



Laporan Program Python Praktikum Lab – 13

Mata Kuliah:

Pemrograman Berorientasi Objek

Oleh:

Gerry Moeis Mahardika Dwi Putra

23091397164

2023E

<https://github.com/gerrymoeis/pbo-4>



PROGRAM STUDI MANAJEMEN INFORMATIKA FAKULTAS VOKASI

UNIVERSITAS NEGERI SURABAYA

2024



Laporan Tugas 1: Polymorphism – Suara Binatang!

Langkah	Praktikum
1.	<pre>import tkinter as tk</pre> <p>Import modul yang dibutuhkan dalam hal ini tkinter as tk.</p> <pre>class Animal: def make_sound(self): return "Miaw Miaw Ninja!"</pre> <p>Selanjutnya buat class Animal sebagai Parent Class Utama. Class Animal ini memiliki method berupa make_sound(), yang mengembalikan nilai berupa Teks Suara yang akan ditampilkan.</p>
2.	<pre># Kelas turunan Bird class Bird(Animal): def make_sound(self): return "Twit Twit" # Kelas turunan Dog class Dog(Animal): def make_sound(self): return "Anjeng!"</pre> <p>Selanjutnya, buat Class anak dari Animal. Disini instruksi modul membuat class Bird dan Dog.</p>



3.

```
root = tk.Tk()
root.title("Suara Binatang!")

# Label untuk menampilkan hasil suara
label_result = tk.Label(root, text="Klik salah satu tombol untuk mendengar suara
hewan.", font=("Arial", 20))
label_result.pack(pady=20)

label_sound = tk.Label(root, text="...", font=("Arial", 16))
label_sound.pack(pady=20)

# Fungsi untuk menampilkan suara berdasarkan jenis hewan yang dipilih
def show_sound(animal):
    label_sound.config(text=animal.make_sound())
```

Setelah itu kita siapkan untuk variable-variable yang diperlukan.
root dari modul tk sebagai window program.
label_result sebagai label text.
label_sound sebagai tempat output suara nantinya.

Lalu kita buat fungsi show_sound sebagai polymorphism dari method make_sound milik masing-masing animal yang diperoleh dari parameter animal.

4.

```
# Tombol untuk memilih Burung
button_bird = tk.Button(root, text="Burung", font=("Arial", 16), command=lambda:
show_sound(Bird()))
button_bird.pack(pady=10)

# Tombol untuk memilih Anjing
button_dog = tk.Button(root, text="Anjing", font=("Arial", 16), command=lambda:
show_sound(Dog()))
button_dog.pack(pady=10)

# Tombol untuk memilih Hewan Umum
button_animal = tk.Button(root, text="Hewan Umum", font=("Arial", 16),
command=lambda: show_sound(Animal()))
button_animal.pack(pady=10)

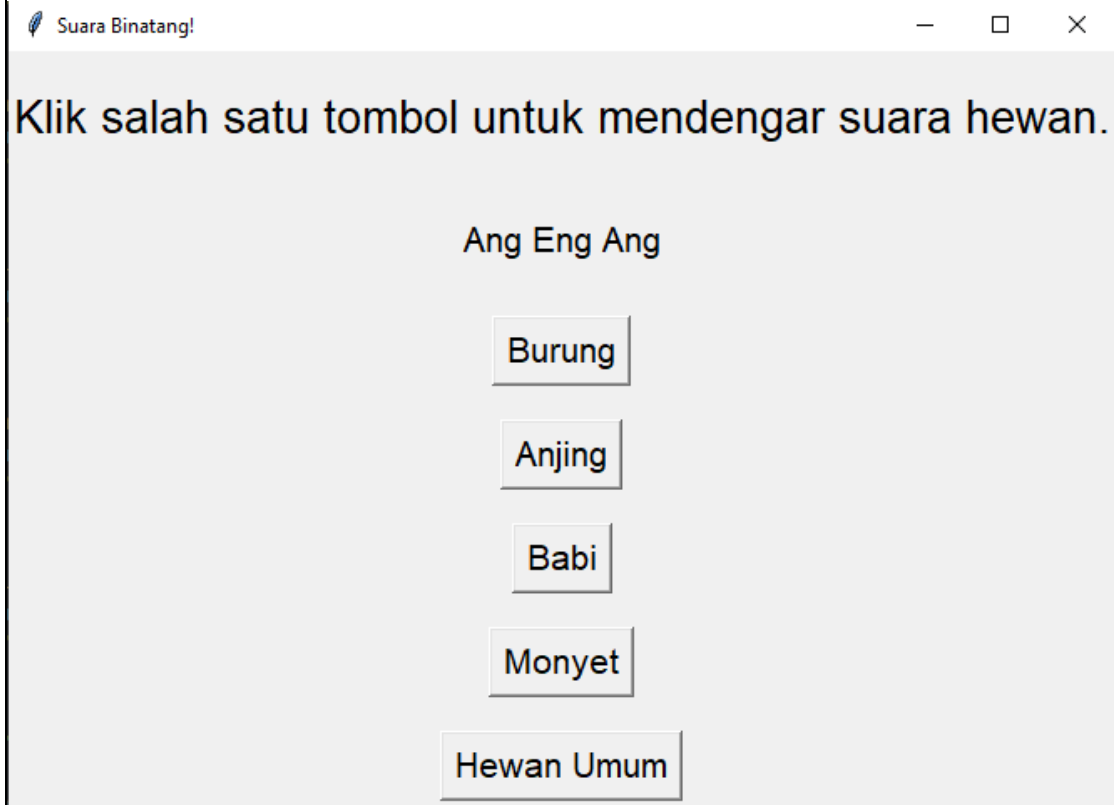
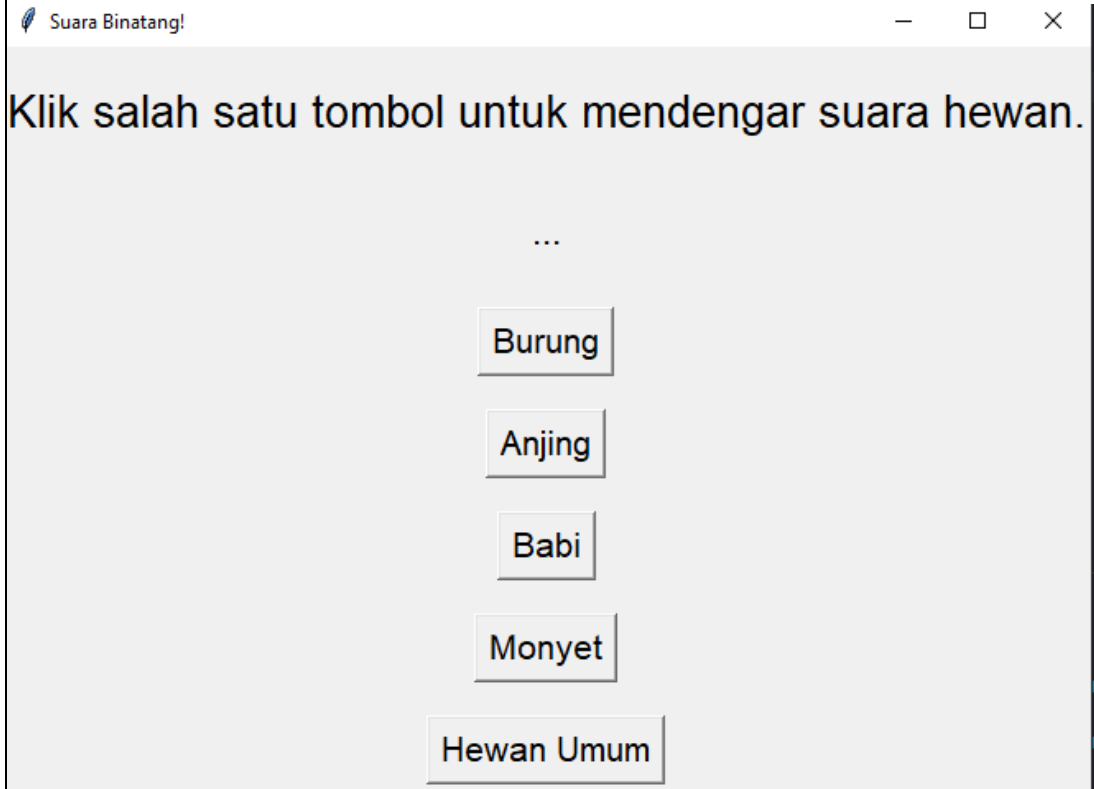
# Menjalankan aplikasi Tkinter
root.mainloop()
```



	<p>Nah langkah terakhir, disini kita buat tombol button untuk masing-masing animal. Umumnya saya memakai looping untuk membuat button-button ini agar tidak mengulang, tetapi sekarang lagi banyak tugas, jadi seyogyanya dulu.</p> <p>button disini dibuat dari tk.Button, dengan input argument, root, text, font, dan command untuk fungsi yang akan dilaksanakan bila button di klik.</p> <p>Terakhir, agar program berjalan terus selama belum di close, maka pastikan untuk menambahkan root.mainloop().</p>
Tugas	<pre># Kelas turunan Bird class Babi(Animal): def make_sound(self): return "Ang Eng Ang" # Kelas turunan Monyet class Monyet(Animal): def make_sound(self): return "U u A A"</pre> <pre># Tombol untuk memilih Babi button_pig = tk.Button(root, text="Babi", font=("Arial", 16), command=lambda: show_sound(Babi())) button_pig.pack(pady=10) # Tombol untuk memilih Monyet button_monkey = tk.Button(root, text="Monyet", font=("Arial", 16), command=lambda: show_sound(Monyet())) button_monkey.pack(pady=10)</pre> <p>Disini saya menambahkan 2 Class untuk hewan-hewan baru dengan output suaranya masing-masing.</p>



Output





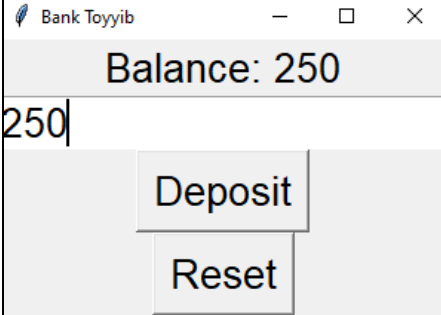
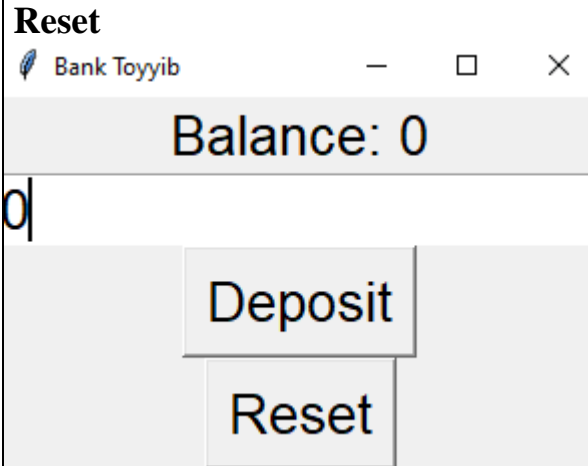
Laporan Tugas 2: Data Hiding – Bank Toyvib

Langkah	Praktikum
1.	<pre>import tkinter as tk from tkinter import messagebox</pre> <p>Untuk persiapan awal disini kita import tkinter dan messagebox.</p> <pre>class BankAccount: def __init__(self): self.__balance = 0 # Private attribute def get_balance(self): return self.__balance def deposit(self, amount): if amount > 0: self.__balance += amount else: raise ValueError("Invalid deposit amount!") def reset_balance(self): self.__balance = 0</pre> <p>Disini kita membuat satu class Utama yaitu class BankAccount. Atributnya balance dan dibuat Private dengan menambahkan underscore sebanyak 3 untuk menyembunyikan atribut ini dari akses public.</p> <p>Class BankAccount ini memiliki 3 method yaitu get_balance(), deposit(), dan reset_balance(). Pembuatan method ini mengimplementasikan konsep Data Hiding.</p>



2.	<pre>def add_balance(): try: amount = int(entry_amount.get()) account.deposit (amount) label_balance.config(text=f"Balance: {account.get_balance()}"), font=("Arial", 20)) except ValueError as e: messagebox.showerror("Error", str(e))</pre> <p>Selanjutnya kita membuat method add_balance() di luar Class. Pembuatan method ini juga memperhatikan konsep Data Hiding.</p>
3.	<pre># GUI Tkinter root = tk.Tk() root.title("Bank Toyyib") account = BankAccount() label_balance = tk.Label(root, text=f"Balance: {account.get_balance()}"), font=("Arial", 20)) label_balance.pack() entry_amount = tk.Entry(root, font=("Arial", 20)) entry_amount.pack() button_deposit = tk.Button(root, text="Deposit", command=add_balance, font=("Arial", 20)) button_deposit.pack() root.mainloop()</pre> <p>Lalu untuk langkah terakhir. Kita membuat variable-variable yang dibutuhkan. Root, pembuatan Objek account, entry_amount, dan button_deposit.</p> <p>Sisanya memakai root.mainloop() untuk memproses looping Program agar terus berjalan.</p>



Tugas	<pre>def reset_balance(self): self.__balance = 0</pre> <pre>def reset(): try: account.reset_balance() label_balance.config(text=f"Balance: {account.get_balance()}", font=("Arial", 20)) entry_amount.delete(0, tk.END) entry_amount.insert(0, 0) except ValueError as e: messagebox.showerror("Error", str(e))</pre> <pre>reset_button = tk.Button(root, text="Reset", command=reset, font=("Arial", 20)) reset_button.pack()</pre> <p>Disini saya menambahkan tombol reset dengan integrasi reset_balance di dalam Class, lalu fungsi reset diluar Class, setelah itu membuat reset_button dengan tkinter.</p>
Output	 <p>Bank Toyyib</p> <p>Balance: 250</p> <p>250 </p> <p>Deposit</p> <p>Reset</p>  <p>Reset</p> <p>Bank Toyyib</p> <p>Balance: 0</p> <p>0 </p> <p>Deposit</p> <p>Reset</p>



Laporan Tugas 3: Overriding – Kalkulasi Bentuk2

Langkah	Praktikum
1.	<pre>import tkinter as tk from tkinter import messagebox from math import pi</pre> <p>Oke langsung saja, import tkinter, messagebox, dan pi dari math.</p> <pre>class Shape: def area(self): return "Not implemented" def perimeter(self): return "Not implemented"</pre> <p>Buat class Shape, sebagai Parent Class Utama.</p>
2.	<pre>class Rectangle(Shape): def __init__(self, length, width): self.length = length self.width = width def area(self): return self.length * self.width def perimeter(self): return 2 * (self.length + self.width) class Circle(Shape): def __init__(self, radius): self.radius = radius def area(self): return pi * self.radius ** 2 def perimeter(self): return 2 * pi * self.radius</pre> <p>Selanjutnya buat 2 Class tambahan yang inherit dari Shape. Yaitu Rectangle dan Circle, dengan override method dan penambahan atribut untuk masing-masing class.</p>

<p>3.</p>	<pre> root = tk.Tk() root.title("Kalkulasi Bentuk2") shape_var = tk.StringVar(value="Rectangle") tk.Radiobutton (root, text="Rectangle", variable=shape_var, value="Rectangle", font=("Arial", 20)).pack() tk.Radiobutton (root, text="Circle", variable=shape_var, value="Circle", font=("Arial", 20)).pack() tk.Label(root, text="Parameter 1:", font=("Arial", 20)).pack() entry_param1 = tk.Entry(root, font=("Arial", 20)) entry_param1.pack() tk.Label(root, text="Parameter 2 (if Rectangle):", font=("Arial", 20)).pack() entry_param2 = tk.Entry(root, font=("Arial", 20)) entry_param2.pack() </pre> <p>Buat dan siapkan variable-variable yang diperlukan. Root, shape_var. Lalu buat UI yang diperlukan seperti pilihan Radiobutton dan Label.</p>
<p>4.</p>	<pre> def calculate(): try: shape_type = shape_var.get() if shape_type == "Rectangle": shape = Rectangle(int(entry_param1.get()), int(entry_param2.get())) elif shape_type == "Circle": shape = Circle(int (entry_param1.get())) else: raise ValueError("Invalid shape") label_result.config(text=f"Area: {shape.area()}, Perimeter: {shape.perimeter()}", font=("Arial", 20)) except ValueError as e: messagebox.showerror("Error", str(e)) </pre> <p>Setelah itu buat fungsi Calculate diluar class-class sebelumnya. Fungsi ini memiliki try except sebagai error handlingnya. Proses yang terjadi sederhananya memilih opsi kalkulasi, lalu menghitung dengan Class masing-masing.</p>



5.	<pre>button_calculate = tk.Button(root, text="Calculate", command=calculate) button_calculate.pack() label_result = tk.Label(root, text="Area:, Perimeter: ", font=("Arial", 20)) label_result.pack() root.mainloop()</pre> <p>Oke terakhir disini kita buat button dan label setelah itu kita run root.mainloop().</p>
Tugas	<pre>def reset(): try: label_result.config(text="Area:, Perimeter: ", font=("Arial", 20)) entry_param1.delete(0, tk.END) entry_param1.insert(0, 0) entry_param2.delete(0, tk.END) entry_param2.insert(0, 0) except ValueError as e: messagebox.showerror("Error", str(e)) reset_button = tk.Button(root, text="Reset", command=reset, font=("Arial", 20)) reset_button.pack()</pre> <p>Disini kita buat fungsi reset untuk mengembalikan nilai dan input kembali ke 0. Tidak lupa membuat reset_button untuk UI nya.</p>

Output	
<div><div>Kalkulasi Bentuk2</div><div><div><input checked="" type="radio"/> Rectangle</div><div><input type="radio"/> Circle</div></div><div>Parameter 1: <input type="text" value="20"/></div><div>Parameter 2 (if Rectangle): <input type="text" value="5"/></div><div>Calculate</div><div>Area: 100, Perimeter: 50</div><div>Reset</div></div>	<div><div>Kalkulasi Bentuk2</div><div><div><input type="radio"/> Rectangle</div><div><input checked="" type="radio"/> Circle</div></div><div>Parameter 1: <input type="text" value="0"/></div><div>Parameter 2 (if Rectangle): <input type="text" value="0"/></div><div>Calculate</div><div>Area:, Perimeter:</div><div>Reset</div></div>
<div><div>Kalkulasi Bentuk2</div><div><div><input type="radio"/> Rectangle</div><div><input checked="" type="radio"/> Circle</div></div><div>Parameter 1: <input type="text" value="10"/></div><div>Parameter 2 (if Rectangle): <input type="text" value="5"/></div><div>Calculate</div><div>Area: 314.1592653589793, Perimeter: 62.83185307179586</div><div>Reset</div></div>	



Laporan Tugas 4: Exception Handling – Bagi Membagi

Langkah	Praktikum
1.	<pre>import tkinter as tk from tkinter import messagebox def divide_numbers(): try: num1 = int (entry_num1.get()) num2 = int (entry_num2.get()) result = num1 / num2 label_result.config(text=f"Result: {result}", font=("Arial", 20)) except ZeroDivisionError: messagebox.showerror("Error", "Division by zero is not allowed.") except ValueError: messagebox.showerror("Error", "Please enter valid numbers.")</pre> <p>Import tkinter dan messagebox.</p> <p>Buat fungsi divide_numbers dengan try dan except untuk ExceptionHandling, spesifiknya disini kita memakai ZeroDivisionError, artinya kita tidak boleh membagi suatu angka dengan angka 0.</p>



2.

```
root = tk.Tk()
root.title("Bagi Membagi")

tk.Label(root, text="Number 1:", font=("Arial", 20)).pack()
entry_num1 = tk.Entry(root, font=("Arial", 20))
entry_num1.pack()

tk.Label(root, text="Number 2:", font=("Arial", 20)).pack()
entry_num2 = tk.Entry(root, font=("Arial", 20))
entry_num2.pack()

button_divide = tk.Button(root, text="Divide", command=divide_numbers,
font=("Arial", 20))
button_divide.pack()

label_result = tk.Label(root, text="Result: ", font=("Arial", 20))
label_result.pack()

root.mainloop()
```

Kode utama membuat root.
Label masing-masing dan juga button.
Ditutup dengan root.mainloop().

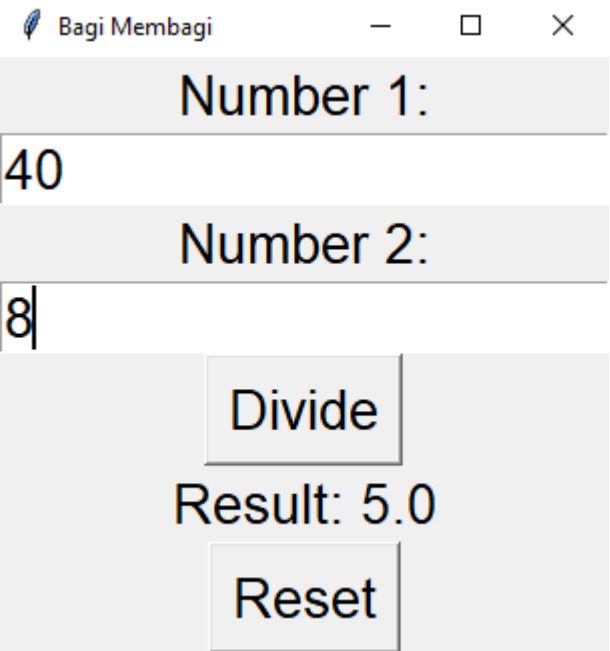
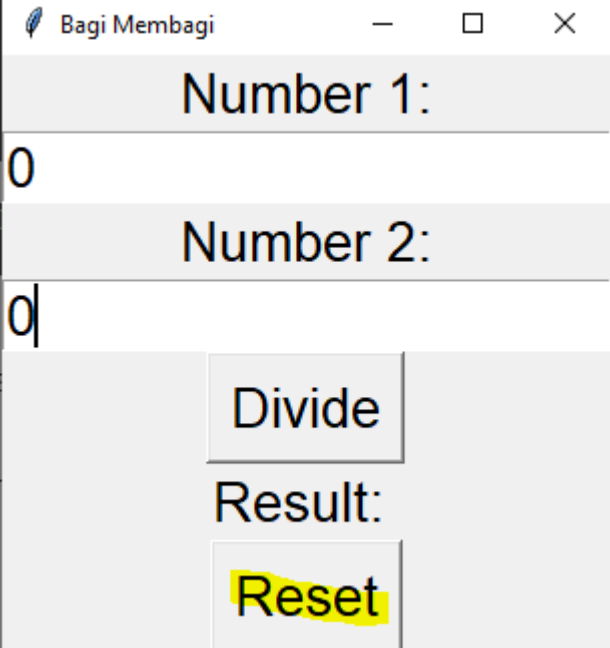


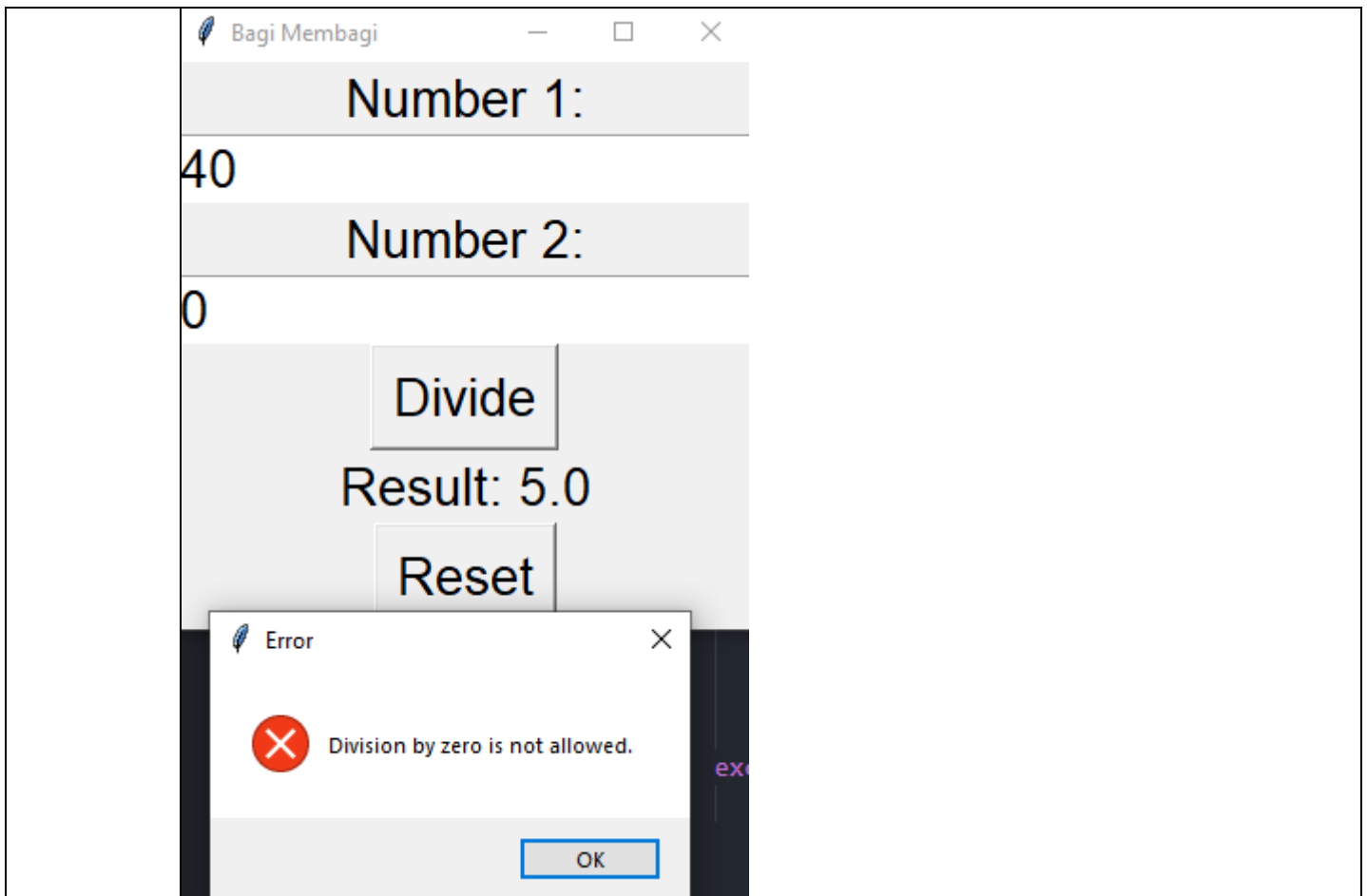
Tugas

```
def reset():  
    try:  
        label_result.config(text="Result: ", font=("Arial", 20))  
  
        entry_num1.delete(0, tk.END)  
        entry_num1.insert(0, 0)  
  
        entry_num2.delete(0, tk.END)  
        entry_num2.insert(0, 0)  
    except ValueError as e:  
        messagebox.showerror("Error", str(e))  
  
reset_button = tk.Button(root, text="Reset", command=reset, font=("Arial", 20))  
reset_button.pack()
```

Buat reset_button yang memanggil fungsi reset
Fungsi reset ini menggunakan Error Handling juga. Dengan proses utamanya untuk mengembalikan nilai entry dan label ke 0 atau kosong.



Output	
	





KEMENTRIAN PENDIDIKAN, KEBUDAYAAN,
RISET, DAN TEKNOLOGI
UNIVERSITAS NEGERI SURABAYA
Kampus Unesa 1, jalan Ketintang Surabaya 60231
Laman: <https://vokasi.unesa.ac.id/> E-mail: vokasi@unesa.ac.id

--#BeraksiBerprestasiBersinergi--