

# **PRAKTIKUM PEMROGRAMAN KOMPUTER**

## **PERTEMUAN XI**

### **PEMROGRAMAN MODULAR**



Disusun oleh:

Nama : Ahmad Fawwas Dzulfahmi  
NIM : 25/567941/SV/27316  
Kelas : RI1A2  
Dosen Pengampu : Ir. Yuris Mulya Saputra, S.T., M.Sc., Ph.D., IPM.

**PROGRAM STUDI D-IV TEKNOLOGI REKAYASA INTERNET**

**DEPARTEMEN TEKNIK ELEKTRO DAN INFORMATIKA**

**SEKOLAH VOKASI**

**UNIVERSITAS GADJAH MADA**

**2025**

## DAFTAR ISI

<b>DAFTAR ISI.....</b>	<b>1</b>
<b>DAFTAR GAMBAR.....</b>	<b>2</b>
<b>BAB I PENDAHULUAN.....</b>	<b>3</b>
<b>1.1    Latar Belakang.....</b>	<b>3</b>
<b>1.3    Tujuan.....</b>	<b>4</b>
<b>BAB II PEMBAHASAN.....</b>	<b>5</b>
<b>2.1    Tugas 1.....</b>	<b>5</b>
<b>2.2    Tugas 2.....</b>	<b>7</b>
<b>2.3    Tugas 3.....</b>	<b>9</b>
<b>2.4    Tugas 4.....</b>	<b>11</b>
<b>BAB III PENUTUP.....</b>	<b>15</b>
<b>3.1    Kesimpulan.....</b>	<b>15</b>
<b>DAFTAR PUSTAKA.....</b>	<b>16</b>

## DAFTAR GAMBAR

Tugas 1.1 : contoh penggunaan def .....	<b>Error! Bookmark not defined.</b>
Tugas 1.2 : contoh penggunaan parameter .....	5
Tugas 1.3 : contoh isi fungsi .....	5
Tugas 1.4 contoh pemanggilan fungsi .....	<b>Error! Bookmark not defined.</b>
Tugas 1.5 : contoh penggunaan docstring.....	<b>Error! Bookmark not defined.</b>
Tugas 2.1 : source code show_data() .....	7
Tugas 2.2 : source code insert_data() .....	8
Tugas 2.3 : source code edit_data() .....	<b>Error! Bookmark not defined.</b>
Tugas 2.4 :source code delete_data().....	<b>Error! Bookmark not defined.</b>
Tugas 2.5 : source code show_menu().....	<b>Error! Bookmark not defined.</b>
Tugas 4.1 : input pengguna, nama dan NIM.....	13
Tugas 4.2 : source code fungsi perhitungan matematika .....	14
Tugas 4.3 :source code fungsi menu.....	14

# **BAB I**

## **PENDAHULUAN**

### **1.1 Latar Belakang**

Pemrograman modular adalah pendekatan dalam pemrograman yang membagi sebuah program besar menjadi bagian-bagian kecil yang disebut modul. Setiap modul memiliki fungsi atau tugas tertentu yang berdiri sendiri, namun tetap dapat berinteraksi dengan modul lain melalui antarmuka yang telah ditentukan. Pendekatan ini bertujuan untuk membuat program lebih terstruktur, mudah dikembangkan, diuji, serta dipelihara. Prinsip utama dalam pemrograman modular meliputi kohesi tinggi (setiap modul hanya melakukan satu jenis tugas), kopling longgar (modul tidak saling bergantung secara berlebihan), serta penerapan abstraksi dan penyembunyian informasi agar detail implementasi suatu modul tidak mengganggu modul lain.

Keuntungan dari pemrograman modular sangat banyak, antara lain mempermudah pembagian kerja antar pengembang, meningkatkan keterbacaan kode, mempermudah proses pengujian dan debugging, serta memungkinkan penggunaan ulang (reusabilitas) kode di proyek lain. Selain itu, sistem yang dibangun secara modular juga lebih fleksibel dan mudah dikembangkan di masa depan. Namun, pendekatan ini juga memiliki tantangan, seperti menentukan batas yang tepat antar modul agar tidak terlalu besar atau terlalu kecil, serta mengelola interaksi antar modul agar tidak menimbulkan ketergantungan yang kompleks.

Dalam praktiknya, konsep modular dapat dilihat pada pemrograman menggunakan Python, seperti pada program yang terdiri dari beberapa file terpisah—misalnya `produk.py`, `transaksi.py`, dan `main.py`. Setiap file berfungsi sebagai modul yang memiliki tanggung jawab spesifik, seperti mengelola data produk, menghitung transaksi, atau menampilkan antarmuka pengguna. Dengan cara ini, jika terjadi perubahan pada logika diskon atau data produk, cukup mengubah modul terkait tanpa perlu mengubah keseluruhan program. Pendekatan ini membuat aplikasi lebih mudah dipelihara, dikembangkan, dan diperluas di masa depan. Secara keseluruhan, pemrograman modular merupakan praktik penting yang mendukung pengembangan perangkat lunak yang efisien, terorganisasi, dan berkelanjutan.

### **1.2 Rumusan Masalah**

Adapun Rumusan Masalah Sebagai Berikut:

1. Mampu membuat sebuah program operasi matematika menggunakan modul.
2. Mampu membuat sebuah program konversi suhu menggunakan modul.
3. Mampu membuat sebuah program sederhana untuk mengelola data mahasiswa.
4. Membuat sebuah program sederhana aplikasi penjualan.

### **1.3 Tujuan**

Adapun Tujuan dari Laporan Praktikum sebagai berikut:

1. Mahasiswa mampu menjelaskan konsep pemrograman modular termasuk fungsi, modul, dan paket.
2. Mahasiswa mampu membagi sebuah program besar menjadi beberapa modul yang reusable.
3. Mahasiswa dapat membuat, mengimpor, dan mengeksekusi modul serta paket Python dengan benar

## BAB II

### PEMBAHASAN

#### 2.1 Tugas 1

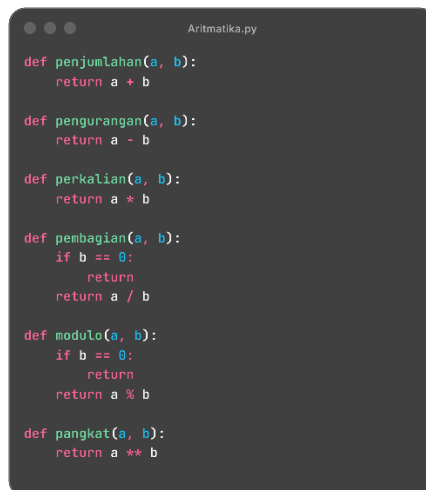
##### 1. Membuat Modul Operasi Matematika

Buat sebuah modul Python bernama `aritmatika.py` yang berisi enam fungsi:

- Penjumlahan
- Pengurangan
- Perkalian
- Pembagian
- Modulo (siswa bagi)
- Pangkat

Kemudian buat program utama `main.py` yang mengimpor modul tersebut, Menerima input angka dari pengguna, dan menampilkan hasil dari semua operasi menggunakan modul `aritmatika`.

Jawaban :



```
Aritmatika.py

def penjumlahan(a, b):
    return a + b

def pengurangan(a, b):
    return a - b

def perkalian(a, b):
    return a * b

def pembagian(a, b):
    if b == 0:
        return
    return a / b

def modulo(a, b):
    if b == 0:
        return
    return a % b

def pangkat(a, b):
    return a ** b
```

*Tugas 1.1 : source code aritmatika*



```
main.py

from aritmatika import *

a = float(input("Angka pertama: "))
b = float(input("Angka kedua: "))

print(f"Penjumlahan : {penjumlahan(a, b)}")
print(f"Pengurangan : {pengurangan(a, b)}")
print(f"Perkalian : {perkalian(a, b)}")
print(f"Pembagian : {pembagian(a, b)}")
print(f"Modulo : {modulo(a, b)}")
print(f"Pangkat : {pangkat(a, b)}")
```

*Tugas 1.2 : contoh penggunaan aritmatika*

Program di atas terdiri dari dua bagian utama, yaitu modul aritmatika.py dan program utama main.py.

Modul aritmatika.py berfungsi sebagai tempat penyimpanan kumpulan fungsi operasi matematika dasar. Di dalamnya terdapat enam fungsi, yaitu penjumlahan(), pengurangan(), perkalian(), pembagian(), modulo(), dan pangkat(). Setiap fungsi menerima dua parameter (a dan b) dan mengembalikan hasil perhitungan sesuai jenis operasinya. Misalnya, fungsi penjumlahan(a, b) mengembalikan nilai  $a + b$ , sedangkan pembagian(a, b) dilengkapi dengan pemeriksaan agar tidak terjadi kesalahan pembagian dengan nol.

Sementara itu, file main.py berperan sebagai program utama yang menggunakan modul tersebut. Program ini mengimpor semua fungsi dari aritmatika.py menggunakan perintah `from aritmatika import *`. Kemudian, program meminta dua input angka dari pengguna dan menyimpannya dalam variabel a dan b. Setelah itu, program menampilkan hasil dari setiap operasi matematika dengan memanggil fungsi yang sesuai dari modul aritmatika

```
● PS C:\Users\Kaizen\Documents\1. Praktikum UGM\PPROKOM XI Ahmad Fawwas Dzulf  
XI Ahmad Fawwas Dzulfahmi/Tugas 1/main.py"  
Angka pertama: 15  
Angka kedua: 20  
Penjumlahan : 35.0  
Pengurangan : -5.0  
Perkalian   : 300.0  
Pembagian   : 0.75  
Modulo      : 15.0  
Pangkat     : 3.3252567300796506e+23  
○ PS C:\Users\Kaizen\Documents\1. Praktikum UGM\PPROKOM XI Ahmad Fawwas Dzulf
```

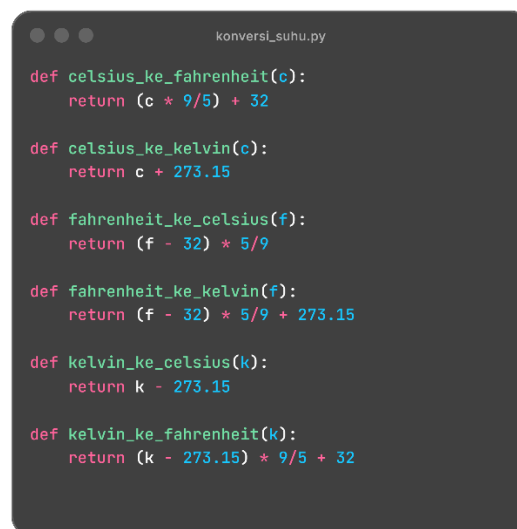
*Tugas 1.3 : output program aritmatika*

## 2.2 Tugas 2

2. Buat program Python yang dapat mengonversi suhu dari dan ke berbagai satuan (Celsius, Fahrenheit, dan Kelvin) dengan pendekatan pemrograman modular. Program ini dibagi menjadi dua file:.

- konversi\_suhu.py → berisi semua fungsi logika konversi suhu.
- main.py → program utama yang menerima input dari pengguna dan menampilkan hasil konversi dengan memanggil fungsi dari modul konversi\_suhu.

Jawaban :



```
konversi_suhu.py

def celsius_ke_fahrenheit(c):
    return (c * 9/5) + 32

def celsius_ke_kelvin(c):
    return c + 273.15

def fahrenheit_ke_celsius(f):
    return (f - 32) * 5/9

def fahrenheit_ke_kelvin(f):
    return (f - 32) * 5/9 + 273.15

def kelvin_ke_celsius(k):
    return k - 273.15

def kelvin_ke_fahrenheit(k):
    return (k - 273.15) * 9/5 + 32
```

*Tugas 2.1 : source code konversi suhu*

Pada file konversi\_suhu.py, berisi kumpulan fungsi yang menangani logika perhitungan konversi suhu antar satuan: Celsius, Fahrenheit, dan Kelvin. Terdapat enam fungsi di dalamnya, yaitu celsius\_ke\_fahrenheit(), celsius\_ke\_kelvin(), fahrenheit\_ke\_celsius(), fahrenheit\_ke\_kelvin(), kelvin\_ke\_celsius(), dan kelvin\_ke\_fahrenheit(). Masing-masing fungsi menerima satu parameter suhu dalam satuan tertentu dan mengembalikan hasil konversinya ke satuan lain berdasarkan rumus konversi suhu yang benar. Misalnya, celsius\_ke\_fahrenheit(c) menggunakan rumus  $(c * 9/5) + 32$ , dan kelvin\_ke\_celsius(k) menggunakan rumus  $k - 273.15$ .



```

main.py

from konversi_suhu import *

def main():
    print("=== PROGRAM KONVERSI SUHU ===")
    print("Pilih satuan asal:")
    print("1. Celsius")
    print("2. Fahrenheit")
    print("3. Kelvin")

    pilihan = input("Masukkan pilihan (1-3): ")

    if pilihan == "1":
        c = float(input("Masukkan suhu dalam Celsius: "))
        print(f"\n{c}°C = {celsius_ke_fahrenheit(c):.2f}°F")
        print(f"{c}°C = {celsius_ke_kelvin(c):.2f}K")

    elif pilihan == "2":
        f = float(input("Masukkan suhu dalam Fahrenheit: "))
        print(f"\n{f}°F = {fahrenheit_ke_celsius(f):.2f}°C")
        print(f"{f}°F = {fahrenheit_ke_kelvin(f):.2f}K")

    elif pilihan == "3":
        k = float(input("Masukkan suhu dalam Kelvin: "))
        print(f"\n{k}K = {kelvin_ke_celsius(k):.2f}°C")
        print(f"{k}K = {kelvin_ke_fahrenheit(k):.2f}°F")

    else:
        print("Pilihan tidak valid!")

if __name__ == "__main__":
    main()

```

*Tugas 2.2 : main konvesi suhu*

Sementara itu, file main.py berfungsi sebagai program utama yang berinteraksi dengan pengguna. Program ini mengimpor semua fungsi dari konversi\_suhu.py menggunakan `from konversi_suhu import *`. Di dalam fungsi `main()`, pengguna diminta memilih satuan suhu asal (Celsius, Fahrenheit, atau Kelvin). Berdasarkan pilihan tersebut, program kemudian meminta input nilai suhu dan menampilkan hasil konversinya ke dua satuan lainnya dengan format angka dua desimal.

```

PS C:\Users\Kaizen\Documents\1. Praktikum UGM\PPROKOM XI Ahmad Fawwas Dzulfahmi> &
XI Ahmad Fawwas Dzulfahmi/Tugas 2/main.py"
=== PROGRAM KONVERSI SUHU ===
Pilih satuan asal:
1. Celsius
2. Fahrenheit
3. Kelvin
Masukkan pilihan (1-3): 1
Masukkan suhu dalam Celsius: 36

36.0°C = 96.80°F
36.0°C = 309.15K
PS C:\Users\Kaizen\Documents\1. Praktikum UGM\PPROKOM XI Ahmad Fawwas Dzulfahmi>

```

*Tugas 2.3 : ouput program konversi suhu*

### 2.3 Tugas 3

3. Buatlah Modul Data Mahasiswa yang berisi dua file yaitu:
  - a. data\_mhs.py berisi fungsi:
    - tambah\_data(nama, nim) untuk menambahkan data mahasiswa ke dalam list.
    - tampilkan\_data() untuk menampilkan semua data mahasiswa yang tersimpan.
  - b. main.py meminta input nama dan NIM dari pengguna, lalu menampilkan daftar mahasiswa yang sudah dimasukkan menggunakan fungsi dari data\_mhs.py.
  - c. Contoh output

```
=== MENU DATA MAHASISWA ===
1. Tambah Data
2. Tampilkan Data
3. Keluar
Pilih menu (1-3): 1
Masukkan nama mahasiswa: Dhia
Masukkan NIM: 230101
Data mahasiswa Dhia (230101) berhasil ditambahkan!

=== MENU DATA MAHASISWA ===
Pilih menu (1-3): 2

=== DAFTAR MAHASISWA ===
1. Dhia (230101)
```

*Tugas 3 : Contoh Output program data mahasiswa*

Jawaban :

```

data_mhs.py

daftar_mahasiswa = []

def tambah_data(nama, nim):
    """Menambahkan data mahasiswa ke dalam list"""
    data = {"nama": nama, "nim": nim}
    daftar_mahasiswa.append(data)
    print(f>Data mahasiswa {nama} ({nim}) berhasil ditambahkan!")

def tampilkan_data():
    """Menampilkan semua data mahasiswa yang tersimpan"""
    if len(daftar_mahasiswa) == 0:
        print("Belum ada data mahasiswa yang tersimpan.")
    else:
        print("\n=== DAFTAR MAHASISWA ===")
        for i, mhs in enumerate(daftar_mahasiswa, start=1):
            print(f"{i}. {mhs['nama']} ({mhs['nim']})")

```

*Tugas 3.1 : source code fungsi data mahasiswa*

File data\_mhs.py berperan sebagai modul penyimpanan dan pengelolaan data. Di dalamnya terdapat list bernama daftar\_mahasiswa untuk menyimpan data setiap mahasiswa dalam bentuk dictionary yang berisi nama dan NIM. Fungsi tambah\_data(nama, nim) digunakan untuk menambahkan data mahasiswa baru ke dalam list tersebut, sedangkan tampilkan\_data() menampilkan seluruh data mahasiswa yang sudah tersimpan. Jika belum ada data, fungsi ini akan menampilkan pesan bahwa data masih kosong.

```

main.py

import data_mhs

def main():
    while True:
        print("\n=== MENU DATA MAHASISWA ===")
        print("1. Tambah Data")
        print("2. Tampilkan Data")
        print("3. Keluar")

        pilihan = input("Pilih menu (1-3): ")

        if pilihan == "1":
            nama = input("Masukkan nama mahasiswa: ")
            nim = input("Masukkan NIM: ")
            data_mhs.tambah_data(nama, nim)

        elif pilihan == "2":
            data_mhs.tampilkan_data()

        elif pilihan == "3":
            print("Terima kasih! Program selesai.")
            break

        else:
            print("Pilihan tidak valid! Silakan coba lagi.")

if __name__ == "__main__":
    main()

```

*Tugas 3.2 : source code menu data mahasiswa*

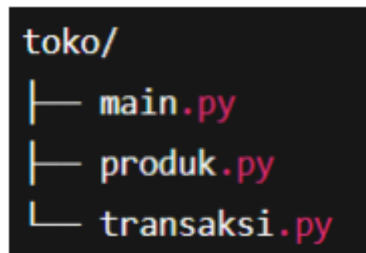
File `main.py` berfungsi sebagai program utama (interface) yang menampilkan menu interaktif kepada pengguna. Pengguna dapat memilih tiga opsi: (1) menambah data mahasiswa, (2) menampilkan data yang sudah tersimpan, atau (3) keluar dari program. Program berjalan dalam loop `while True` sehingga menu akan terus ditampilkan sampai pengguna memilih untuk keluar. File ini mengimpor modul `data_mhs` untuk memanggil fungsi-fungsi dari file tersebut.

*Tugas 3.3 : ouput program data mahasiswa*

## 2.4 Tugas 4

4. Buatlah Paket Aplikasi Penjualan.

- Struktur



- produk.py berisi daftar produk dan harga.
- transaksi.py berisi fungsi untuk menghitung total pembelian dan diskon.
- main.py menampilkan daftar produk, meminta input produk yang dibeli dan jumlahnya, lalu menampilkan total pembayaran setelah diskon menggunakan fungsi dari transaksi.py.
- Contoh output:

```
=== APLIKASI PENJUALAN TOKO PYTHON ===

=== DAFTAR PRODUK ===
1. Keyboard - Rp150000
2. Mouse - Rp80000
3. Flashdisk - Rp50000
Pilih produk (nomor): 1
Masukkan jumlah beli: 3

=== STRUK PEMBAYARAN ===
Produk      : Keyboard
Harga Satuan : Rp150000
Jumlah Beli  : 3
Total Harga  : Rp450000
Diskon       : Rp45000.0
Total Bayar  : Rp405000.0

Apakah ingin belanja lagi? (y/n): n
Terima kasih telah berbelanja!
```

*Tugas 4 : contoh output*

Jawaban :

```

main.py

# main.py
# Program utama untuk aplikasi penjualan toko Python

import produk
import transaksi

def main():
    print("=== APLIKASI PENJUALAN TOKO PYTHON ===")

    while True:
        print()
        produk.tampilkan_produk()

        try:
            pilihan = int(input("\nPilih produk (nomor): "))
            if pilihan not in produk.daftar_produk:
                print("Nomor produk tidak valid!")
                continue

            jumlah = int(input("Masukkan jumlah beli: "))
            if jumlah <= 0:
                print("Jumlah harus lebih dari 0!")
                continue

            # Ambil data produk
            barang = produk.daftar_produk[pilihan]
            nama = barang["nama"]
            harga = barang["harga"]

            # Hitung total dan diskon
            total_harga = transaksi.hitung_total(harga, jumlah)
            diskon = transaksi.hitung_diskon(total_harga)
            total_bayar = total_harga - diskon

            # Tampilkan struk
            print("\n=== STRUK PEMBAYARAN ===")
            print(f"Produk      : {nama}")
            print(f"Harga Satuan : Rp{harga}")
            print(f"Jumlah Beli  : {jumlah}")
            print(f"Total Harga  : Rp{total_harga}")
            print(f"Diskon       : Rp{diskon}")
            print(f"Total Bayar  : Rp{total_bayar}\n")

            # Tanya lanjut
            lagi = input("Apakah ingin belanja lagi? (y/n): ").lower()
            if lagi != 'y':
                print("Terima kasih telah berbelanja!")
                break

        except ValueError:
            print("Input tidak valid! Masukkan angka yang benar.")

if __name__ == "__main__":
    main()

```

*Tugas 4.1 : source code menu utama fitur transaksi*

file main.py di sinilah pengguna berinteraksi langsung. Program menampilkan daftar produk dari produk.py, lalu meminta pengguna memilih produk dan jumlah barang yang ingin dibeli. Setelah itu, program mengambil data produk yang dipilih dan menghitung total harga, diskon, serta total yang harus dibayar dengan bantuan fungsi dari transaksi.py. Hasil perhitungan ditampilkan dalam bentuk struk pembayaran yang berisi nama produk, harga satuan, jumlah beli, total harga, besar diskon, dan total bayar akhir. Setelah struk muncul, pengguna akan ditanya apakah ingin belanja lagi atau keluar dari program.

```

produk.py

daftar_produk = {
    1: {"nama": "Keyboard", "harga": 150000},
    2: {"nama": "Mouse", "harga": 80000},
    3: {"nama": "Flashdisk", "harga": 50000}
}

def tampilkan_produk():
    """Menampilkan daftar produk"""
    print("=== DAFTAR PRODUK ===")
    for nomor, info in daftar_produk.items():
        print(f"{nomor}. {info['nama']} - Rp{info['harga']}")

```

*Tugas 4.2 : source code daftar produk*

```

transaksi.py

def hitung_total(harga_satuan, jumlah):
    """Menghitung total harga tanpa diskon"""
    return harga_satuan * jumlah

def hitung_diskon(total):
    """Menghitung diskon berdasarkan total pembelian"""
    if total >= 500000:
        return total * 0.1 # Diskon 10%
    elif total >= 200000:
        return total * 0.05 # Diskon 5%
    else:
        return 0

```

*Tugas 4.3 :source code aritmatika sederhana transaksi*

File transaksi.py berfungsi untuk melakukan perhitungan harga. Fungsi `hitung_total()` digunakan untuk menghitung total harga berdasarkan harga satuan dan jumlah barang yang dibeli, sedangkan `hitung_diskon()` digunakan untuk menghitung potongan harga atau diskon berdasarkan total belanja. Program ini memberikan diskon 10% jika total belanja mencapai Rp500.000 atau lebih, dan 5% jika mencapai Rp200.000 atau lebih.

```

KeyboardInterrupt
PS C:\Users\Kaizen\Documents\1. Praktikum UGM\PPROKOM XI Ahmad Fawwas Dzulfahmi> & C:/Users/Kaizen/AppData/Local/
XI Ahmad Fawwas Dzulfahmi/Tugas 4/main.py
=== APLIKASI PENJUALAN TOKO PYTHON ===

=== DAFTAR PRODUK ===
1. Keyboard - Rp150000
2. Mouse - Rp80000
3. Flashdisk - Rp50000

Pilih produk (nomor): 1
Masukkan jumlah beli: 3

=== STRUK PEMBAYARAN ===
Produk      : Keyboard
Harga Satuan : Rp150000
Jumlah Beli  : 3
Total Harga  : Rp450000
Diskon       : Rp22500.0
Total Bayar  : Rp427500.0

Apakah ingin belanja lagi? (y/n): n
Terima kasih telah berbelanja!
PS C:\Users\Kaizen\Documents\1. Praktikum UGM\PPROKOM XI Ahmad Fawwas Dzulfahmi>

```

*Tugas 4.4 : output program transaksi*

## **BAB III**

### **PENUTUP**

#### **3.1 Kesimpulan**

Berdasarkan hasil praktikum mengenai pemrograman modular, dapat disimpulkan bahwa pemrograman modular merupakan pendekatan pengembangan program dengan membagi program besar menjadi beberapa bagian (modul) yang memiliki fungsi dan tugas spesifik. Melalui pembagian ini, program menjadi lebih terstruktur, mudah dipahami, dan dapat dikembangkan secara lebih efisien.

Pada praktikum ini, mahasiswa telah menerapkan konsep modular dalam beberapa studi kasus, yaitu pembuatan modul operasi matematika, modul konversi suhu, modul pengelolaan data mahasiswa, serta paket aplikasi penjualan. Setiap modul dibuat secara terpisah dan kemudian digunakan kembali dalam program utama melalui proses impor. Hal ini menunjukkan bahwa penggunaan modul dan fungsi dapat meningkatkan keterbacaan kode, memudahkan proses pemeliharaan, serta mendorong reusabilitas kode dalam proyek yang berbeda.

Dengan demikian, pemrograman modular terbukti membantu dalam pengembangan program yang lebih efektif dan sistematis, serta menjadi dasar penting dalam pengembangan perangkat lunak skala kecil maupun besar.



## DAFTAR PUSTAKA

- GeeksforGeeks. (2018, September 7). *Modular Approach in Programming*. GeeksforGeeks. <https://www.geeksforgeeks.org/software-engineering/modular-approach-in-programming/>
- Lecture 7: Modular Programming: Modules and Signatures*. (2025). Cornell.edu. <https://www.cs.cornell.edu/courses/cs3110/2010fa/lectures/lec07.html>
- MacDonald, M. (2023, June 28). *Modular programming: Definitions, benefits, and Predictions*. Blueprint - Blog by Tiny. <https://www.tiny.cloud/blog/modular-programming-principle/>
- Modular Programming and Modules | Python Tutorial | python-course.eu*. (n.d.). Python-Course.eu. <https://python-course.eu/python-tutorial/modules-and-modular-programming.php>
- The Advantages of Modular Software and Programming – Gwentech Embedded*. (n.d.). <https://gwentechembedded.com/the-advantages-of-modular-software-and-programming/>
- What is Modularity in Software Engineering | Institute of Data*. (2024, February 23). <https://www.institutedata.com/us/blog/modularity-in-software-engineering/>
- Wikipedia Contributors. (2019, February 15). *Modular programming*. Wikipedia; Wikimedia Foundation. [https://en.wikipedia.org/wiki/Modular\\_programming](https://en.wikipedia.org/wiki/Modular_programming)