2311104005
Muhammad Rifqi Al Baqi Ananta
S1SE-07-01

TP (Tugas Pendahuluan)

1. KodePos.js
   Output:

```
mralb@belphegor MINGW64 /d/TUGAS ITTP/SEMESTER 4/KPL_MuhammadRifqiAlBaqiAnanta_2311104005_SE07/04_OOP/TP_OOP_2311104005 (main)
$ node KodePos.js
Daftar Nama Kelurahan:
Batununggal, Kujangsari, Mengger, Wates, Cijaura, Jatisari, Margasari, Sekejati, Kebonwaru, Maleer, Samoja
Masukkan nama kelurahan: Wates
Kode Pos Wates: 40256
```

   Penjelasan: Kode ini menggunakan konsep table-driven programming, di mana data kode pos disimpan dalam struktur Map sebagai tabel pencarian, memungkinkan akses cepat tanpa perlu menggunakan banyak pernyataan if-else. Kelas KodePos memiliki konstruktor yang menginisialisasi kodePosMap dengan pasangan nama kelurahan dan kode pos. Metode getKodePos(kelurahan) berfungsi untuk mengambil kode pos berdasarkan input kelurahan, mengembalikan nilai yang sesuai atau pesan "Kode pos tidak ditemukan" jika kelurahan tidak ada dalam tabel. Dalam program utama, sebuah instance KodePos dibuat, lalu daftar kelurahan ditampilkan di konsol. Selanjutnya, program mencari kode pos untuk kelurahan "Wates" dengan memanggil getKodePos(), dan hasilnya ditampilkan. Pendekatan table-driven ini membuat kode lebih bersih, mudah diperluas, dan efisien dibandingkan dengan pendekatan berbasis if-else atau switch-case.

2. DoorMachine.js
   Output:

```
mralb@belphegor MINGW64 /d/TUGAS ITTP/SEMESTER 4/KPL_MuhammadRifqiAlBaqiAnanta_2311104005_SE07/04_OOP/TP_OOP_2311104005 (main)
$ node DoorMachine.js
Pintu berubah dari Terkunci ke Terbuka
Pintu tidak terkunci
Pintu berubah dari Terbuka ke Terkunci
Pintu terkunci
```

   Penjelasan: Kode ini menerapkan state-based condition, di mana objek DoorMachine memiliki properti state untuk merepresentasikan status pintu ("Terkunci" atau "Terbuka"). Metode ubahState(stateBaru) mengubah state pintu dan mencetak transisi state yang terjadi. Kemudian, metode tampilkanStatus() menampilkan status pintu berdasarkan nilai state saat ini dengan menggunakan objek lookup sederhana sebagai alternatif dari if-else. Pendekatan ini mengikuti pola Finite State Machine (FSM), karena objek hanya bisa berada dalam state tertentu dan berubah melalui transisi yang telah ditentukan. Dalam fungsi main(), instance DoorMachine dibuat, lalu ubahState() dipanggil untuk mensimulasikan perubahan status pintu dari "Terkunci" ke "Terbuka" dan sebaliknya. Dengan pendekatan state-based, kode menjadi lebih modular dan memudahkan pengelolaan transisi antar-state tanpa memerlukan banyak kondisi bercabang.

Jurnal

1. KodeBuah.js
   Output:

```
mralb@belphegor MINGW64 /d/TUGAS ITTP/SEMESTER 4/KPL_MuhammadRifqiAlBaqiAnanta_2311104005_SE07/04_OOP/Jurnal_OOP_2311104005 (main)
$ node KodeBuah.js
Daftar Nama Buah:
Apel, Aprikot, Alpukat, Pisang, Paprika, Blackberry, Ceri, Kelapa, Jagung, Kurma, Durian, Anggur, Melon, Semangka
Masukkan nama buah: Ceri
Kode Buah Ceri: G00
```

Penjelasan: Kode ini menggunakan pendekatan table-driven programming, di mana data pasangan nama buah dan kode buah disimpan dalam struktur Map untuk memungkinkan pencarian yang cepat tanpa perlu menggunakan banyak if-else atau switch-case. Kelas KodeBuah memiliki konstruktor yang menginisialisasi kodeBuahMap dengan daftar buah beserta kode uniknya. Metode getKodeBuah(namaBuah) digunakan untuk mengambil kode buah berdasarkan input nama buah, mengembalikan nilai yang sesuai atau menampilkan pesan "Kode tidak ditemukan" jika nama buah tidak ada dalam tabel. Dalam program utama, instance KodeBuah dibuat, kemudian daftar buah yang tersedia ditampilkan di konsol. Program juga mensimulasikan pencarian kode buah untuk "Ceri" dengan memanggil getKodeBuah(), dan hasilnya ditampilkan. Dengan pendekatan table-driven, kode menjadi lebih sederhana, mudah diperluas, dan lebih efisien dibandingkan dengan struktur kondisi bercabang.

2. PosisiKarakterGame.js

   Output:

```
mralb@belphegor MINGW64 /d/TUGAS ITTP/SEMESTER 4/KPL_MuhammadRifqiAlBaqiAnanta_2311104005_SE07/04_OOP/Jurnal_OOP_2311104005 (main)
$ node PosisiKarakterGame.js
Posisi berubah dari Berdiri ke Terbang
posisi take off
Posisi berubah dari Terbang ke Mendarat
posisi landing
Posisi berubah dari Mendarat ke Jongkok
posisi istirahat
```

   Penjelasan: Kode ini menggunakan pendekatan state-based programming dengan elemen table-driven programming. Kelas PosisiKarakterGame merepresentasikan karakter dalam game dengan state awal "Berdiri". Metode ubahState(stateBaru) mengubah state karakter, mencetak perubahan dari state sebelumnya ke state baru, lalu memanggil tampilkanAksi() untuk menunjukkan aksi yang sesuai. Dalam tampilkanAksi(), digunakan objek lookup table (aksi) untuk mencocokkan state dengan aksi yang dilakukan, sehingga menghindari penggunaan banyak if-else. Pendekatan state-based memastikan bahwa karakter selalu berada dalam salah satu dari beberapa state yang telah ditentukan ("Berdiri", "Jongkok", "Terbang", "Mendarat"), sementara penggunaan table-driven dalam tampilkanAksi() membuat kode lebih bersih dan mudah diperluas. Dalam program utama, instance PosisiKarakterGame dibuat, lalu metode ubahState() dipanggil untuk mensimulasikan perubahan posisi karakter dalam game.