat program untuk menyambungkan database dengan python.

1. Program Menghubungkan Database dengan localhost xampp

Untuk menghubungkan database dengan python diperlukan program python terlebih dahulu dengan nama file db.py sebagai berikut:

import mysql.connector as mc

class DBConnection:

    def \_\_init\_\_(self):

        self.host = "localhost"

        self.port = 3306

        self.name = "pbo"

        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."

Penjelas program:

* Menginpor Modul MySQL Connector



Kode ini mengimpor modul mysql.connector dan memberikan alias mc untuk memudahkan penggunaan.

* Kelas DBConnection



* + Metode Inisialisasi (\_\_init\_\_)

A screen shot of a computer program

Description automatically generated

* Metode \_\_init\_\_ adalah konstruktor dari kelas. Ini menginisialisasi berbagai atribut dan membuat koneksi ke database saat objek dibuat.
* Untuk menyambungkan databasenya sesuaikan self.name = “(sesuaikan dengan nama database)”.
  + Metode Koneksi (connect):

A computer screen shot of a program

Description automatically generated

Metode connect berusaha untuk membuat koneksi ke database. Jika berhasil, itu mengatur atribut connected menjadi True dan menginisialisasi kursor untuk menjalankan kueri SQL. Jika terjadi kesalahan, atribut connected diatur menjadi False.

* + Metode Putus Koneksi (disconnect):

A screen shot of a computer code

Description automatically generated

Metode disconnect menutup koneksi database jika saat ini terhubung.

* + Metode Operasi Database:
* findOne: Menjalankan kueri SQL untuk mengambil satu baris.
* findAll: Menjalankan kueri SQL untuk mengambil semua baris.
* insert: Menjalankan kueri SQL untuk penyisipan.
* update: Menjalankan kueri SQL untuk pembaruan dengan nilai parameter.
* delete: Menjalankan kueri SQL untuk penghapusan.
* show: Menjalankan kueri SQL untuk mengambil satu baris.
* Semua metode ini memulai dengan membuat koneksi (self.connect()) sebelum menjalankan operasi database. Setelah operasi, mereka mengonfirmasi perubahan ke database dan menyimpan jumlah baris yang terpengaruh di atribut self.affected.
  + Properti Informasi Server (info):

A screenshot of a computer

Description automatically generated

Properti info memberikan informasi tentang status server. Jika terhubung, itu mengembalikan detail server; jika tidak, itu menunjukkan bahwa server sedang offline

1. Contoh Studi Kasus

Disini memakai contoh studi kasus menginput matakuliah kedalam database. Berikut agar database matakuliah bisa terpanggil kedalam program. Buat file yang bernama 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, self.\_\_kodeprodi, 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 updateBykodemk(self, kodemk):

        self.conn = mydb()

        val = (self.\_\_namamkmk, self.\_\_sks,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 deleteBykodemk(self, kodemk):

        self.conn = mydb()

        sql="DELETE FROM matakuliah WHERE kodemk='" + 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.\_\_kode\_prodi = self.result[4]

        self.conn.disconnect

        return self.result

    def getBykodemk(self, kodemk):

        a=str(kodemk)

        b=a.strip()

        self.conn = mydb()

        sql="SELECT \* FROM matakuliah WHERE kodemk='" + 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

(Note: Untuk studi kasus yang lain sesuaikan saja programnya)

1. Syntax Cek Koneksi, Menampilan, Menambah, Mengupdate, Menghapus Isi Database
2. Uji Koneksi

# Test Koneksi Database

A = DBConnection()

B = A.info

print(B)

Syntax diatas untuk mengecek apakah database sudah terkoneksi satu sama lain. Jika sudah terkoneksi ditandai dengan kalimat “*Server is running on localhost using port 3306*” pada terminal dan apabila belum terkoneksi ditandai dengan “*Server is offline*”.

1. Menampilkan

A = matakuliah()

B = A.getAllData()

print(" Menampilkan Semua Data")

print(B)

Syntax diatas untuk menampilkan isi pada database.

1. Menambah

A = matakuliah()

A.kodemk = "C1CX322T"

A.namamk = "Pemrograman 2"

A.sks = "1"

A.simpan()

B = A.getAllData()

print("Entry Data")

print(B)

Syntax diatas untuk menambahkan isi kedalam database.

1. Mengupdate

A = matakuliah()

kodemk = "C1CX322T"

A.namamk = "Pemrograman 2 (PBO)"

A.sks = "3"

A.updateBykodemk(kodemk)

B = A.getAllData()

print("Update Data")

print(B)

Syntax diatas untuk mengupdate isi pada database.

1. Menghapus

A = matakuliah()

kodemk = "C1CX322T"

A.deleteBykodemk(kodemk)

B = A.getAllData()

print("Delete Data")

print(B)

Syntax diatas untuk menghapus isi pada database.

# Pertemuan 12 Database Using Remote MySQL Server on freedb.tech

1. Freedb

FreeDB adalah Penyedia Hosting Mysql Gratis, mereka menyediakan database gratis sebesar 50MB. Langkah menggunakan freedb:

1. Membuat akun freedb atau login jika sudah memliliki akun
2. Ketika sudah masuk tampilan dashboard maka harus Create DB untuk membuat database, untuk nama database disesuaikan. Selanjutnya Creat User untuk dibuat login pada phpmyadmin, nama user disesuaikan dengan keinginan.

A screenshot of a computer

Description automatically generated

1. Jika sudah Create DB dan Create User maka tampilan berubah secara otomatis yang dimana menampilkan data – datanya.

A screenshot of a computer

Description automatically generated

1. Kemudian klik pada logo yang seperti 3 koin bertumpuk yang nantinya akan dialihkan ke tampilan login phpmyadmin. Kemudian masukkan host, username dan password yang ada pada dashboard.

A screenshot of a login box

Description automatically generated

1. Ketika sudah log in maka akan masuk kedalam phpmyadmin. Lalu buat database.
2. Program Untuk Menghubungkan Database dengan Sever freedb.tech

Untuk menghubungkan database server freedb.tech dengan python tinggal memakai file db.py pada pertemuan 10 dan cukup mengubah bagian berikut ini:

    def \_\_init\_\_(self):

        self.host = "sql.freedb.tech"

        self.port = 3306

        self.name = "freedb\_db\_contoh"

        self.user = "freedb\_nama123"

        self.password = "Ptq3Y38Dw365h&P"

        self.conn = None

        self.cursor = None

        self.result = None

        self.connected = False

        self.affected = 0

        self.connect()

Penjelasan:

* Hanya perlu mengganti self.host, self.name, self.user, self.password disesuaikan dengan yang ada di dashboard freedb

(Note: Untuk tampilan GUI bisa memakai dari studi kasus mata kuliah tadi pada pertemuan 10, disesuaikan dengan kebutuhan)

# Petemuan 13 Aplikasi Login Multi User Python-MySQL

1. Database Login Form

Disini untuk database menggunakan database local tidak memakai sever freedb, jadi perlu mengaktifkan xampp caranya seperti dipertemuan 9, lalu masuk ke localhost phpmyadmin dan buat database baru yang bernama db\_login, lalu buat tabel yang bernama users. Untuk struktur usernya sebagai berikut:

A screenshot of a computer

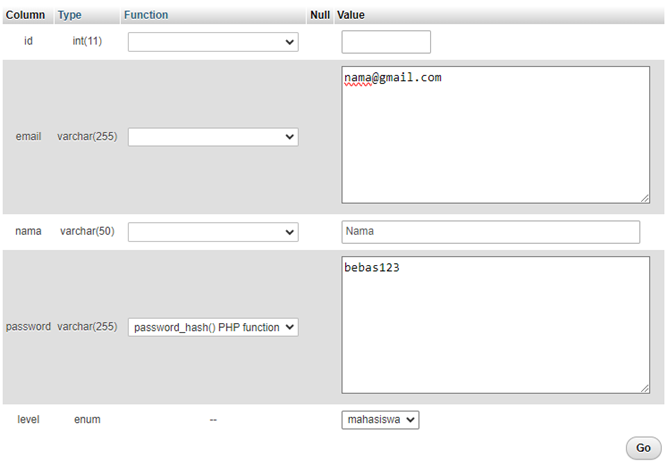
Description automatically generated

Jika sudah membuat struktur untuk membuat akun login caranya sebagai berikut:

* Masuk menu insert



* Jika sudah masuk menu insert isi data seperti contoh dibawah ini



(Note: untuk level disesuaikan dengan jabatan)

* Jika sudah mengisi tekan “Go”, lalu selesai maka akun sudah terbuat

1. Program

File yang digunakan oleh kami yaitu db.py, User.py, Dashboard.py, FrmLogin.py, FrmPersegi.py, FrmSegitiga.py, FrmLingkaran.

* Db.py

Untuk db.py nya disesusaikan nama database seperti diatas yang beranama db\_login. Program dashboard disini digunakan untuk menggabungkan beberapa file python.

* Users.py

Sebelum itu perlu menginstall library bcrypt dengen mengetikkan pip isntall bcrypt pada CMD. User.py digunakan untuk mengambil isi tabel database pada database db\_login.

* FrmLogin.py

FrmLogin.py sendiri adalah program untuk melakukan login berbasi windows serta terdapat pengecekan akun yang terdafatar pada database. Di program ini jangan lupa mengimport Users.py, jika tidak mengimportkan Users.py maka akan terjadi error dikarenakan di Users.py ini adalah isi dari databasenya dan tempat pemrosesan yang meghubungkan database.

* FrmLingkaran.py, FrmSegitiga.py, FrmPersegi.py

Program ini berisi perhitungan bangun datar berbasis windows.

* Dashboard

Dashbord ini adalah program untuk menyatukan semua program yang sudah kami dijelaskan diatas dengan cara mengimport program – program diatas. Beserta tempat pengeksekusi program – programnya.

1. Full Syntax

* Db.py

import mysql.connector as mc

class DBConnection:

    def \_\_init\_\_(self):

        self.host = 'localhost'

        self.port = 3306

        self.name = 'db\_login'

        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 database

A = DBConnection()

B = A.info

print(B)

* 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)

        b = self.cekPassword(password)

        if(a==True and b==True):

            self.\_\_loginvalid=True

        else:

            self.\_\_loginvalid=False

        val = []

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

        return val

* 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 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==True):

            self.update\_main\_window(val)

            self.parent.destroy()

        else:

            messagebox.showwarning("showwarning", "Login Gagal ")

        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()

* 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()

* 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()

* 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()

* 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. Tampilan Output

* Dashboard

A white rectangular object with black border

Description automatically generated

* FrmLogin

A screenshot of a computer

Description automatically generated

* FrmLingkaran

A screenshot of a computer

Description automatically generated

* FrmSegitiga

A screenshot of a computer

Description automatically generated

* FrmPersegi

A screenshot of a computer

Description automatically generated

# Pertemuan 14 Tugas Modul

Pada pertemuan 14 membuat modul praktikum pertemuan 1 sampai pertemuan 14