

**TUGAS UJIAN AKHIR SEMESTER
PEMROGRAMAN BERORIENTASI OBJEK**



DISUSUN OLEH :

1. Della Erlina (G1F022019)
2. Diamond Panca Lady (G1F022027)

Nama Asisten Dosen :

1. Randi Julian Saputra (G1A019066)

Dosen Pengampu :

1. Kurnia Anggriani, S.T., M.T.
2. Arie Vatesia, S.T., M.TI, Ph.D.

**PROGRAM STUDI SISTEM INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS BENGKULU
2023/2024**

BAB I

LANDASAN TEORI

1.1 Apa itu *Python*

Python adalah bahasa pemrograman interpretatif multiguna dengan filosofi perancangan yang berfokus pada tingkat keterbacaan kode. *Python* diklaim sebagai bahasa yang menggabungkan kapabilitas, kemampuan, dengan sintaksis kode yang sangat jelas, dan dilengkapi dengan fungsionalitas pustaka standar yang besar serta komprehensif. *Python* bisa dibilang bahasa pemrograman dengan tujuan umum yang dikembangkan secara khusus untuk membuat *source code* mudah dibaca. *Python* juga memiliki *library* yang lengkap sehingga memungkinkan programmer untuk membuat aplikasi yang mutakhir dengan menggunakan *source code* yang terlihat sederhana. *Python* merupakan bahasa pemrograman komputer yang biasa dipakai untuk membangun situs, software/aplikasi, mengotomatiskan tugas dan melakukan analisis data. Bahasa pemrograman ini termasuk bahasa tujuan umum. Artinya, ia bisa digunakan untuk membuat berbagai program berbeda, bukan khusus untuk masalah tertentu saja. Karena sifatnya yang serba guna dan mudah digunakan, ia menjadi bahasa pemrograman yang paling banyak digunakan. Terutama untuk mereka yang masih pemula. Berdasarkan survei pengembang *Stack Overflow* tahun 2022, *Python* menjadi bahasa pemrograman terpopuler keempat. Sebanyak hampir 50% dari responden mengatakan bahwa mereka menggunakan hampir setengah dari waktu kerja mereka dengan menggunakan bahasa pemrograman ini, nama *Python* sendiri berasal dari *Monty Python*.

Python memiliki berbagai kegunaan yang melibatkan pengembangan perangkat lunak, analisis data, kecerdasan buatan, pengembangan web, otomatisasi tugas, dan lainnya. Sebagai bahasa pemrograman yang mudah dipelajari dan mudah dipahami, *Python* sering digunakan oleh pemula dalam dunia pemrograman. Kejelasan sintaksisnya dan keberagaman modul dan pustaka yang tersedia membuat *Python* menjadi pilihan populer untuk pengembangan prototipe, aplikasi skala besar, dan proyek penelitian. Dalam analisis data, *Python* digunakan secara luas melalui pustaka seperti NumPy, pandas, dan matplotlib, sementara dalam kecerdasan buatan, TensorFlow dan PyTorch menjadikan *Python* bahasa yang paling umum digunakan. Di dunia web, framework seperti Django dan Flask mendukung pengembangan aplikasi web dengan cepat. Kelebihan *Python* termasuk produktivitas tinggi, komunitas yang besar dan aktif, serta fleksibilitas yang menjadikannya bahasa yang sangat serbaguna dalam berbagai konteks pengembangan perangkat lunak.

1.2 Apa itu OOP

Dalam bahasa pemrograman Python, mode operasi Object-Oriented Programming (OOP) mengacu pada penggunaan konsep OOP untuk merancang dan mengembangkan program. OOP adalah paradigma pemrograman yang berfokus pada objek dan hubungan antara objek-objek tersebut atau dapat kita simpulkan juga bahwa Pemrograman berorientasi objek (OOP) adalah metode penataan program dengan menggabungkan properti dan perilaku terkait ke dalam objek individual .

Pemrograman Berorientasi Objek (OOP) memiliki beberapa konsep inti atau bagian yang membentuk dasar kerangka kerja untuk merancang dan mengembangkan aplikasi. Beberapa bagian utama dari OOP meliputi:

1. **Kelas:** Kelas adalah blueprint atau cetakan untuk menciptakan objek dan Mendefinisikan atribut (variabel) dan metode (fungsi) yang akan dimiliki oleh objek.
2. **Objek:** Objek adalah instance konkret dari suatu kelas dan mewakili entitas yang memiliki atribut dan perilaku tertentu.
3. **Atribut:** Atribut adalah variabel yang menyimpan data di dalam objek dan mewakili karakteristik atau properti objek.
4. **Metode:** Metode adalah fungsi yang terkait dengan suatu objek dan menentukan perilaku atau tindakan yang dapat dilakukan oleh objek.
5. **Enkapsulasi:** Enkapsulasi adalah konsep untuk menyembunyikan rincian implementasi objek dan hanya mengekspos fungsionalitas yang diperlukan dan melindungi data dalam objek dari akses langsung dari luar objek.
6. **Pewarisan:** Pewarisan memungkinkan kelas baru (subclass atau turunan) mewarisi atribut dan metode dari kelas yang sudah ada (superclass atau induk) dan meningkatkan reusabilitas dan memungkinkan pengembangan hierarki kelas.
7. **Polimorfisme:** Polimorfisme memungkinkan objek dari kelas yang berbeda untuk merespons metode dengan cara yang sama dan memberikan fleksibilitas dalam mengganti atau menggabungkan perilaku berbeda dalam cara yang konsisten.
8. **Abstraksi:** Abstraksi melibatkan pengelompokan atribut dan metode yang berhubungan untuk membentuk konsep yang lebih tinggi dan menyederhanakan kompleksitas dan memfokuskan pada fitur penting.

BAB II

SOAL DAN PEMBAHASAN

2.1 Soal

1. Buatlah program sederhana dengan menggunakan python dan buat laporan bagaimana konsep OOP diterapkan pada pemrograman tersebut. Program yang dipilih yaitu Migrasi burung dilacak menggunakan Python: Peneliti menggunakan modul GPS untuk melacak hewan seperti kapan dan ke bagian mana mereka bepergian. Kumpulan data Python memberikan detail lokasi menggunakan GPS dan memberikan gambaran tentang tanggal, waktu, lintang, bujur, dan kecepatan burung.

2.2 Pembahasan

1. Program Migrasi Burung menggunakan python

Source Code :

```
import random
from datetime import datetime, timedelta

class GPS:
    def __init__(self, nama):
        self.nama = nama
        self.lokasi = []
        self.kecepatan = random.uniform(5, 20)

    def tambah_lokasi(self, lintang, bujur):
        waktu_sekarang = datetime.now()
        lokasi_baru = {
            'waktu': waktu_sekarang,
            'lintang': lintang,
            'bujur': bujur,
            'kecepatan': self.kecepatan
        }
        self.lokasi.append(lokasi_baru)

    def info_migrasi(self):
        print(f'Detail Migrasi Burung {self.nama}:')
        for lokasi in self.lokasi:
            print(f'Terlacak: {lokasi["waktu"]}, Lintang: {lokasi["lintang"]}, Bujur: {lokasi["bujur"]},
            Kecepatan: {lokasi["kecepatan"]} m/s")
```

```

# Membuat objek Kenari dan Elang
kenari = GPS("Kenari")
elang = GPS("Elang")

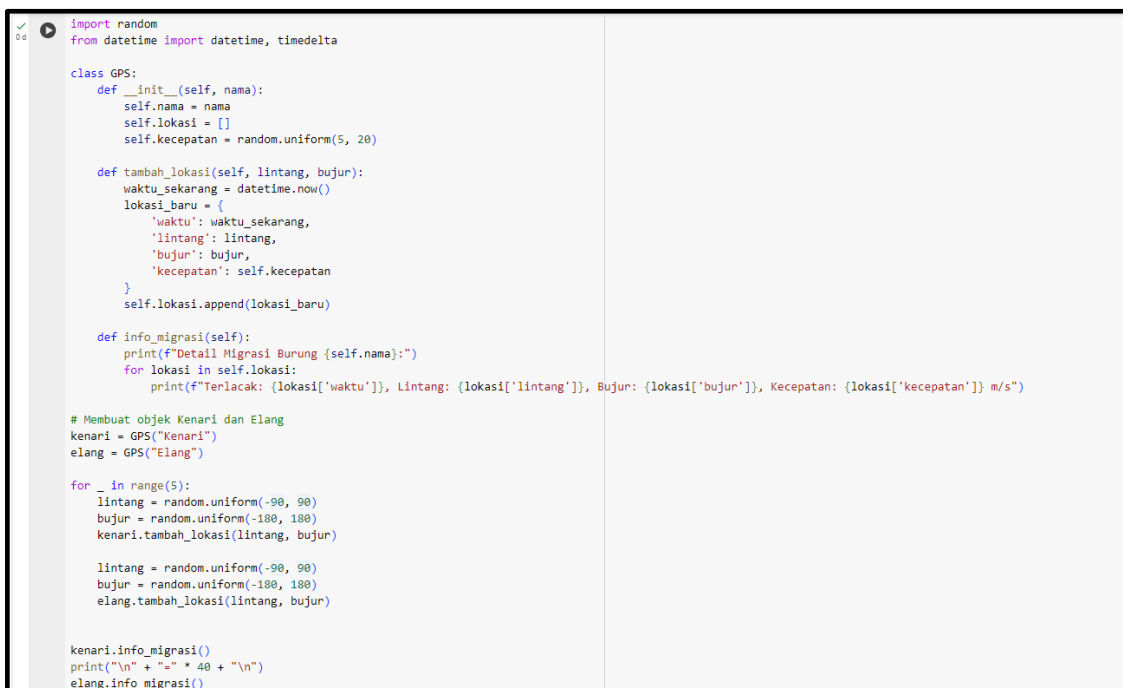
for _ in range(5):
    lintang = random.uniform(-90, 90)
    bujur = random.uniform(-180, 180)
    kenari.tambah_lokasi(lintang, bujur)

    lintang = random.uniform(-90, 90)
    bujur = random.uniform(-180, 180)
    elang.tambah_lokasi(lintang, bujur)

kenari.info_migrasi()
print("\n" + "=" * 40 + "\n")
elang.info_migrasi()

```

Screenshot code :



```

import random
from datetime import datetime, timedelta

class GPS:
    def __init__(self, nama):
        self.nama = nama
        self.lokasi = []
        self.kecepatan = random.uniform(5, 20)

    def tambah_lokasi(self, lintang, bujur):
        waktu_sekarang = datetime.now()
        lokasi_baru = {
            'waktu': waktu_sekarang,
            'lintang': lintang,
            'bujur': bujur,
            'kecepatan': self.kecepatan
        }
        self.lokasi.append(lokasi_baru)

    def info_migrasi(self):
        print(f"Detail Migrasi Burung {self.nama}:")
        for lokasi in self.lokasi:
            print(f"Terlacak: {lokasi['waktu']}, Lintang: {lokasi['lintang']}, Bujur: {lokasi['bujur']}, Kecepatan: {lokasi['kecepatan']} m/s")

# Membuat objek Kenari dan Elang
kenari = GPS("Kenari")
elang = GPS("Elang")

for _ in range(5):
    lintang = random.uniform(-90, 90)
    bujur = random.uniform(-180, 180)
    kenari.tambah_lokasi(lintang, bujur)

    lintang = random.uniform(-90, 90)
    bujur = random.uniform(-180, 180)
    elang.tambah_lokasi(lintang, bujur)

kenari.info_migrasi()
print("\n" + "=" * 40 + "\n")
elang.info_migrasi()

```

Gambar 2.1 *Source code* program

Pembahasan :

Pada gambar 2.1 ini menjelaskan mengenai kode-kode yang digunakan dalam pembuatan program sederhana migrasi burung yang dilacak dengan menggunakan python. Pada kode ini terdapat *library* `import random` from `datetime` dan `timedelta` yaitu digunakan untuk menghasilkan tanggal dan waktu secara acak.

Kemudian kami juga menerapkan konsep OOP yaitu seperti membuat sebuah kelas yang bernama GPS yang memiliki atribut seperti nama, lokasi dan kecepatan, serta terdapat objek dari GPS seperti kenari dan elang dan juga memiliki *method*. Lalu menggunakan kode `def __init__(self, nama):` untuk mendefinisikan dan mengakses/mengatur atribut ketika objek baru dibuat secara otomatis.

Kode `def tambah_lokasi(self, lintang, bujur):` digunakan untuk menambahkan lokasi baru ke dalam daftar lokasi objek GPS yang berisikan waktu sekarang, bujur, lintang dan kecepatan. Kemudian kode *method* `def info_migrasi(self):` digunakan untuk mencetak informasi detail migrasi burung dari objek GPS dan juga menampilkan rincian setiap lokasi termasuk tanggal dan waktu, bujur, lintang, dan kecepatan burung.

Setelah itu kita dapat membuat 2 objek kenari dan elang seperti kode ini `kenari = GPS("Kenari")`, kemudian `for_in range(5):` digunakan untuk melakukan perulangan lokasi secara acak sebanyak 5x untuk setiap burung dan terakhir kita dapat memanggil *method* `info_migrasi` untuk menampilkan seluruh informasi detail migrasi burung untuk setiap objek.

```
Detail Migrasi Burung Kenari:
Terlacak: 2023-12-12 15:05:10.088843, Lintang: -58.33243106756662, Bujur: -52.11895011644731, Kecepatan: 15.005390002762208 m/s
Terlacak: 2023-12-12 15:05:10.088850, Lintang: -76.55171969718383, Bujur: 57.280465839861705, Kecepatan: 15.005390002762208 m/s
Terlacak: 2023-12-12 15:05:10.088853, Lintang: -72.48154326850866, Bujur: -112.9875499106919, Kecepatan: 15.005390002762208 m/s
Terlacak: 2023-12-12 15:05:10.088856, Lintang: 62.549853522051166, Bujur: -39.520367023948694, Kecepatan: 15.005390002762208 m/s
Terlacak: 2023-12-12 15:05:10.088859, Lintang: -49.514454402334955, Bujur: -33.370558877966005, Kecepatan: 15.005390002762208 m/s

=====

Detail Migrasi Burung Elang:
Terlacak: 2023-12-12 15:05:10.088848, Lintang: 71.15599076343187, Bujur: -176.868401404203, Kecepatan: 15.823757329062987 m/s
Terlacak: 2023-12-12 15:05:10.088851, Lintang: 62.52451197050942, Bujur: -141.91315379506736, Kecepatan: 15.823757329062987 m/s
Terlacak: 2023-12-12 15:05:10.088854, Lintang: 51.216139388118194, Bujur: 35.44312892614826, Kecepatan: 15.823757329062987 m/s
Terlacak: 2023-12-12 15:05:10.088857, Lintang: 4.91993203425622, Bujur: -73.03434668530643, Kecepatan: 15.823757329062987 m/s
Terlacak: 2023-12-12 15:05:10.088860, Lintang: -15.87518797767482, Bujur: 58.53425117370554, Kecepatan: 15.823757329062987 m/s
```

Gambar 2.2 *Output* yang dihasilkan dari program

Pembahasan :

Pada gambar 2.2 ini adalah output yang dihasilkan pada program, yang dimana pada output ini terdiri dari detail migrasi burung seperti tanggal dan waktu, bujur, lintang dan kecepatan terbang burung yang ditampilkan secara acak dan berulang sebanyak 5x pada setiap objek.

- Setiap baris mewakili satu data lokasi yang mencakup informasi seperti waktu terlacak, lintang, bujur, dan kecepatan.
- Objek Kenari dan Elang memiliki kecepatan yang sama untuk setiap data lokasi, karena kecepatan diinisialisasi saat objek dibuat.
- Output ini memberikan informasi terstruktur mengenai pergerakan migrasi dari setiap burung selama waktu tertentu.

BAB III

PENUTUP

3.1 Kesimpulan

Kesimpulan yang dapat kami berikan kepada pembaca yaitu bahwa Pemrograman Berorientasi Objek (OOP) dalam Python adalah paradigma pemrograman yang memungkinkan pengorganisasian kode secara konsep objek, di mana data dan fungsionalitas terkait dibungkus bersama dalam suatu unit yang disebut objek. Dalam OOP Python, objek adalah instance dari kelas, yang merupakan blueprint untuk membuat objek. OOP memungkinkan konsep pewarisan, polimorfisme, dan enkapsulasi, yang meningkatkan modularitas, fleksibilitas, dan pemahaman kode. Pemisahan fungsi dan data ke dalam objek membuat pengembangan dan pemeliharaan program menjadi lebih terstruktur, mudah dimengerti, dan dapat diubah dengan lebih mudah, memfasilitasi pengembangan perangkat lunak yang lebih efisien dan dapat dipelihara. Pada kode yang telah dibuat menggunakan konsep enkapsulasi.

3.2 Saran

Setelah membuat laporan ini, saran yang dapat kami berikan yaitu sebelum membuat program ada baiknya kita benar-benar memahami terlebih dahulu mengenai konsep-konsep yang berhubungan dengan penyelesaian pemrograman yang diinginkan maka diwajibkan mencari informasi terlebih dahulu mengenai hal yang ingin dikerjakan, supaya ketika mengerjakan project yang diberikan menjadi lebih mengerti.

DAFTAR PUSTAKA

- Armansyah (2019). “ *PEMROGRAMAN BERORIENTASI OBJEK(Edisi Revisi II)*”. Diakses pada 12 Desember 2023 melalui [http://repository. uinsu.ac.id/12303 /1/Armansyah%20-%20Diktat%20PBO.pdf](http://repository.uinsu.ac.id/12303/1/Armansyah%20-%20Diktat%20PBO.pdf)
- Ma’arif Alvian. (2020). “*BUKU AJAR PEMROGRAMAN LANJUT BAHASA PEMROGRAMAN PYTHON*”. Diakses pada 12 Desember 2023 melalui <https://eprints.uad.ac.id/32743/1/buku%20python.pdf>
- Rahman Sayuti, dkk. (2023). “*PYTHON: DASAR DAN PEMROGRAMAN BERORIENTASI OBJEK*”. Diakses pada 12 Desember 2023 melalui <file:///C:/Users/Lenovo/Downloads/OJS+EBOOK+PYTHON+DASAR+DAN+PEMROGRAMAN.pdf>