TUGAS

PEMROGRAMAN BERBASIS OBJEK

DISUSUN OLEH:

AGUNG ADI RANGGA

105841102323

PROGRAM STUDI INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS MUHAMMADIYAH MAKASSAR

2025

Tugas: Studi Kasus Implementasi Class & Object, Enkapsulasi, dan Inheritance

1. **Perpustakaan**
2. **Akademik**

Implementasi :

1. **Perpustakaan**
   - **Source Code :**

```
class Buku:
    def __init__(self, judul, penulis, isbn, tahun_terbit):
        # Atribut "publik"
        self.judul = judul
        self.penulis = penulis
        self._isbn = isbn
        self._tahun_terbit = tahun_terbit
        self._tersedia = True

    def get_isbn(self):
        return self._isbn

    def get_tahun_terbit(self):
        return self._tahun_terbit

    def set_tahun_terbit(self, tahun_baru):
        if isinstance(tahun_baru, int) and tahun_baru >= 0:
            self._tahun_terbit = tahun_baru
            return True
        return False

    # Cek status
    def is_tersedia(self):
        return self._tersedia

    def pinjam(self):
```

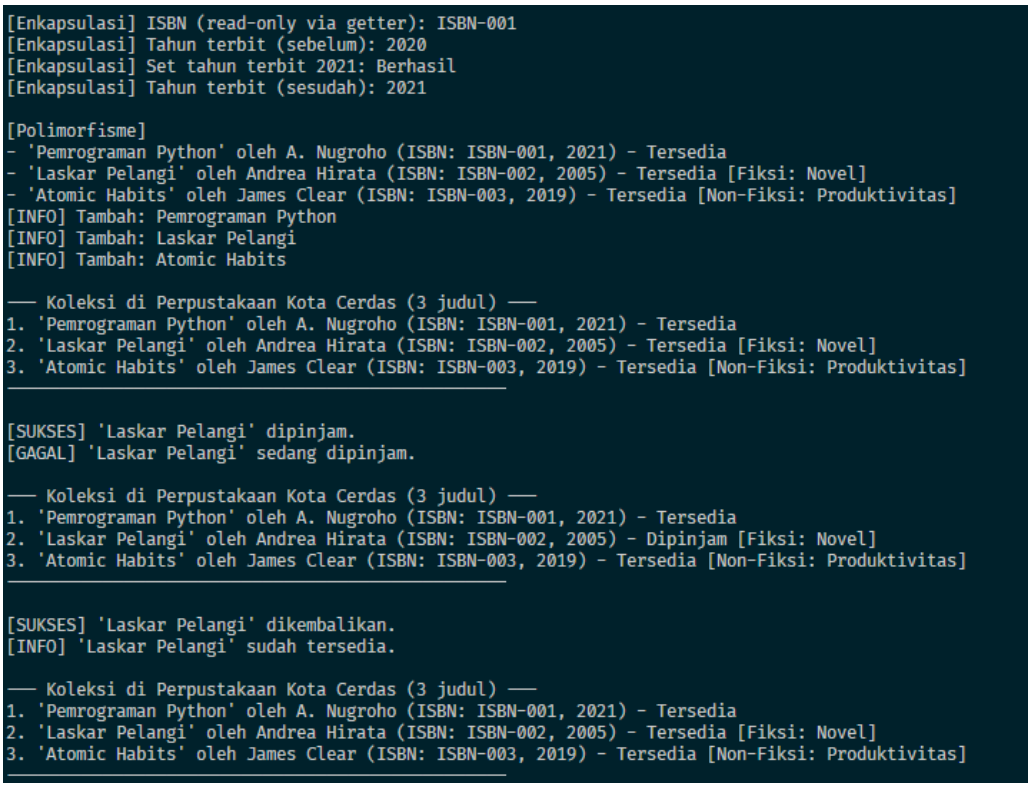
```
        if self._tersedia:
            self._tersedia = False
            return True
        return False

    def kembalikan(self):
        if not self._tersedia:
            self._tersedia = True
            return True
        return False

    # Dapat dioverride oleh subclass
    def info(self):
        status = "Tersedia" if self._tersedia else "Dipinjam"
        return '"' + self.judul + '" oleh ' + self.penulis + " (ISBN: " + self._isbn + ",
" + str(self._tahun_terbit) + ") - " + status

    def __str__(self):
        return self.info()


# 2) INHERITANCE + OVERRIDE
class BukuFiksi(Buku):
    """Mewarisi Buku. Tambah atribut 'genre' dan override info()."""
    def __init__(self, judul, penulis, isbn, tahun_terbit, genre):
        # Panggil constructor class induk
        Buku.__init__(self, judul, penulis, isbn, tahun_terbit)
        self.genre = genre

    # Override
    def info(self):
        return Buku.info(self) + " [Fiksi: " + self.genre + "]"
```

```
class BukuNonFiksi(Buku):
    """Mewarisi Buku. Tambah atribut 'bidang' dan override info()."""
    def __init__(self, judul, penulis, isbn, tahun_terbit, bidang):
        Buku.__init__(self, judul, penulis, isbn, tahun_terbit)
        self.bidang = bidang

    # Override
    def info(self):
        return Buku.info(self) + " [Non-Fiksi: " + self.bidang + "]"



# 3) KELAS MANAJEMEN: Perpustakaan
class Perpustakaan:
    """Mengelola koleksi buku dan operasi pinjam/kembalikan."""
    def __init__(self, nama_perpustakaan):
        self.nama = nama_perpustakaan
        self.daftar_buku = []  # list of Buku

    def tambah_buku(self, buku):
        self.daftar_buku.append(buku)
        print("[INFO] Tambah:", buku.judul)

    def tampilkan_semua_buku(self):
        if len(self.daftar_buku) == 0:
            print(self.nama + " belum memiliki koleksi.")
            return
        print("\n--- Koleksi di " + self.nama + " (" + str(len(self.daftar_buku)) + "
judul) ---")
        nomor = 1
        for bk in self.daftar_buku:
            print(str(nomor) + ".", bk)
            nomor += 1
        print("-------------------------------------------------\n")
```

```
    def _cari_buku(self, judul_buku):
        judul_lower = judul_buku.lower()
        for bk in self.daftar_buku:
            if bk.judul.lower() == judul_lower:
                return bk
        return None

    def pinjam_buku(self, judul_buku):
        bk = self._cari_buku(judul_buku)
        if bk is None:
            print("[GAGAL] '" + judul_buku + "' tidak ditemukan.")
            return
        if bk.pinjam():
            print("[SUKSES] '" + bk.judul + "' dipinjam.")
        else:
            print("[GAGAL] '" + bk.judul + "' sedang dipinjam.")

    def kembalikan_buku(self, judul_buku):
        bk = self._cari_buku(judul_buku)
        if bk is None:
            print("[GAGAL] '" + judul_buku + "' tidak ditemukan.")
            return
        if bk.kembalikan():
            print("[SUKSES] '" + bk.judul + "' dikembalikan.")
        else:
            print("[INFO] '" + bk.judul + "' sudah tersedia.")



    # Membuat objek (class -> object)
buku1 = Buku("Pemrograman Python", "A. Nugroho", "ISBN-001", 2020)
buku2 = BukuFiksi("Laskar Pelangi", "Andrea Hirata", "ISBN-002", 2005,
"Novel")
```

```
buku3 = BukuNonFiksi("Atomic Habits", "James Clear", "ISBN-003", 2019,
"Produktivitas")

    # Enkapsulasi (akses via metode, bukan langsung ubah atribut terlindungi)
print("[Enkapsulasi] ISBN (read-only via getter):", buku1.get_isbn())
print("[Enkapsulasi] Tahun terbit (sebelum):", buku1.get_tahun_terbit())
berhasil = buku1.set_tahun_terbit(2021)
print("[Enkapsulasi] Set tahun terbit 2021:", "Berhasil" if berhasil else "Gagal")
print("[Enkapsulasi] Tahun terbit (sesudah):", buku1.get_tahun_terbit())

    # Inheritance + Override (polimorfisme: info() berbeda untuk subclass)
print("\n[Polimorfisme]")
print("-", buku1.info())
print("-", buku2.info())
print("-", buku3.info())

    # Kelola perpustakaan
perpus = Perpustakaan("Perpustakaan Kota Cerdas")
perpus.tambah_buku(buku1)
perpus.tambah_buku(buku2)
perpus.tambah_buku(buku3)

perpus.tampilkan_semua_buku()

    # Operasi pinjam/kembalikan pakai metode resmi
perpus.pinjam_buku("Laskar Pelangi")
perpus.pinjam_buku("Laskar Pelangi")  # gagal (sudah dipinjam)
perpus.tampilkan_semua_buku()

perpus.kembalikan_buku("Laskar Pelangi")
perpus.kembalikan_buku("Laskar Pelangi")  # info (sudah tersedia)
perpus.tampilkan_semua_buku()
```

- **Output :**



Gambar 1.1

2. Akademik

- **Source Code**

```
# 1) CLASS DASAR
class Orang:
    def __init__(self, nama, umur):
        self.nama = nama
        self.umur = umur

    def peran(self):
        return "Orang"

    def info(self):
        return "Nama: " + self.nama + " | Umur: " + str(self.umur) + " | Peran: " +
self.peran()


# 2) INHERITANCE + ENKAPSULASI + OVERRIDE
```

```
class Mahasiswa(Orang):
    def __init__(self, nama, umur, nim, prodi):
        super().__init__(nama, umur)
        self._nim = nim            # enkapsulasi (gunakan getter/setter)
        self.prodi = prodi
        self._nilai = []           # enkapsulasi daftar nilai

    # Getter/Setter sederhana
    def get_nim(self):
        return self._nim

    def set_nim(self, nim_baru):
        if isinstance(nim_baru, str) and len(nim_baru) >= 5:
            self._nim = nim_baru
            return True
        return False

    def tambah_nilai(self, nilai):
        if isinstance(nilai, (int, float)) and 0 <= nilai <= 100:
            self._nilai.append(float(nilai))
            return True
        return False

    def rata_nilai(self):
        if len(self._nilai) == 0:
            return 0.0
        total = 0.0
        for n in self._nilai:
            total = total + n
        return total / len(self._nilai)

    # Override
    def peran(self):
        return "Mahasiswa"
```

```
    def info(self):
        base = Orang.info(self)
        return base + " | NIM: " + self._nim + " | Prodi: " + self.prodi + " | Rata
Nilai: " + str(round(self.rata_nilai(), 2))


class Dosen(Orang):
    def __init__(self, nama, umur, nidn, jurusan):
        super().__init__(nama, umur)
        self._nidn = nidn          # enkapsulasi
        self.jurusan = jurusan

    def get_nidn(self):
        return self._nidn

    def set_nidn(self, nidn_baru):
        if isinstance(nidn_baru, str) and len(nidn_baru) >= 5:
            self._nidn = nidn_baru
            return True
        return False

    # Override
    def peran(self):
        return "Dosen"

    def info(self):
        base = Orang.info(self)
        return base + " | NIDN: " + self._nidn + " | Jurusan: " + self.jurusan


# 3) MATA KULIAH (ENCAPSULASI + INHERITANCE + OVERRIDE)
class MataKuliah:
    def __init__(self, nama, sks):
```

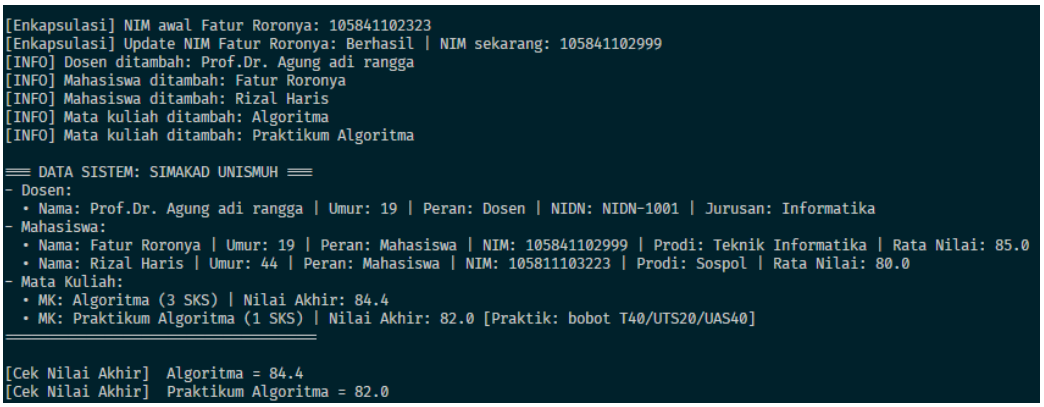
```
        self.nama = nama
        self.sks = sks
        self._tugas = None
        self._uts = None
        self._uas = None

    def set_nilai_tugas(self, nilai):
        if isinstance(nilai, (int, float)) and 0 <= nilai <= 100:
            self._tugas = float(nilai)
            return True
        return False

    def set_nilai_uts(self, nilai):
        if isinstance(nilai, (int, float)) and 0 <= nilai <= 100:
            self._uts = float(nilai)
            return True
        return False

    def set_nilai_uas(self, nilai):
        if isinstance(nilai, (int, float)) and 0 <= nilai <= 100:
            self._uas = float(nilai)
            return True
        return False

    def hitung_nilai_akhir(self):
        # Bobot standar: Tugas 20%, UTS 30%, UAS 50%
        if self._tugas is None or self._uts is None or self._uas is None:
            return None
        return 0.2 * self._tugas + 0.3 * self._uts + 0.5 * self._uas

    def info(self):
        akhir = self.hitung_nilai_akhir()
        akhir_str = "Belum lengkap" if akhir is None else str(round(akhir, 2))
```

```
        return "MK: " + self.nama + " (" + str(self.sks) + " SKS) | Nilai Akhir: " +
akhir_str


class MataKuliahPraktik(MataKuliah):
    # Override bobot: Tugas 40%, UTS 20%, UAS 40%
    def hitung_nilai_akhir(self):
        if self._tugas is None or self._uts is None or self._uas is None:
            return None
        return 0.4 * self._tugas + 0.2 * self._uts + 0.4 * self._uas

    def info(self):
        base = MataKuliah.info(self)
        return base + " [Praktik: bobot T40/UTS20/UAS40]"


class SistemAkademik:
    def __init__(self, nama_sistem):
        self.nama = nama_sistem
        self.daftar_mahasiswa = []
        self.daftar_dosen = []
        self.daftar_mk = []

    def tambah_mahasiswa(self, m):
        self.daftar_mahasiswa.append(m)
        print("[INFO] Mahasiswa ditambah:", m.nama)

    def tambah_dosen(self, d):
        self.daftar_dosen.append(d)
        print("[INFO] Dosen ditambah:", d.nama)

    def tambah_mk(self, mk):
        self.daftar_mk.append(mk)
        print("[INFO] Mata kuliah ditambah:", mk.nama)
```

```
    def tampilkan_data(self):
        print("\n=== DATA SISTEM:", self.nama, "===")
        print("- Dosen:")
        if len(self.daftar_dosen) == 0:
            print("  (Kosong)")
        else:
            for d in self.daftar_dosen:
                print("  •", d.info())

        print("- Mahasiswa:")
        if len(self.daftar_mahasiswa) == 0:
            print("  (Kosong)")
        else:
            for m in self.daftar_mahasiswa:
                print("  •", m.info())

        print("- Mata Kuliah:")
        if len(self.daftar_mk) == 0:
            print("  (Kosong)")
        else:
            for mk in self.daftar_mk:
                print("  •", mk.info())
        print("====================================\n")


# Objek orang akademik
dosen1 = Dosen("Prof.Dr. Agung adi rangga", 19, "NIDN-1001", "Informatika")
mhs1 = Mahasiswa("Fatur Roronya", 19, "105841102323", "Teknik
Informatika")
mhs2 = Mahasiswa("Rizal Haris", 44, "105811103223", "Sospol")

# Enkapsulasi (getter/setter)
print("[Enkapsulasi] NIM awal Fatur Roronya:", mhs1.get_nim())
```

```
berhasil = mhs1.set_nim("105841102999")
print("[Enkapsulasi] Update NIM Fatur Roronya:", "Berhasil" if berhasil else
"Gagal", "| NIM sekarang:", mhs1.get_nim())

# Nilai (enkapsulasi)
mhs1.tambah_nilai(80)
mhs1.tambah_nilai(90)
mhs2.tambah_nilai(75)
mhs2.tambah_nilai(85)

# Mata kuliah
mk1 = MataKuliah("Algoritma", 3)
mk1.set_nilai_tugas(85)
mk1.set_nilai_uts(78)
mk1.set_nilai_uas(88)

mk2 = MataKuliahPraktik("Praktikum Algoritma", 1)
mk2.set_nilai_tugas(90)
mk2.set_nilai_uts(70)
mk2.set_nilai_uas(80)

# Sistem akademik
sistem = SistemAkademik("SIMAKAD UNISMUH")
sistem.tambah_dosen(dosen1)
sistem.tambah_mahasiswa(mhs1)
sistem.tambah_mahasiswa(mhs2)
sistem.tambah_mk(mk1)
sistem.tambah_mk(mk2)

sistem.tampilkan_data()

# Cek override nilai akhir
print("[Cek Nilai Akhir] ", mk1.nama, "=", mk1.hitung_nilai_akhir())
print("[Cek Nilai Akhir] ", mk2.nama, "=", mk2.hitung_nilai_akhir())
```

- **Output**



Gambar 1.2