

LAPORAN PRAKTIKUM

PEMROGRAMAN BERORIENTASI OBJEK (PBO) – [TUGAS 3]



Disusun Oleh
Reza Chairul Manam
120140086

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI SUMATERA
2025**

1 Soal Nomor 1

1.1 Input Soal

Di tugas ini, kalian akan membuat kalkulator sederhana yang menggunakan beberapa **Dunder Method** untuk melakukan operasi seperti:

- +, -, *, /, ^ (eksponen), dan log.

1.2 Source Code

```
# Tugas-1:
import math

class Kalkulator: # Kelas Kalkulator
    def __init__(self, nilai):
        self.nilai = nilai

    def __add__(self, other): # Method untuk menambahkan
        nilai
        return Kalkulator(self.nilai + other.nilai)

    def __sub__(self, other): # Method untuk mengurangi
        nilai
        return Kalkulator(self.nilai - other.nilai)

    def __mul__(self, other): # Method untuk mengalikan
        nilai
        return Kalkulator(self.nilai * other.nilai)

    def __truediv__(self, other): # Method untuk membagi
        nilai
        if other.nilai == 0:
            raise ValueError("Pembagian dengan nol tidak
                               diperbolehkan!")
        return Kalkulator(self.nilai / other.nilai)

    def __pow__(self, other): # Method untuk menghitung
        nilai pangkat
        return Kalkulator(self.nilai ** other.nilai)

    def log(self, base=10): # Method untuk menghitung nilai
        logaritma
```

```

        if self.nilai <= 0:
            raise ValueError("Logaritma hanya bisa
                               diterapkan pada bilangan positif!")
        return Kalkulator(math.log(self.nilai, base))

    def __str__(self): # Method untuk mengubah nilai
                        menjadi string
        return str(self.nilai)

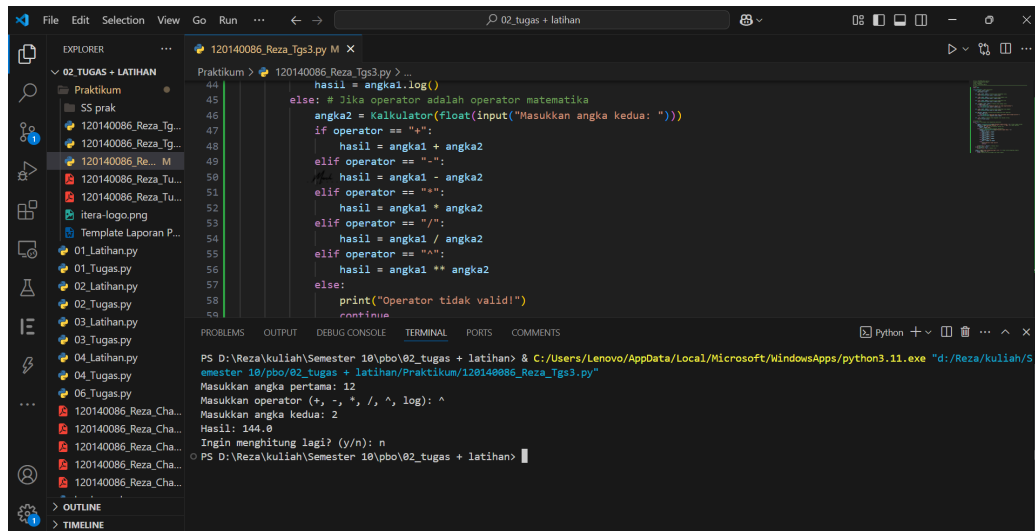
# Program Utama
while True: # Perulangan untuk menghitung kalkulator
    try:
        angka1 = Kalkulator(float(input("Masukkan angka
                                           pertama: "))) # Input angka pertama
        operator = input("Masukkan operator (+, -, *, /, ^,
                           log): ") # Input operator
        if operator == "log": # Jika operator adalah log
            hasil = angka1.log()
        else: # Jika operator adalah operator matematika
            angka2 = Kalkulator(float(input("Masukkan angka
                                              kedua: ")))
            if operator == "+":
                hasil = angka1 + angka2
            elif operator == "-":
                hasil = angka1 - angka2
            elif operator == "*":
                hasil = angka1 * angka2
            elif operator == "/":
                hasil = angka1 / angka2
            elif operator == "^":
                hasil = angka1 ** angka2
            else:
                print("Operator tidak valid!")
                continue

        print(f"Hasil: {hasil}") # Output hasil
    except ValueError as e:
        print(f"Error: {e}") # Output error

    lanjut = input("Ingin menghitung lagi? (y/n): ") #
    # Input untuk mengulang program
    if lanjut.lower() != "y":
        break # Keluar dari perulangan jika input tidak y

```

1.3 Output Hasil



The screenshot shows a Visual Studio Code editor with a Python file named `120140086_Reza_Tgs3.py`. The code implements a calculator with a `Kalkulator` class and a `log` function. The terminal output shows the program running and calculating $12 \times 12 = 144$.

```
44     hasil = angka1.log()
45 else: # Jika operator adalah operator matematika
46     angka2 = Kalkulator(float(input("Masukkan angka kedua: ")))
47     if operator == "+":
48         hasil = angka1 + angka2
49     elif operator == "-":
50         hasil = angka1 - angka2
51     elif operator == "*":
52         hasil = angka1 * angka2
53     elif operator == "/":
54         hasil = angka1 / angka2
55     elif operator == "**":
56         hasil = angka1 ** angka2
57     else:
58         print("Operator tidak valid!")
59         continue
```

```
PS D:\Reza\kuliah\Semester 10\pbo\02_tugas + latihan> & C:/Users/Lenovo/AppData/Local/Microsoft/WindowsApps/python3.11.exe "d:/Reza/kuliah/S
semester 10/pbo/02_tugas + latihan/Praktikum/120140086_Reza_Tgs3.py"
Masukkan angka pertama: 12
Masukkan operator (+, -, *, /, ^, log): ^
Masukkan angka kedua: 2
Hasil: 144.0
Ingin menghitung lagi? (y/n): n
PS D:\Reza\kuliah\Semester 10\pbo\02_tugas + latihan>
```

1.4 Penjelasan

Program ini merupakan implementasi dari kelas `Kalkulator` yang dapat melakukan operasi matematika dasar dengan konsep *operator overloading* dalam pemrograman berorientasi objek (OOP).

Kelas `Kalkulator` memiliki atribut nilai sebagai angka yang akan dioperasikan, serta berbagai metode seperti `__add__()`, `__sub__()`, `__mul__()`, `__truediv__()`, dan `__pow__()` untuk menangani operasi penjumlahan, pengurangan, perkalian, pembagian, dan perpangkatan. Selain itu, terdapat metode `log()` untuk menghitung logaritma dengan basis default 10.

Program utama menjalankan perulangan untuk menerima input dari pengguna. Pengguna memasukkan dua angka dan memilih operator matematika yang akan digunakan. Jika operator adalah `log`, hanya satu angka yang digunakan, sedangkan operator lainnya membutuhkan dua angka. Program menangani kesalahan seperti pembagian dengan nol atau input yang tidak valid menggunakan `try-except`. Setelah menampilkan hasil perhitungan, pengguna diberikan opsi untuk melakukan perhitungan lain atau keluar dari program.

2 Soal Nomor 2

2.1 Input Soal

Di tugas ini, kalian akan mensimulasikan pewarisan golongan darah anak dari orang tua. Untuk tugas ini, kalian akan membuat 3 kelas:

- **Father**
- **Mother**
- **Child**

Kelas **Father** dan **Mother** akan memiliki properti `blood_types`, yang nantinya akan diinput oleh pengguna. Kelas **Child** akan menerima properti tersebut, memilih salah satu alel secara acak dari setiap orang tua, dan menentukan golongan darahnya. Probabilitas pemilihan alel adalah **50-50** untuk ayah dan ibu.

2.2 Source Code

```
# Tugas 2:
import random # Import modul random

class Father: # Kelas Father
    def __init__(self, blood_type):
        self.blood_type = blood_type

class Mother: # Kelas Mother
    def __init__(self, blood_type):
        self.blood_type = blood_type

class Child: # Kelas Child
    def __init__(self, father, mother):
        self.father_blood = father.blood_type
        self.mother_blood = mother.blood_type
        self.blood_type = self.determine_blood_type() #
            Memanggil method determine_blood_type

    def determine_blood_type(self): # Method untuk
        menentukan golongan darah anak
        blood_map = { # Dictionary untuk menentukan
            golongan darah anak
```

```

        ('A', 'A'): 'A', ('A', 'B'): 'AB', ('A', 'O'):
            'A', ('A', 'AB'): random.choice(['A', 'AB'])
        ,
        ('B', 'B'): 'B', ('B', 'O'): 'B', ('B', 'AB'):
            random.choice(['B', 'AB']),
        ('O', 'O'): 'O', ('O', 'AB'): random.choice(['A
            ', 'B']),
        ('AB', 'AB'): random.choice(['A', 'B', 'AB'])
    }

    alel_father = random.choice(self.father_blood) #
        Memilih alel golongan darah ayah
    alel_mother = random.choice(self.mother_blood) #
        Memilih alel golongan darah ibu
    return blood_map.get((alel_father, alel_mother),
        blood_map.get((alel_mother, alel_father))) #
        Menentukan golongan darah anak berdasarkan alel
        ayah dan ibu

def display_info(self): # Method untuk menampilkan
    informasi golongan darah
    print(f"Golongan darah ayah: {self.father_blood}")
    print(f"Golongan darah ibu: {self.mother_blood}")
    print(f"Golongan darah anak: {self.blood_type}")

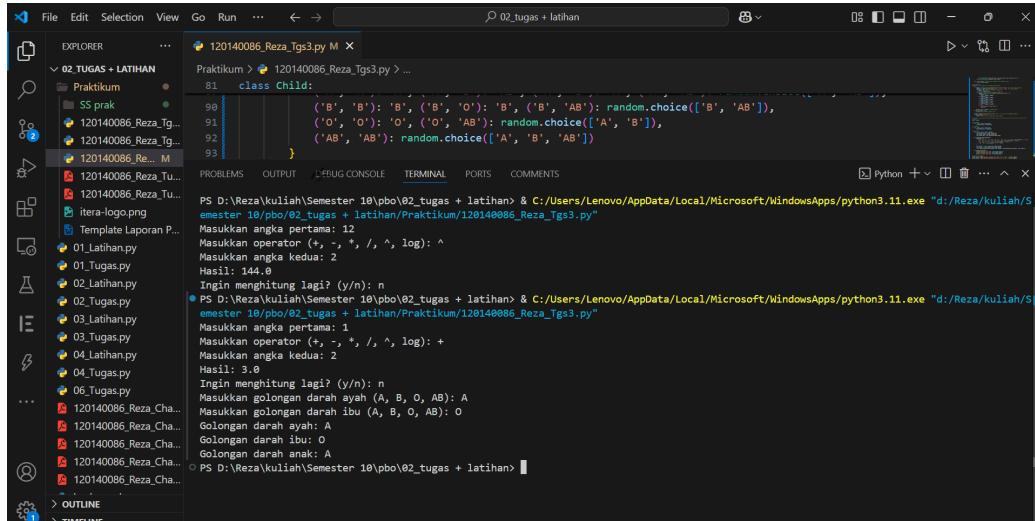
# Input dari pengguna
father_blood = input("Masukkan golongan darah ayah (A, B, O
    , AB): ").strip().upper() # Input golongan darah ayah
mother_blood = input("Masukkan golongan darah ibu (A, B, O,
    AB): ").strip().upper() # Input golongan darah ibu

father = Father(father_blood) # Objek Father
mother = Mother(mother_blood) # Objek Mother
child = Child(father, mother) # Objek Child

child.display_info() # Output informasi golongan darah anak

```

2.3 Output Hasil



The screenshot shows a VS Code editor with a file explorer on the left, a code editor in the center, and a terminal at the bottom. The code editor displays a Python script named `120140086_Reza_Tgs3.py` with a `class Child:` definition. The terminal shows the execution of the script, which prompts the user for input and displays the output.

```
class Child:
    ('B', 'B'): 'B', ('B', 'O'): 'B', ('B', 'AB'): random.choice(['B', 'AB']),
    ('O', 'O'): 'O', ('O', 'AB'): random.choice(['A', 'B']),
    ('AB', 'AB'): random.choice(['A', 'B', 'AB'])

PS D:\Reza\kuliah\Semester 10\pbo\02_tugas + latihan> & C:/Users/Lenovo/AppData/Local/Microsoft/WindowsApps/python3.11.exe "d:/Reza/kuliah/S
Semester 10/pbo/02_tugas + latihan/Praktikum/120140086_Reza_Tgs3.py"
Masukkan angka pertama: 12
Masukkan operator (+, -, *, /, ^, log): ^
Masukkan angka kedua: 2
Hasil: 144.0
Ingin menghitung lagi? (y/n): n
PS D:\Reza\kuliah\Semester 10\pbo\02_tugas + latihan> & C:/Users/Lenovo/AppData/Local/Microsoft/WindowsApps/python3.11.exe "d:/Reza/kuliah/S
Semester 10/pbo/02_tugas + latihan/Praktikum/120140086_Reza_Tgs3.py"
Masukkan angka pertama: 1
Masukkan operator (+, -, *, /, ^, log): +
Masukkan angka kedua: 2
Hasil: 3.0
Ingin menghitung lagi? (y/n): n
Masukkan golongan darah ayah (A, B, O, AB): A
Masukkan golongan darah ibu (A, B, O, AB): O
Golongan darah ayah: A
Golongan darah ibu: O
Golongan darah anak: A
PS D:\Reza\kuliah\Semester 10\pbo\02_tugas + latihan>
```

2.4 Penjelasan

Golongan darah manusia ditentukan oleh kombinasi alel dari kedua orang tua, di mana setiap individu mewarisi satu alel dari ayah dan satu dari ibu. Kombinasi ini mengikuti pola pewarisan yang dikendalikan oleh tiga alel utama: **A**, **B**, dan **O**, dengan alel **A** dan **B** bersifat dominan terhadap **O**. Oleh karena itu, pasangan dengan golongan darah tertentu dapat memiliki anak dengan berbagai kemungkinan golongan darah, tergantung pada genotipe mereka. Misalnya, individu dengan golongan darah **A** bisa memiliki genotipe **AA** atau **AO**, sementara individu dengan golongan darah **AB** pasti memiliki genotipe **AB**. Dengan demikian, pewarisan golongan darah dapat diprediksi menggunakan prinsip dasar genetika.

3 Lampiran

Pada bagian ini disertakan tautan yang berkaitan dengan tugas praktikum. Tautan GitHub berisi kode sumber yang telah dibuat selama praktikum, sedangkan tautan Overleaf merupakan tempat penyusunan laporan ini.

Repository GitHub Tugas: Berikut adalah tautan menuju repository GitHub yang berisi seluruh kode sumber tugas praktikum:

<https://github.com/rezachairul/Latihan-PBO-2025/tree/main/Praktikum>

Dokumen Laporan di Overleaf: Untuk melihat laporan ini secara langsung di Overleaf, gunakan tautan berikut:

<https://www.overleaf.com/read/bxrvfsgybfry#3c9404>