



PROGRAM BESTDRIVER

JUNI 2023

Disusun oleh:

Rizka Dwi Putri	I0322110
Rizky Indra Maulana	I0322112
Tiara Vionadita Pattikawa	I0222123
Wildan Faris Ashari	I0322129
Alya Zahirah Rachmatullah	I0322133

KATA PENGANTAR

Puji syukur penulis ucapkan atas kehadiran Tuhan Yang Maha Esa atas segala rahmat dan hidayah-Nya penulis dapat menyelesaikan penyusunan tugas ini dengan judul Laporan *Project Team: Program BestDriver*.

Laporan ini penulis susun untuk memenuhi tugas dari Mata Kuliah Praktikum Program Komputer bagi Mahasiswa Program Studi Teknik Industri, Universitas Sebelas Maret.

Dalam penulisan laporan ini, penulis mendapatkan banyak bimbingan ilmu serta pengetahuan baru yang bermanfaat selama masa pembuatan dari semua pihak yang terlibat. Oleh karena itu, pada kesempatan ini penulis ingin mengucapkan terima kasih yang sebesar-besarnya kepada Bapak I Wayan Suletra, S.T.,M.T. dan Bapak Dr. Eko Liquidanu, S.T.,M.T. selaku Dosen Praktikum Program Komputer yang telah membimbing penulis dari yang tidak tahu menjadi tahu. Terima kasih juga kami ucapkan kepada asisten Laboratorium Perancangan dan Optimasi Sistem Industri (Lab. POSI) yang telah meluangkan waktu untuk mendampingi kami dalam menyelesaikan laporan ini, serta semua pihak yang terlibat baik secara moril maupun materiil pada penyusunan makalah ini.

Penulis menyadari akan jauhnya laporan ini dari kata sempurna, untuk itu penulis secara terbuka akan menerima kritik maupun saran yang membangun dari pembaca laporan ini sebagai bentuk edukasi dan penyempurnaan pembuatan laporan selanjutnya. Penulis berharap laporan ini dapat memberikan manfaat bagi semua pihak.

Surakarta, 17 Juni 2023

Penulis

DAFTAR ISI

KATA PENGANTAR.....	i
DAFTAR ISI.....	ii
BAB I PENDAHULUAN	1
1.1 Latar Belakang Masalah	1
1.2 Deskripsi Masalah	1
1.3 Tujuan.....	1
BAB II DIAGRAM ALIR	3
2.1 Diagram Alir (<i>Flowchart</i>) Program BestDriver	3
2.2 Penjelasan Diagram Alir (<i>Flowchart</i>) Program BestDriver	6
BAB III KODE PEMROGRAMAN	10
3.1 Database Program.....	10
3.1.1 Database Daftar Harga Layanan Jasa di Pulau Jawa.....	10
3.1.2 Database Daftar Jarak Antar Wilayah di Jawa.....	11
3.2 Modul Program.....	13
3.2.1 Modul Car Dalam.....	13
3.2.2 Modul Car Luar	16
3.2.3 Modul Box Dalam	20
3.2.4 Modul Box Luar	22
3.2.5 Modul Ride	24
3.2.6 Modul Send.....	26
3.3 Program Utama.....	29
3.3.1 Syntax Import Modul	29
3.3.2 Def Main	30
3.3.3 Def Layanan Kurir	31
3.3.4 Def Layanan Kurir Dalam Kota	32

3.3.5 Def Layanan Kurir Luar Kota	34
3.3.6 Def Layanan Transportasi	36
3.3.7 Def Layanan Transportasi Mobil	37
3.3.8 Def Layanan Transportasi Motor	40
BAB IV HASIL DAN PEMBAHASAN	42
5.1 Hasil Running Program Python	42

BAB I

PENDAHULUAN

1.1 Latar Belakang Masalah

Dalam era digital dan kemajuan teknologi, layanan transportasi dan kurir *online* telah mengalami perkembangan pesat di Indonesia. Aplikasi-aplikasi seperti Gojek, Grab, dan Maxim telah menjadi bagian penting dari kehidupan sehari-hari masyarakat. Mereka tidak hanya menyediakan layanan transportasi roda dua dan roda empat, tetapi juga pengiriman barang, pengiriman dokumen, dan banyak lagi.

Namun, dengan berbagai pilihan aplikasi yang tersedia, *user* sering kali merasa kesulitan dalam memilih opsi yang terbaik. Misalnya, seseorang yang ingin memesan layanan transportasi dari titik A ke titik B harus membandingkan harga dari Gojek, Grab, dan Maxim secara terpisah. Proses ini dapat memakan waktu yang cukup lama dan menyulitkan *user*, terutama jika mereka ingin mencari harga terbaik atau jika mereka sedang terburu-buru.


Hal ini menciptakan kebutuhan akan sebuah solusi yang dapat membantu *user* dalam membandingkan harga dan layanan dari aplikasi Gojek, Grab, dan Maxim secara efisien. Dengan menggunakan sebuah program rekomendasi transportasi *online*, *user* dapat dengan mudah melihat informasi terkini tentang harga dan jenis layanan yang ditawarkan oleh ketiga aplikasi tersebut dalam satu tampilan. Hal ini akan menghemat waktu dan usaha *user*, serta memberikan kejelasan dalam memilih opsi yang paling sesuai dengan kebutuhan mereka.

1.2 Deskripsi Masalah

Masalah yang dihadapi oleh *user* adalah kesulitan dalam membandingkan harga dari aplikasi Gojek, Grab, dan Maxim secara efisien. Saat ini, mereka harus membuka masing-masing aplikasi secara terpisah dan mencari informasi harga yang mereka butuhkan. Hal ini tidak hanya memakan waktu, tetapi juga menyulitkan *user* untuk menemukan pilihan terbaik yang sesuai dengan kebutuhan dan anggaran mereka.

1.3 Tujuan

Tujuan dari pembuatan program BestDriver ini adalah untuk memberikan solusi yang efisien dan memudahkan *user* dalam membandingkan harga dari aplikasi Gojek, Grab, dan Maxim. Dengan menggunakan program ini, *user* tidak perlu lagi membuka



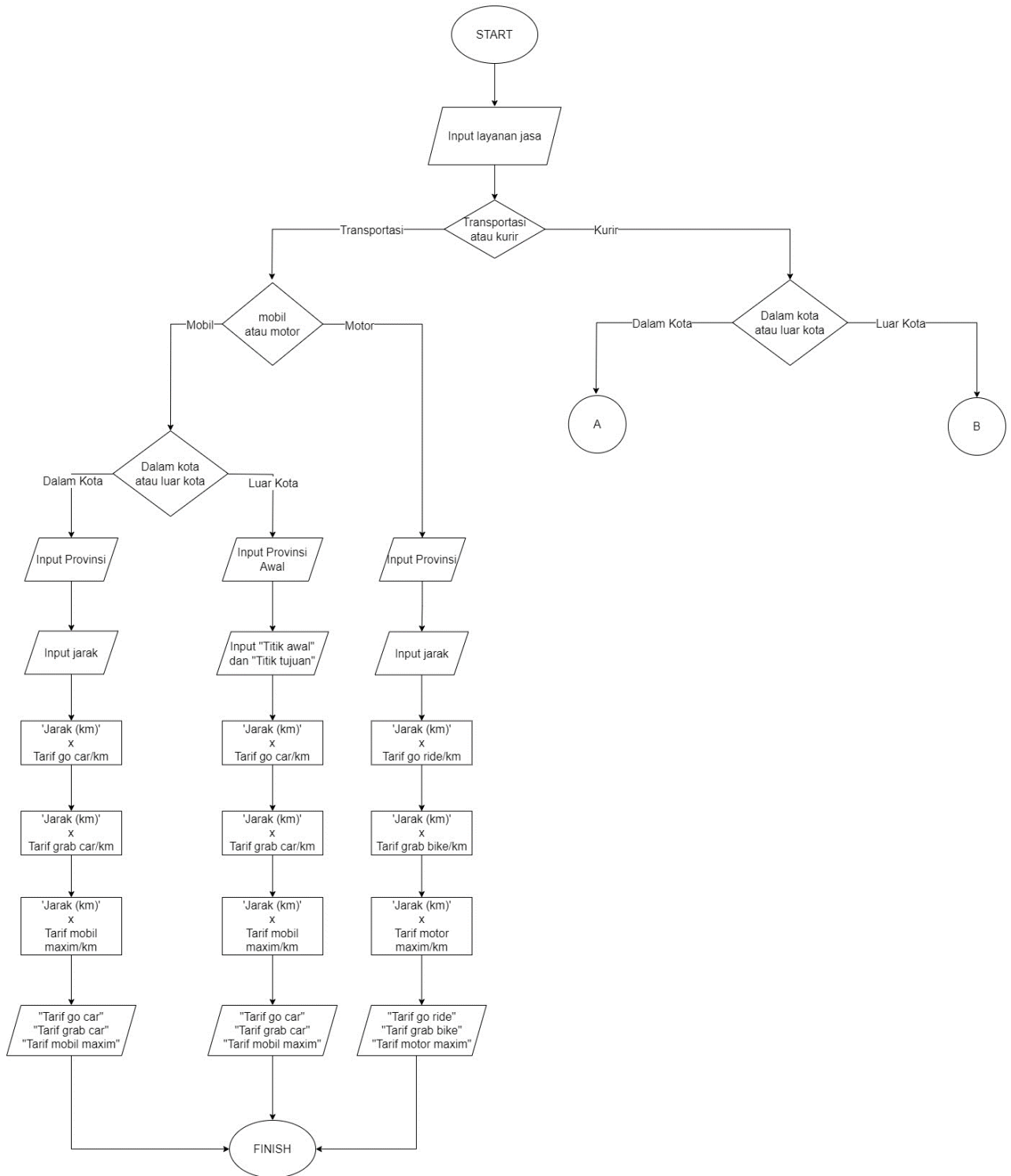
ketiga aplikasi secara terpisah, melainkan dapat melihat harga yang ditawarkan oleh ketiga aplikasi tersebut dalam satu tampilan.

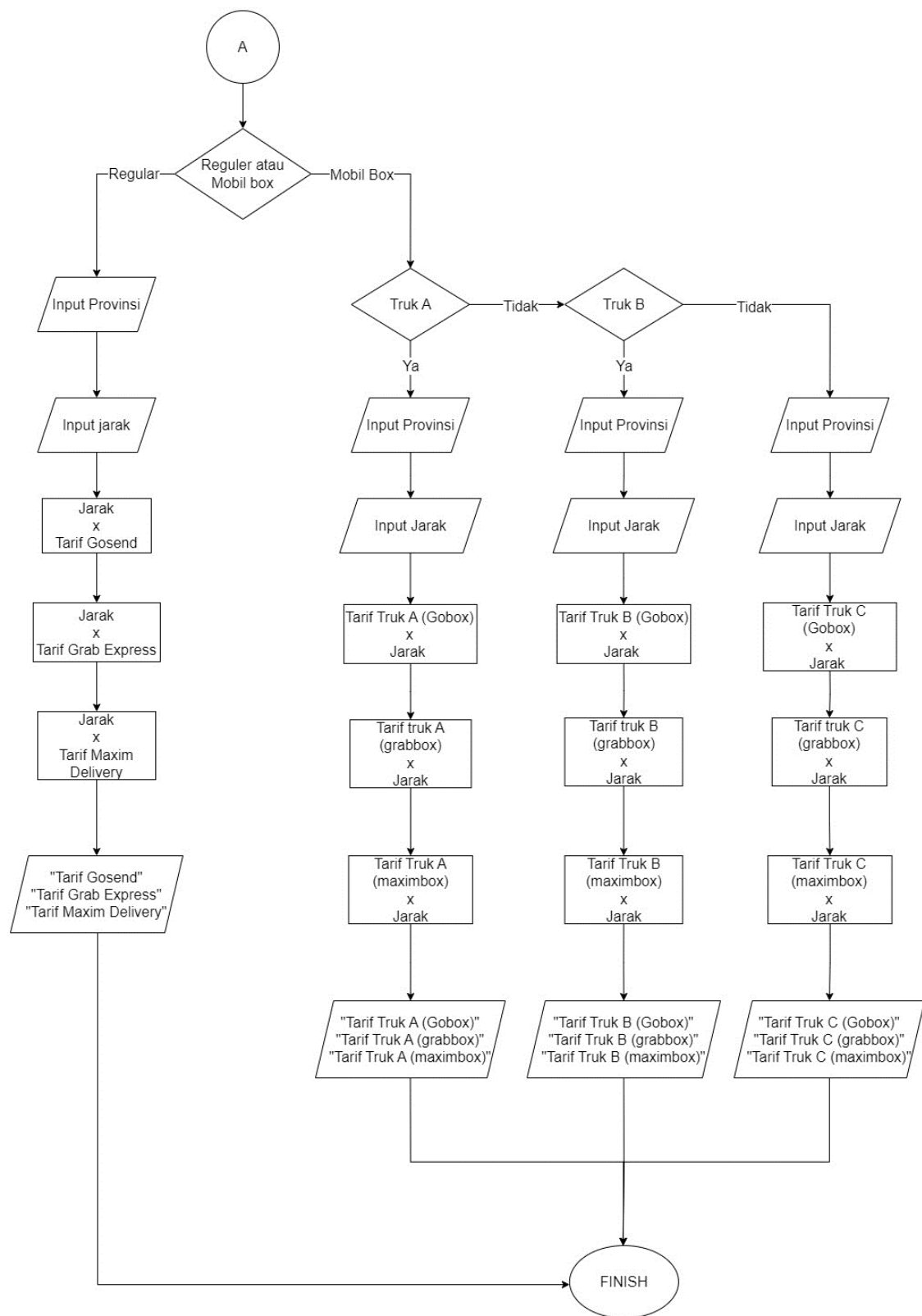
Program rekomendasi ini akan memberikan informasi harga yang akurat dan terbaru dari aplikasi Gojek, Grab, dan Maxim, sehingga *user* dapat dengan cepat melihat pilihan yang tersedia dan memilih opsi terbaik yang sesuai dengan kebutuhan mereka. Hal ini akan menghemat waktu dan usaha *user* dalam mencari harga terbaik serta meningkatkan pengalaman *user* dalam menggunakan layanan transportasi *online*.

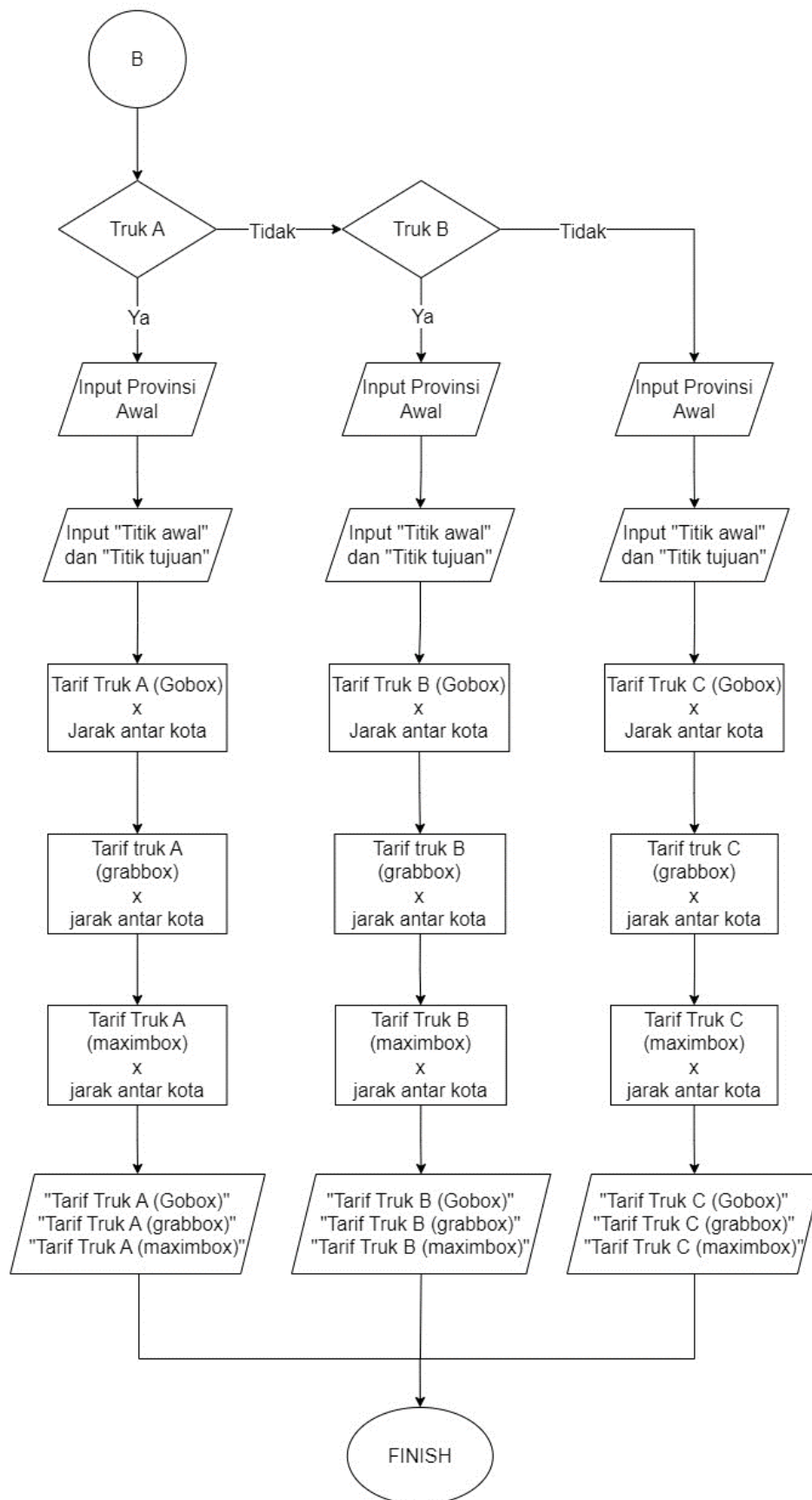
BAB II

DIAGRAM ALIR

2.1 Diagram Alir (*Flowchart*) Program BestDriver







2.2 Penjelasan Diagram Alir (*Flowchart*) Program BestDriver

Judul : Proses Rekomendasi Layanan

Tujuan : Menggambarkan langkah-langkah yang harus diikuti untuk mengetahui layanan rekomendasi.

Proses :

Langkah 1: *User* membuka aplikasi BestDriver

Langkah 2: *User* memilih layanan yang diinginkan dari daftar layanan yang tersedia yaitu transportasi atau kurir.

- Jika *user* memilih layanan ‘Transportasi’:

Langkah 3: *User* memilih jenis transportasi yang ingin digunakan. Adapun pilihan transportasi yang tersedia yakni motor atau mobil.

- Jika *user* memilih transportasi ‘Motor’:

Langkah 4: *User* memasukkan detail provinsi.

Langkah 5: *User* memasukkan detail jarak yang akan ditempuh dari lokasi awal ke lokasi tujuan dalam satuan kilometer.

- Input *user* akan dikirim ke sistem untuk diolah dan *user* menerima informasi rekomendasi layanan jasa motor.

- Jika *user* memilih transportasi ‘Mobil’:

Langkah 4: *User* memilih layanan jasa mobil dalam kota atau luar kota.

- Jika *user* memilih layanan ‘Luar Kota’:

Langkah 5: *User* memasukkan detail provinsi.

Langkah 6: *User* memasukkan detail titik awal dan titik tujuan.

- Input *user* akan dikirim ke sistem untuk diolah dan *user* menerima informasi rekomendasi layanan jasa mobil.

- Jika *user* memilih layanan ‘Dalam Kota’:

Langkah 5: *User* memasukkan detail provinsi awal.

Langkah 6: *User* memasukkan detail jarak yang akan ditempuh dari lokasi awal ke lokasi tujuan dalam satuan kilometer.

- Input *user* akan dikirim ke sistem untuk diolah dan *user* menerima informasi rekomendasi layanan jasa mobil.

- Jika *user* memilih layanan ‘Kurir’:

Langkah 3: *User* memilih jenis layanan jasa kurir yang ingin digunakan. Adapun pilihan layanan yang tersedia adalah dalam kota atau luar kota,

- Jika *user* memilih layanan ‘Dalam Kota’:

Langkah 4: *User* memilih tipe pelayanan. Adapun tipe pelayanan yang tersedia yakni reguler atau mobil box.

- Jika *user* memilih tipe ‘Reguler’:

Langkah 5: *User* memasukkan detail provinsi.

Langkah 6: *User* memasukkan detail jarak yang akan ditempuh dari lokasi awal ke lokasi tujuan dalam satuan kilometer.

- Input *user* akan dikirim ke sistem untuk diolah dan *user* menerima informasi rekomendasi layanan jasa kurir tipe reguler.

- Jika *user* memilih tipe ‘Mobil Box’:

Langkah 5: *User* memilih jenis truk yang diinginkan. Adapun pilihan jenis truk yang tersedia yakni kecil (*small*), sedang (*medium*), dan besar (*large*).

- Jika *user* memilih truk ‘*Small*’:

Langkah 6: *User* memasukkan detail provinsi.

Langkah 7: *User* memasukkan detail jarak yang akan ditempuh dari lokasi awal ke lokasi tujuan dalam satuan kilometer.

- Input *user* akan dikirim ke sistem untuk diolah dan *user* menerima informasi rekomendasi layanan jasa kurir tipe *small*.

- Jika *user* memilih truk '*Medium*':

Langkah 6: *User* memasukkan detail provinsi.

Langkah 7: *User* memasukkan detail jarak yang akan ditempuh dari lokasi awal ke lokasi tujuan dalam satuan kilometer.

➤ Input *user* akan dikirim ke sistem untuk diolah dan *user* menerima informasi rekomendasi layanan jasa kurir tipe *medium*.

- Jika *user* memilih truk '*Large*':

Langkah 6: *User* memasukkan detail provinsi.

Langkah 7: *User* memasukkan detail jarak yang akan ditempuh dari lokasi awal ke lokasi tujuan dalam satuan kilometer.

➤ Input *user* akan dikirim ke sistem untuk diolah dan *user* menerima informasi rekomendasi layanan jasa kurir tipe *large*.

- Jika *user* memilih layanan '*Luar Kota*':

Langkah 4: *User* memilih jenis truk yang diinginkan. Adapun pilihan jenis truk yang tersedia yakni kecil (*small*), sedang (*medium*), dan besar (*large*).

- Jika *user* memilih truk '*Small*':

Langkah 5: *User* memasukkan detail provinsi awal.

Langkah 6: *User* memasukkan detail titik awal dan titik tujuan.

➤ Input *user* akan dikirim ke sistem untuk diolah dan *user* menerima informasi rekomendasi layanan jasa kurir tipe *small*.

- Jika *user* memilih truk '*Medium*':

Langkah 5: *User* memasukkan detail provinsi awal.

Langkah 6: *User* memasukkan detail titik awal dan titik tujuan.

➤ Input *user* akan dikirim ke sistem untuk diolah dan *user* menerima informasi rekomendasi layanan jasa kurir tipe *medium*.

- Jika *user* memilih truk '*Large*':

Langkah 5: *User* memasukkan detail provinsi awal.

Langkah 6: *User* memasukkan detail titik awal dan titik tujuan.

- Input *user* akan dikirim ke sistem untuk diolah dan *user* menerima informasi rekomendasi layanan jasa kurir tipe *large*.

BAB III

KODE PEMROGRAMAN

3.1 Database Program

3.1.1 Database Daftar Harga Layanan Jasa di Pulau Jawa

```
1  {
2    "maxim":
3    [{
4      "layanan": "Bike", "provinsi":
5      {
6        "Jawa Tengah": {
7          "harga_per_km": 1400,
8          "harga_minimum": 8900
9        },
10       "Jawa Timur": {
11         "harga_per_km": 2600,
12         "harga_minimum": 8.600
13       },
14       "Jawa Barat": {
15         "harga_per_km": 1900,
16         "harga_minimum": 8200
17       },
18       "Banten": {
19         "harga_per_km": 1000,
20         "harga_minimum": 8900
21       },
22       "Yogyakarta": {
23         "harga_per_km": 2300,
24         "harga_minimum": 8.900
25       },
26       "Jakarta": {
27         "harga_per_km": 2500,
28         "harga_minimum": 11.200
29       }
30     }
31   },
```

Dalam modul di atas, terdapat informasi yang sangat berguna bagi *user* terkait dengan harga-harga layanan yang ditawarkan oleh Maxim Bike, Gojek, dan Grab. Bagi *user* Maxim Bike, terdapat daftar harga per kilometer yang ditentukan dan harga minimum, keduanya bernilai berbeda di setiap provinsi. Selain Maxim Bike, modul ini juga memberikan informasi tentang jasa transportasi mobil dan motor yang disediakan oleh Gojek dan Grab. Tarif jasa yang ditawarkan oleh kedua platform tersebut juga bervariasi sesuai dengan provinsi *user*. Sehingga, *user* perlu mengacu pada

daftar harga dan menghitung biaya perjalanan berdasarkan harga per kilometer dan jarak yang ditempuh, serta memperhatikan harga minimum yang berlaku.

Dengan adanya informasi ini, bila dipanggil ke kode utama dapat diperoleh perkiraan biaya yang diperlukan untuk menggunakan layanan transportasi yang sesuai dengan kebutuhan dan lokasi. Selain itu, *user* juga dapat membandingkan harga antara Maxim Bike, Gojek, dan Grab untuk memilih opsi yang paling sesuai dengan preferensi dan anggaran mereka.

3.1.2 Database Daftar Jarak Antar Wilayah di Jawa

```
148 },
149 "Jawa Tengah": {
150     "Magelang": {
151         "Pekalongan": 169,
152         "Salatiga": 47,
153         "Semarang": 81,
154         "Surakarta": 103,
155         "Tegal": 230
156     },
157     "Pekalongan": {
158         "Magelang": 171,
159         "Salatiga": 138,
160         "Semarang": 99,
161         "Surakarta": 194,
162         "Tegal": 79
163     },
164     "Salatiga": {
165         "Magelang": 48,
166         "Pekalongan": 145,
167         "Semarang": 50,
168         "Surakarta": 61,
169         "Tegal": 207
170     },
171     "Semarang": {
172         "Magelang": 83,
173         "Pekalongan": 99,
174         "Salatiga": 50,
175         "Surakarta": 105,
176         "Tegal": 161
177     },
178     "Surakarta": {
179         "Magelang": 104,
180         "Pekalongan": 193,
181         "Salatiga": 61,
182         "Semarang": 105,
183         "Tegal": 255
184     },
185 }
```

Pada modul di atas adalah bentuk kode python yang merupakan suatu representasi data yang dikemas dalam bentuk *dictionary* yang menggambarkan jarak antar beberapa kota di salah satu provinsi di Indonesia, yaitu Jawa Tengah. Sebagai contoh, pada wilayah Jawa Tengah ia akan memiliki sub-wilayah dalam bentuk kota-kota di dalamnya dan

jarak antar kota yang ada berbentuk angka yang dihitung berdasarkan jarak antar kota per-kilometernya.

Kode di atas akan disatukan dengan kode yang juga menyediakan berbagai informasi tentang jarak antara kota-kota di berbagai wilayah. Dengan menggunakan kode dan data di atas, kita dapat mengakses jarak antara dua kota dalam provinsi yang dipilih.

```
1  {
2    "Banten": {
3      "Cilegon": {
4        "Serang": 17,
5        "Tangerang": 83,
6        "Tangerang Selatan": 90,
7        "Jakarta Barat": 92,
8        "Jakarta Pusat": 99,
9        "Jakarta Selatan": 105,
10       "Jakarta Timur": 112,
11       "Jakarta Utara": 117,
12       "Bandung": 247,
13       "Banjar": 387,
14       "Bekasi": 126,
15       "Bogor": 132,
16       "Cimahi": 239,
17       "Cirebon": 315,
18       "Depok": 112,
19       "Sukabumi": 206,
20       "Tasikmalaya": 355,
21       "Magelang": 610,
22       "Pekalongan": 456,
23       "Salatiga": 578,
24       "Semarang": 538,
25       "Surakarta": 644,
26       "Tegal": 385,
27       "Yogyakarta": 658,
28       "Batu": 878,
29       "Blitar": 848,
30       "Kediri": 808,
31       "Madiun": 733,
32       "Malang": 947,
33       "Mojokerto": 835,
34       "Pasuruan": 921,
35       "Probolinggo": 958,
36       "Surabaya": 878
37     },
38   }
```

Pada modul ini kode juga bekerja sebagai representasi data yang berbentuk *nested dictionary* yang pada kode nya membentuk jarak antar antara beberapa kota di provinsi Jawa dengan Banten di mana *key* adalah nama kota tujuan dan *value* ialah jarak dalam kilometer dari kota asal. Dengan Menggunakan struktur data dan kode ini dapat dilakukan pemodelan jarak antar kota dari luar suatu provinsi.

3.2 Modul Program

3.2.1 Modul Car Dalam

```
1 import json
2
3 class ProvinsiTidakTersediaError(Exception):
4     pass
```

‘Import json’ adalah sintaks dalam bahasa pemrograman yang berfungsi untuk mengimpor modul ‘json’ ke dalam program Python. Modul ‘json’ sendiri merupakan modul *built-in* dalam bahasa pemrograman Python yang berisikan fungsi-fungsi untuk bekerja dengan format data JSON (*JavaScript Object Notation*).

Kelas ‘ProvinsiTidakTersediaError’ merupakan turunan dari kelas bawaan Python yang disebut ‘Exception’. Kelas ‘Exception’ digunakan untuk membuat pengecualian atau kesalahan spesifik dalam program sehingga pada saat menjalankan kode, program dapat menangani situasi *error* yang terjadi. Pernyataan ‘pass’ digunakan sebagai pernyataan kosong atau penanda di mana kode dapat ditambahkan di masa mendatang.

- **Def Get input**

```
6 def get_input():
7     input_provinsi = input("Input Provinsi Asal : ")
8     try:
9         jarak = float(input("Jarak Tempuh (km): "))
10    except ValueError:
11        raise ValueError("Input jarak harus berupa angka")
12    return input_provinsi, jarak
```

‘get_input()’ didefinisikan sebagai fungsi untuk memperoleh input *user* berupa provinsi asal dan jarak tempuh. Pertama-tama, *user* diminta untuk memasukkan nama provinsi asal. Nilai yang dimasukkan *user* akan disimpan dalam variabel ‘input_provinsi’. Selanjutnya, *user* diminta untuk memasukkan jarak yang akan ditempuh dalam satuan kilometer. Nilai yang dimasukkan *user* akan disimpan dalam variabel ‘jarak’. Kemudian terdapat fungsi ‘Try-except’ yang digunakan untuk memastikan bahwa input jarak berbentuk angka atau tipe data ‘float’. Jika *user* memasukkan nilai yang tidak dapat dikonversi menjadi tipe data ‘float’, maka fungsi akan menghasilkan ‘ValueError’ dengan pesan “Input jarak harus berupa angka”. Dengan demikian, jika tidak terdapat kesalahan pada input *user*, nilai ‘input_provinsi’ dan ‘jarak’ akan dikembalikan sebagai hasil dari fungsi ‘get_input()’.

- **Def Layanan Mobil dalam Kota**

```
15 def layanan_mobil_dalam_kota(layanan, input_provinsi, jarak):
16     with open("daftarharga.json", "r") as file1:
17         data1 = json.load(file1)
18
19     for item in data1[layanan]:
20         if item["layanan"] == "Go Car" or item["layanan"] == "Grab Car" or item["layanan"] == "Car":
21             provinsi_data = item["provinsi"]
22             if input_provinsi not in provinsi_data:
23                 raise ProvinsiTidakTersediaError("Provinsi yang anda input tidak terdapat di Pulau Jawa")
24             harga_minimum = provinsi_data.get(input_provinsi)["harga_minimum"]
25             harga_per_km = provinsi_data.get(input_provinsi)["harga_per_km"]
26             if jarak <= 3:
27                 total_harga = harga_minimum
28             else:
29                 total_harga = harga_minimum + (harga_per_km * jarak)
30             return total_harga
31
32     raise Exception("Input Anda Tidak Sesuai, Silakan Cek Kembali")
```

Program `layanan_mobil_dalam_kota` didefinisikan sebuah fungsi yang digunakan untuk menghitung harga taksi berdasarkan jenis layanan, provinsi asal, dan jarak tempuh. Fungsi ini membaca data harga mobil/taxi dari file "daftarharga.json" dan mengolahnya. Fungsi ini menerima tiga parameter yaitu `layanan`, `input_provinsi` (provinsi asal/awal), dan `jarak`.

Langkah pertama, fungsi akan membaca data harga mobil dari file "daftarharga.json" dan menyimpannya dalam variabel `data1` menggunakan fungsi `json.load()`. Selanjutnya, fungsi melakukan iterasi melalui setiap elemen dalam daftar harga yang sesuai dengan jenis layanan yang diberikan (`Go Car`, `Grab Car`, atau `Car`). Jika provinsi tujuan yang diinputkan tidak terdapat dalam daftar provinsi yang tersedia, fungsi akan menghasilkan pengecualian `ProvinsiTidakTersediaError` dengan pesan "Provinsi yang Anda input tidak terdapat di Pulau Jawa".

Jika provinsi tujuan tersedia, fungsi akan mengolah harga minimum dan harga per kilometer dari data provinsi tersebut. Jika jarak tempuh mobil kurang dari atau sama dengan 3 kilometer, fungsi akan menghitung total harga pengiriman berdasarkan harga minimum provinsi tersebut. Jika jarak tempuh mobil lebih dari 3 kilometer, fungsi akan menghitung total harga mobil dengan menambahkan harga minimum dengan perkalian harga per kilometer dengan jarak tempuh.

Terakhir, fungsi akan mengembalikan total harga mobil. Jika tidak ada layanan yang sesuai dengan input yang diberikan, fungsi akan menghasilkan pengecualian `Exception` dengan pesan "Input Anda Tidak Sesuai, Silakan Cek Kembali".

- **Def Main**

```
34 def main():
35     try:
36         input_provinsi, jarak = get_input()
37         harga_MaximCar = layanan_mobil_dalam_kota("maxim", input_provinsi, jarak)
38         harga_GoCar = layanan_mobil_dalam_kota("gojek", input_provinsi, jarak)
39         harga_GrabCar = layanan_mobil_dalam_kota("Grab", input_provinsi, jarak)
40         print("Harga Maxim Car : ", harga_MaximCar)
41         print("Harga Go Car : ", harga_GoCar)
42         print("Harga Grab Car : ", harga_GrabCar)
43
44     except ProvinsiTidakTersediaError as e:
45         print("Terjadi kesalahan : ", str(e))
46     except ValueError as e:
47         print("Terjadi kesalahan : ", str(e))
48     except Exception as e:
49         print("Terjadi Kesalahan : ", str(e))
50
51 if __name__ == "__main__":
52     main()
```

Program `main` adalah titik masuk utama dari program. Ini menggunakan fungsi `layanan_mobil_dalam_kota` untuk menghitung tarif taksi dengan berbagai layanan, provinsi asal, dan jarak pengiriman yang diinputkan.

Pada awalnya, program mencoba menjalankan blok kode dalam blok `try`. Jika terjadi pengecualian selama eksekusi, program menangani pengecualian tersebut dalam blok `except`. Fungsi `main` memanggil fungsi `get_input()` untuk mendapatkan `input_provinsi` dan `jarak` dari *user*. Selanjutnya, program memanggil fungsi `layanan_mobil_dalam_kota` tiga kali dengan argumen yang berbeda, yaitu "maxim", "gojek", dan "Grab", bersama dengan `input_provinsi` dan `jarak`. Ini menghasilkan tiga variabel `harga_MaximCar`, `harga_GoCar`, dan `harga_GrabCar`, yang masing-masing berisi harga tarif mobil untuk layanan yang sesuai. Setelah itu, program mencetak harga mobil untuk setiap layanan menggunakan perintah `print`. Kemudian, blok `except` menangani pengecualian yang mungkin terjadi selama eksekusi program.

Jika terjadi `ProvinsiTidakTersediaError`, pesan kesalahan yang dihasilkan akan dicetak. Jika terjadi `ValueError`, pesan kesalahan yang dihasilkan juga akan dicetak. Jika terjadi pengecualian lainnya, pesan kesalahan yang dihasilkan akan dicetak.

Blok `if __name__ == "__main__":` memastikan bahwa fungsi `main` hanya dijalankan jika program ini dieksekusi langsung, bukan diimpor sebagai modul.

3.2.2 Modul Car Luar

```
1  import json
2
3  class ProvinsiTidakTersediaError(Exception):
4      pass
5
6  class AsalTidakTersedia(Exception):
7      pass
8
9  class TujuanTidakTersedia(Exception):
10     pass
```

‘Import json’ adalah sintaks dalam bahasa pemrograman yang berfungsi untuk mengimpor modul ‘json’ ke dalam program Python. Modul ‘json’ sendiri merupakan modul *built-in* dalam bahasa pemrograman Python yang berisikan fungsi-fungsi untuk bekerja dengan format data JSON (*JavaScript Object Notation*).

Kelas ‘ProvinsiTidakTersediaError’, ‘AsalTidakTersediaError’, dan ‘TujuanTidakTersedia’ merupakan turunan dari kelas bawaan Python yang disebut ‘Exception’. Kelas ‘Exception’ digunakan untuk membuat pengecualian atau kesalahan spesifik dalam program sehingga pada saat menjalankan kode program dapat menangani situasi *error* yang terjadi. Pernyataan ‘pass’ digunakan sebagai pernyataan kosong atau penanda di mana kode dapat ditambahkan di masa mendatang.

- **Def Get Input**

```
12  def get_input1():
13      input_provinsi = input("Input Provinsi Asal : ")
14      return input_provinsi
15
16  def get_input2():
17      input_asal = input("Kota Asal : ")
18      input_tujuan = input("Kota Tujuan : ")
19      return input_asal, input_tujuan
```

Fungsi pertama, ‘get_input1()’ didefinisikan sebagai fungsi untuk memperoleh input *user* berupa provinsi asal. Pada baris pertama, ‘input_provinsi = input("Input Provinsi Asal : ")’, *user* akan diminta untuk memasukkan provinsi asal melalui input yang diberikan dalam bentuk teks. Setelah *user* memasukkan nilai, nilai tersebut akan disimpan dalam variabel ‘input_provinsi’. Kemudian, nilai tersebut akan dikembalikan menggunakan pernyataan ‘return input_provinsi’.

Fungsi kedua, `get_input2()` didefinisikan sebagai fungsi untuk memperoleh input *user* berupa kota asal dan kota tujuan. Pada baris pertama, `input_asal = input("Kota Asal : ")`, *user* akan diminta untuk memasukkan kota asal. Pada baris berikutnya, `input_tujuan = input("Kota Tujuan : ")`, *user* akan diminta untuk memasukkan kota tujuan. Input ini juga berupa teks yang diberikan oleh *user*. Setelah kedua nilai tersebut dimasukkan, keduanya akan dikembalikan menggunakan pernyataan `return input_asal, input_tujuan`.

- **Def Layanan Mobil Luar Kota**

```
22 def layanan_mobil_luar_kota(layanan, input_provinsi, input_asal, input_tujuan):
23     with open("daftarharga.json", "r") as file1:
24         data1 = json.load(file1)
25
26     with open("jarakkotadalamprov.json", "r") as file2:
27         data2 = json.load(file2)
28
29     provinsi_data = data2.get(input_provinsi)
30
31     if provinsi_data is None:
32         raise ProvinsiTidakTersediaError("Provinsi yang anda input tidak terdapat di Pulau Jawa")
33
34     if input_asal not in provinsi_data:
35         raise AsalTidakTersedia("Asal tidak tersedia dalam provinsi ini")
36
37     if input_tujuan not in provinsi_data[input_asal]:
38         raise TujuanTidakTersedia("Tujuan tidak tersedia")
39
40     if input_provinsi == "Yogyakarta":
41         print("Mohon Maaf Layanan Tidak Tersedia Di Provinsi Ini")
42         return None
43
44     else:
45         for item in data1[layanan]:
46             if item["layanan"] == "Go Car" or item["layanan"] == "Grab Car" or item["layanan"] == "Car":
47                 provinsi_data2 = item["provinsi"]
48                 for asal in provinsi_data:
49                     for tujuan in provinsi_data[asal]:
50                         jarak = provinsi_data[asal][tujuan]
51                         if input_asal == asal and input_tujuan == tujuan:
52                             harga_minimum = provinsi_data2.get(input_provinsi)["harga_minimum"]
53                             harga_per_km = provinsi_data2.get(input_provinsi)["harga_per_km"]
54                             if jarak <= 3:
55                                 total_harga = harga_minimum
56                             else:
57                                 total_harga = harga_minimum + (harga_per_km * jarak)
58                             return total_harga
59         raise Exception("Input Anda Tidak Sesuai, Silakan Cek Kembali")
```

Program `layanan_mobil_luar_kota` didefinisikan sebagai sebuah fungsi yang digunakan untuk menghitung harga taksi berdasarkan jenis layanan, provinsi asal, kota asal, dan kota tujuan. Fungsi ini membaca dan mengolah data harga mobil/taksi dari file "daftarharga.json" lalu disimpan sebagai `data1`. Fungsi `layanan_mobil_luar_kota` juga membaca dan memproses daftar jarak kota dalam provinsi dari file "jarakkotadalamprov.json". Data yang dibaca disimpan ke dalam variabel `data2`.

Fungsi ini menerima tiga parameter yaitu ``layanan``, ``input_provinsi`` (provinsi asal/awal), ``input_asal`` (kota asal), dan ``input_tujuan`` (kota tujuan).

Menggunakan metode ``get()`` pada ``data2`` dengan parameter ``input_provinsi`` untuk mendapatkan data provinsi yang sesuai dengan input provinsi yang diberikan. Data provinsi tersebut disimpan dalam variabel ``provinsi_data``. Jika ``provinsi_data`` bernilai ``None``, program akan melempar ``ProvinsiTidakTersediaError`` dengan pesan yang sesuai, menandakan bahwa provinsi yang diinput tidak ditemukan di Pulau Jawa. Jika ``input_asal`` tidak terdapat dalam ``provinsi_data``, program akan melempar ``AsalTidakTersedia`` dengan pesan yang sesuai, menandakan bahwa asal yang diinput tidak tersedia dalam provinsi tersebut. Jika ``input_tujuan`` tidak terdapat dalam ``provinsi_data[input_asal]``, program akan melempar ``TujuanTidakTersedia`` dengan pesan yang sesuai, menandakan bahwa tujuan yang diinput tidak tersedia. Jika ``input_provinsi`` adalah "Yogyakarta", program akan mencetak pesan "Mohon Maaf Layanan Tidak Tersedia Di Provinsi Ini" dan mengembalikan ``None``.

Jika ``input_provinsi`` bukan "Yogyakarta", program akan melakukan iterasi melalui data harga yang sesuai dengan ``layanan`` dari ``data1``. Dalam iterasi tersebut, program akan memeriksa apakah layanan adalah "Go Car", "Grab Car", atau "Car". Jika ya, program akan mendapatkan data provinsi dari item tersebut. Program akan melakukan iterasi melalui data provinsi dan tujuan yang ditemukan dalam ``provinsi_data``. Untuk setiap pasangan asal dan tujuan, program akan mendapatkan jarak dari ``provinsi_data``. Jika ``input_asal`` dan ``input_tujuan`` cocok dengan pasangan asal dan tujuan saat ini, program akan mendapatkan harga minimum dan harga per kilometer dari data provinsi yang sesuai dengan ``input_provinsi``.

Jika jarak kurang dari atau sama dengan 3, program akan menghitung total harga dengan menggunakan harga minimum. Jika tidak, program akan menghitung total harga dengan menggunakan harga minimum ditambah harga per kilometer dikali jarak. Program akan mengembalikan ``total_harga`` yang dihitung. Jika tidak ada hasil yang ditemukan dalam iterasi, program akan melempar ``Exception`` dengan pesan "Input Anda Tidak Sesuai, Silakan Cek Kembali".

- **Def Main**

```

62 def main():
63     try:
64         input_provinsi = get_input1()
65         if input_provinsi == "Yogyakarta":
66             print("Mohon Maaf Layanan Tidak Tersedia Di Provinsi Ini")
67
68         else:
69             input_asal, input_tujuan = get_input2()
70             harga_MaximCar = layanan_mobil_luar_kota("maxim", input_provinsi, input_asal, input_tujuan)
71             harga_GoCar = layanan_mobil_luar_kota("gojek", input_provinsi, input_asal, input_tujuan)
72             harga_GrabCar = layanan_mobil_luar_kota("Grab", input_provinsi, input_asal, input_tujuan)
73             print("Harga Maxim Car : ", harga_MaximCar)
74             print("Harga Go Car : ", harga_GoCar)
75             print("Harga Grab Car : ", harga_GrabCar)
76
77     except ProvinsiTidakTersediaError as e:
78         print("Terjadi kesalahan : ", str(e))
79     except AsalTidakTersedia as e:
80         print("Terjadi kesalahan : ", str(e))
81     except TujuanTidakTersedia as e:
82         print("Terjadi kesalahan : ", str(e))
83     except Exception as e:
84         print("Terjadi kesalahan : ", str(e))
85
86 if __name__ == "__main__":
87     main()

```

Fungsi `main()` dimulai dengan menggunakan pernyataan `try-except` untuk menangani pengecualian yang mungkin terjadi saat menjalankan program. Fungsi `get_input1()` dipanggil untuk mendapatkan input provinsi dari *user*. Nilai input tersebut disimpan dalam variabel `input_provinsi`. Jika `input_provinsi` adalah "Yogyakarta", program akan mencetak pesan "Mohon Maaf Layanan Tidak Tersedia Di Provinsi Ini". Jika `input_provinsi` bukan "Yogyakarta", program akan melanjutkan dengan memanggil fungsi `get_input2()` untuk mendapatkan input asal dan tujuan dari *user*. Nilai input tersebut disimpan dalam variabel `input_asal` dan `input_tujuan`. Program akan memanggil fungsi `layanan_mobil_luar_kota()` tiga kali dengan argumen yang berbeda, yaitu "maxim", "gojek", dan "Grab". Fungsi ini akan menghitung harga mobil luar kota berdasarkan input yang diberikan. Harga dari layanan Maxim Car disimpan dalam variabel `harga_MaximCar`, harga dari layanan Go Car disimpan dalam variabel `harga_GoCar`, dan harga dari layanan Grab Car disimpan dalam variabel `harga_GrabCar`. Program akan mencetak harga dari masing-masing layanan menggunakan pernyataan `print()`. Jika terjadi pengecualian `ProvinsiTidakTersediaError`, `AsalTidakTersedia`, atau `TujuanTidakTersedia`, program akan menangkap pengecualian tersebut dan mencetak pesan kesalahan yang sesuai. Jika terjadi pengecualian lainnya, program akan menangkapnya dan mencetak pesan kesalahan umum. Bagian terakhir `if __name__ == "__main__":` memastikan

bahwa fungsi ``main()`` hanya dieksekusi jika program dieksekusi langsung (bukan diimpor sebagai modul).

3.2.3 Modul Box Dalam

```
1  import json
2
3  class ProvinsiTidakTersediaError(Exception):
4      pass
5
6  def get_input():
7      input_provinsi = input("Input Provinsi Asal : ")
8      try:
9          jarak = float(input("Jarak Tempuh (km): "))
10     except ValueError:
11         raise ValueError("Input jarak harus berupa angka")
12     return input_provinsi, jarak
13
14
15 def layanan_mobil_dalam_kota(layanan, input_provinsi, jarak):
16     with open("daftarharga.json", "r") as file1:
17         data1 = json.load(file1)
18
19     for item in data1[layanan]:
20         if item["layanan"] == "Go Car" or item["layanan"] == "Grab Car" or item["layanan"] == "Car":
21             provinsi_data = item["provinsi"]
22             if input_provinsi not in provinsi_data:
23                 raise ProvinsiTidakTersediaError("Provinsi yang anda input tidak terdapat di Pulau Jawa")
24             harga_minimum = provinsi_data.get(input_provinsi)["harga_minimum"]
25             harga_per_km = provinsi_data.get(input_provinsi)["harga_per_km"]
26             if jarak <= 3:
27                 total_harga = harga_minimum
28             else:
29                 total_harga = harga_minimum + (harga_per_km * jarak)
30             return total_harga
31
32     raise Exception("Input Anda Tidak Sesuai, Silakan Cek Kembali")
33
34 def main():
35     try:
36         input_provinsi, jarak = get_input()
37         harga_MaximCar = layanan_mobil_dalam_kota("maxim", input_provinsi, jarak)
38         harga_GoCar = layanan_mobil_dalam_kota("gojek", input_provinsi, jarak)
39         harga_GrabCar = layanan_mobil_dalam_kota("Grab", input_provinsi, jarak)
40         print("Harga Maxim Car : ", harga_MaximCar)
41         print("Harga Go Car : ", harga_GoCar)
42         print("Harga Grab Car : ", harga_GrabCar)
43
44     except ProvinsiTidakTersediaError as e:
45         print("Terjadi kesalahan : ", str(e))
46     except ValueError as e:
47         print("Terjadi kesalahan : ", str(e))
48     except Exception as e:
49         print("Terjadi Kesalahan : ", str(e))
50
51 if __name__ == "__main__":
52     main()
53
```

Modul diatas adalah sebuah program yang menghitung harga jasa pengiriman barang berdasarkan ukuran, provinsi, dan jarak. Program ini menggunakan file JSON untuk menyimpan data harga pengiriman. Fungsi `get_input()` digunakan untuk mengambil input dari *user*, sementara fungsi `layanan_box_dalam_kota()` menghitung harga pengiriman. Program utama `main()` memanggil fungsi-fungsi ini dan menampilkan hasilnya.

- `'import json'`, modul ini digunakan untuk membaca file JSON yang berisi data harga pengiriman.
- `'class UkuranError(Exception)'` Mendefinisikan kelas `'UkuranError'` dan `'ProvinsiTidakTersediaError'` sebagai subclass dari kelas `'Exception'`. Kelas ini digunakan untuk menghasilkan pengecualian khusus yang dapat ditangkap dan diatasi secara terpisah.
- Fungsi `'get_input()'` digunakan untuk mengambil input dari *user*, seperti ukuran box, provinsi asal, dan jarak pengiriman. Data ukuran box dan harga pengiriman ditampilkan dalam bentuk tabel yang diambil dari string `'data'`.
- `'layanan_box_dalam_kota()'` digunakan untuk menghitung harga pengiriman berdasarkan layanan, ukuran box, provinsi, dan jarak. Fungsi ini membaca data harga pengiriman dari file JSON "daftarharga.json" dan mencari harga yang sesuai dengan parameter yang diberikan.
- Fungsi `'main()'` merupakan fungsi utama yang akan dijalankan saat program dijalankan. Fungsi ini memanggil fungsi `'get_input()'` untuk mendapatkan input dari *user*, kemudian memanggil fungsi `'layanan_box_dalam_kota()'` untuk menghitung harga pengiriman dengan layanan yang berbeda. Harga-harga yang dihitung kemudian ditampilkan.
- Blok `'try-except'` digunakan untuk menangkap pengecualian yang mungkin terjadi selama eksekusi program. Jika terjadi pengecualian `'UkuranError'`, `'ValueError'`, atau `'ProvinsiTidakTersediaError'`, pesan kesalahan akan ditampilkan. Jika terjadi pengecualian lainnya, pesan kesalahan umum akan ditampilkan.
- Terakhir, blok `'if __name__ == "__main__":'` digunakan untuk memastikan bahwa fungsi `'main()'` hanya dijalankan jika file ini dieksekusi sebagai program utama, bukan sebagai modul yang diimpor oleh program lain.

3.2.4 Modul Box Luar

```
1  import json
2
3  class ProvinsiTidakTersediaError(Exception):
4      pass
5
6  class AsalTidakTersedia(Exception):
7      pass
8
9  class TujuanTidakTersedia(Exception):
10     pass
11
12 class ukuranerror(Exception):
13     pass
14
15 def get_input():
16     data = '''small = 200 x 130 x 120 cm (berat max. 1000kg)
17     medium = 200 x 130 x 130 cm (berat max. 2000kg)
18     large = 300 x 160 x 130 cm (berat max. 2000kg)'''
19     rows = data.split('\n')
20     print("| {:<7s} | {:^18s} | {:^17s} |".format("Ukuran", "Dimensi", "Berat Maks.))
21     print("-" * 40)
22     for row in rows:
23         row_data = row.split('=')
24         ukuran = row_data[0].strip()
25         dimensi = row_data[1].split(' ')[0].strip()
26         berat = row_data[1].split(' ')[1].split(' ')[0].strip()
27         print("| {:<7s} | {:^15s} | {:^12s} |".format(ukuran, dimensi, berat))
28         print("-" * 40)
29
30     input_ukuran = input("Ukuran Box : ")
31     input_provinsi = input("Input provinsi Asal : ")
32     input_asal = input("Kota Asal : ")
33     input_tujuan = input("Kota Tujuan : ")
34     return input_ukuran, input_provinsi, input_asal, input_tujuan
35
36
37 def layananboxluarkota(layanan, input_ukuran, input_provinsi, input_asal, input_tujuan):
38     with open("daftarharga.json", "r") as file1:
39         data1 = json.load(file1)
40
41     with open("jarakkotaluarprov.json", "r") as file2:
42         data2 = json.load(file2)
43
44     provinsi_data = data2.get(input_provinsi)
45
46     if provinsi_data is None:
47         raise ProvinsiTidakTersediaError("Provinsi yang anda input tidak terdapat di Pulau Jawa")
48
49     if input_asal not in provinsi_data:
50         raise AsalTidakTersedia("Asal tidak tersedia dalam provinsi ini")
51
52     if input_tujuan not in provinsi_data[input_asal]:
53         raise TujuanTidakTersedia("Tujuan tidak tersedia")
54
55     for asal in provinsi_data:
56         for tujuan in provinsi_data[asal]:
57             jarak = provinsi_data[asal][tujuan]
```

```

59
60         for item in data[layanan]:
61             if item.get("layanan") == "Maxim Box" or item.get("layanan") == "Go Box" or item.get("layanan") == "Grab Instant":
62                 ukuran_data = item.get("ukuran", {}).get(input_ukuran)
63                 if ukuran_data is None:
64                     raise ukuranerror("Ukuran tidak sesuai")
65                 if ukuran_data is not None and input_asal == asal and input_tujuan == tujuan:
66                     harga_minimum = ukuran_data.get("provinsi", {}).get(input_provinsi, {}).get("harga_minimum")
67                     harga_per_km = ukuran_data.get("provinsi", {}).get(input_provinsi, {}).get("harga_per_km")
68                     if harga_minimum is not None and harga_per_km is not None:
69                         if jarak <= 3:
70                             total_harga = harga_minimum
71                         else:
72                             total_harga = harga_minimum + (harga_per_km * jarak)
73                     return total_harga
74
75         raise Exception("Input Anda Tidak Sesuai, Silakan Cek Kembali")
76
77 def main():
78     try :
79         input_ukuran, input_provinsi, input_asal, input_tujuan = get_input()
80         harga_MaximBox = layananboxluarkota("maxim", input_ukuran, input_provinsi, input_asal, input_tujuan)
81         harga_GoBox = layananboxluarkota("gojek", input_ukuran, input_provinsi, input_asal, input_tujuan)
82         harga_GrabInstant = layananboxluarkota("Grab", input_ukuran, input_provinsi, input_asal, input_tujuan)
83         print("Harga Maxim Box : ", harga_MaximBox)
84         print("Harga Go Box : ", harga_GoBox)
85         print("Harga Grab Instant : ", harga_GrabInstant)
86
87     except ukuranerror as e:
88         print("Terjadi kesalahan : ", str(e))
89     except ProvinsiTidakTersediaError as e:
90         print("Terjadi kesalahan : ", str(e))
91     except AsalTidakTersedia as e:
92         print("Terjadi kesalahan : ", str(e))
93     except TujuanTidakTersedia as e:
94         print("Terjadi kesalahan : ", str(e))
95     except Exception as e:
96         print("Terjadi kesalahan : ", str(e))
97
98 if __name__ == "__main__":
99     main()
100

```

Modul di atas adalah implementasi program untuk menghitung harga layanan pengiriman barang menggunakan layanan Maxim Box, Go Box, dan Grab Instant.

- Modul `json` diimpor untuk membaca file JSON yang berisi data harga dan jarak antar kota.
- `ProvinsiTidakTersediaError` Digunakan untuk menunjukkan bahwa provinsi yang diinput tidak tersedia di Pulau Jawa.
- `AsalTidakTersedia` Digunakan untuk menunjukkan bahwa kota asal yang diinput tidak tersedia dalam provinsi tersebut.
- `TujuanTidakTersedia` Digunakan untuk menunjukkan bahwa kota tujuan yang diinput tidak tersedia.
- `ukuranerror` Digunakan untuk menunjukkan bahwa ukuran yang diinput tidak sesuai dengan ukuran yang tersedia.
- `get_input()` adalah fungsi yang digunakan untuk mendapatkan input dari user. Fungsi ini akan mencetak daftar ukuran dan dimensi box yang tersedia, kemudian

meminta input ukuran box, provinsi asal, kota asal, dan kota tujuan. Fungsi ini mengembalikan input yang diberikan oleh *user*.

- ``layanandboxluarkota()`` adalah fungsi yang menghitung harga layanan pengiriman berdasarkan input yang diberikan. Fungsi ini membaca data harga dari file JSON "daftarharga.json" dan data jarak antar kota dari file JSON "jarakkotaluarprov.json". Fungsi ini juga melakukan pengecekan terhadap input yang diberikan, seperti memeriksa apakah provinsi, asal, dan tujuan tersedia dalam data yang ada. Jika data input valid, maka fungsi ini akan menghitung harga layanan berdasarkan jarak antar kota dan ukuran box yang dipilih. Fungsi ini mengembalikan harga total layanan.
- ``main()`` adalah fungsi utama yang menjalankan program. Fungsi ini memanggil ``get_input()`` untuk mendapatkan input dari *user*, kemudian memanggil ``layanandboxluarkota()`` untuk menghitung harga layanan menggunakan tiga layanan yang berbeda (Maxim Box, Go Box, dan Grab Instant). Hasil perhitungan harga layanan akan dicetak ke layar. Jika terjadi kesalahan dalam input atau pemrosesan, kesalahan tersebut akan ditangkap dan pesan kesalahan akan dicetak.
- Pada blok ``if __name__ == "__main__":``, fungsi ``main()`` akan dipanggil jika script ini dieksekusi secara langsung (bukan diimpor sebagai modul).

3.2.5 Modul Ride

```
1  import json
2
3  class ProvinsiTidakTersediaError(Exception):
4      pass
5
6  def get_input():
7      input_provinsi = input("Input Provinsi Asal : ")
8      try:
9          jarak = float(input("Jarak Tempuh (km): "))
10     except ValueError:
11         raise ValueError("Input jarak harus berupa angka")
12     return input_provinsi, jarak
13
14 def layanan_motor(layanan, input_provinsi, jarak):
15     with open("daftarharga.json", "r") as file1:
16         data1 = json.load(file1)
17
18     for item in data1[layanan]:
19         if item["layanan"] == "Go Ride" or item["layanan"] == "Grab Bike" or item["layanan"] == "Bike":
20             provinsi_data = item["provinsi"]
21             if input_provinsi not in provinsi_data:
22                 raise ProvinsiTidakTersediaError("Provinsi yang anda input tidak terdapat di Pulau Jawa")
23             harga_minimum = provinsi_data.get(input_provinsi)["harga_minimum"]
24             harga_per_km = provinsi_data.get(input_provinsi)["harga_per_km"]
25             if jarak <= 3:
26                 total_harga = harga_minimum
27             else:
28                 total_harga = harga_minimum + (harga_per_km * jarak)
29             return total_harga
30
31     raise Exception("Input Anda Tidak Sesuai, Silakan Cek Kembali")
```

```

1  import json
2
3  class ProvinsiTidakTersediaError(Exception):
4      pass
5
6  def get_input():
7      input_provinsi = input("Input Provinsi Asal : ")
8      try:
9          jarak = float(input("Jarak Tempuh (km): "))
10     except ValueError:
11         raise ValueError("Input jarak harus berupa angka")
12     return input_provinsi, jarak
13
14 def layanan_motor(layanan, input_provinsi, jarak):
15     with open("daftarharga.json", "r") as file1:
16         data1 = json.load(file1)
17
18     for item in data1[layanan]:
19         if item["layanan"] == "Go Ride" or item["layanan"] == "Grab Bike" or item["layanan"] == "Bike":
20             provinsi_data = item["provinsi"]
21             if input_provinsi not in provinsi_data:
22                 raise ProvinsiTidakTersediaError("Provinsi yang anda input tidak terdapat di Pulau Jawa")
23             harga_minimum = provinsi_data.get(input_provinsi)["harga_minimum"]
24             harga_per_km = provinsi_data.get(input_provinsi)["harga_per_km"]
25             if jarak <= 3:
26                 total_harga = harga_minimum
27             else:
28                 total_harga = harga_minimum + (harga_per_km * jarak)
29             return total_harga
30
31     raise Exception("Input Anda Tidak Sesuai, Silakan Cek Kembali")

```

Kode pada modul ini merupakan penerapan dari suatu program yang bekerja sebagai layanan pengiriman dengan sepeda motor di Pulau Jawa. Program ini nantinya akan meminta *user* untuk menginput provinsi asal maupun jarak yang berbentuk kilometer. Lalu, program ini akan menghitung biaya pengiriman berdasarkan tiga jenis layanan sepeda motor, yaitu Grab Bike, Maxim Bike, Go Ride.

Kode-kode di atas akan bekerja, seperti adanya ‘Exception’ untuk mendefinisikan ‘ProvinsiTidakTersediaError’ yang berarti provinsi yang diinput tidak ada di Pulau Jawa. Meminta *user* untuk menginput jarak dan memastikan bahwa yang diinput berupa angka. Membuat kode apabila provinsi tidak ditemukan maka ‘ProvinsiTidakTersediaError’ akan diangkat namun jika provinsi yang dituju itu ditemukan fungsi akan menghitung total harga pengiriman berdasarkan harga minimum dan harga per kilometer yang telah tercantum pada daftar harga. Menjalankan fungsi utama program guna mendapatkan input *user* lalu memanggil ‘layanan_motor’ untuk menghitung harga pengiriman dari setiap layanan yang ada lalu dicetak. Melalui kode pada modul ini akan diperoleh perkiraan harga pengiriman menggunakan layanan sepeda motor yang didapati dari provinsi asal dan juga jarak tempuh yang telah diinput *user*.

3.2.6 Modul Send

```
1 import json
2
3 class ProvinsiTidakTersediaError(Exception):
4     pass
```

‘Import json’ adalah sintaks dalam bahasa pemrograman yang berfungsi untuk mengimpor modul ‘json’ ke dalam program Python. Modul ‘json’ sendiri merupakan modul *built-in* dalam bahasa pemrograman Python yang berisikan fungsi-fungsi untuk bekerja dengan format data JSON (*JavaScript Object Notation*).

Kelas ‘ProvinsiTidakTersediaError’ merupakan turunan dari kelas bawaan Python yang disebut ‘Exception’. Kelas ‘Exception’ digunakan untuk membuat pengecualian atau kesalahan spesifik dalam program sehingga pada saat menjalankan kode program dapat menangani situasi *error* yang terjadi. Pernyataan ‘pass’ digunakan sebagai pernyataan kosong atau penanda di mana kode dapat ditambahkan di masa mendatang.

- **Def Get Input**

```
6 def get_input():
7     input_provinsi = input("Input Provinsi Asal : ")
8     try:
9         jarak = float(input("Jarak Pengiriman (km): "))
10    except ValueError:
11        raise ValueError("Input jarak harus berupa angka")
12    return input_provinsi, jarak
```

‘get_input()’ didefinisikan sebagai fungsi untuk memperoleh input *user* berupa provinsi asal dan jarak pengiriman. Pertama-tama, *user* diminta untuk memasukkan nama provinsi asal. Nilai yang dimasukkan *user* akan disimpan dalam variabel ‘input_provinsi’. Selanjutnya, *user* diminta untuk memasukkan jarak pengiriman dalam satuan kilometer. Nilai yang dimasukkan *user* akan disimpan dalam variabel ‘jarak’. Kemudian fungsi ‘Try-except’ digunakan untuk memastikan bahwa input jarak berbentuk angka atau tipe data ‘float’. Jika *user* memasukkan nilai yang tidak dapat dikonversi menjadi tipe data ‘float’, maka fungsi akan menghasilkan ‘ValueError’ dengan pesan “Input jarak harus berupa angka”. Dengan demikian, jika tidak terdapat kesalahan pada input *user*, nilai ‘input_provinsi’ dan ‘jarak’ akan dikembalikan sebagai hasil dari fungsi ‘get_input()’

- **Def layanan send**

```
15 def layanan_send(layanan, input_provinsi, jarak):
16     with open("daftarharga.json", "r") as file1:
17         data1 = json.load(file1)
18
19
20     for item in data1[layanan]:
21         if item["layanan"] == "Go Send" or item["layanan"] == "Grab Same Day" or item["layanan"] == "Maxim Delivery":
22             provinsi_data = item["provinsi"]
23             if input_provinsi not in provinsi_data:
24                 raise ProvinsiTidakTersediaError("Provinsi yang anda input tidak terdapat di Pulau Jawa")
25             harga_minimum = provinsi_data.get(input_provinsi)["harga_minimum"]
26             harga_per_km = provinsi_data.get(input_provinsi)["harga_per_km"]
27             if jarak <= 3:
28                 total_harga = harga_minimum
29             else:
30                 total_harga = harga_minimum + (harga_per_km * jarak)
31             return total_harga
32
33     raise Exception("Input Anda Tidak Sesuai, Silakan Cek Kembali")
```

Program `layanan_send` didefinisikan sebuah fungsi yang digunakan untuk menghitung harga pengiriman berdasarkan jenis layanan, provinsi asal, dan jarak pengiriman. Fungsi ini membaca data harga pengiriman dari file "daftarharga.json" dan memprosesnya. Fungsi ini menerima tiga parameter: `layanan` (jenis layanan pengiriman), `input_provinsi` (provinsi tujuan), dan `jarak` (jarak pengiriman).

Pertama, fungsi membaca data harga pengiriman dari file "daftarharga.json" dan menyimpannya dalam variabel `data1` menggunakan fungsi `json.load()`. Selanjutnya, fungsi melakukan iterasi melalui setiap elemen dalam daftar harga yang sesuai dengan jenis layanan yang diberikan (`Go Send`, `Grab Same Day`, atau `Maxim Delivery`). Jika provinsi tujuan yang diinputkan tidak terdapat dalam daftar provinsi yang tersedia, fungsi akan menghasilkan pengecualian `ProvinsiTidakTersediaError` dengan pesan "Provinsi yang anda input tidak terdapat di Pulau Jawa".

Jika provinsi tujuan tersedia, fungsi akan mengambil harga minimum dan harga per kilometer dari data provinsi tersebut. Jika jarak pengiriman kurang dari atau sama dengan 3 kilometer, fungsi akan menghitung total harga pengiriman berdasarkan harga minimum. Jika jarak pengiriman lebih dari 3 kilometer, fungsi akan menghitung total harga pengiriman dengan menambahkan harga minimum dengan perkalian harga per kilometer dengan jarak pengiriman.

Terakhir, fungsi akan mengembalikan total harga pengiriman. Jika tidak ada layanan yang sesuai dengan input yang diberikan, fungsi akan menghasilkan pengecualian `Exception` dengan pesan "Input Anda Tidak Sesuai, Silakan Cek Kembali".

- **Def Main**

```
35 ∨ def main():
36 ∨     try:
37         input_provinsi, jarak = get_input()
38         harga_MaximDelivery = layanan_send("maxim", input_provinsi, jarak)
39         harga_GoSend = layanan_send("gojek", input_provinsi, jarak)
40         harga_GrabSameDay = layanan_send("Grab", input_provinsi, jarak)
41         print("Harga Maxim Delivery : ", harga_MaximDelivery)
42         print("Harga Go Send : ", harga_GoSend)
43         print("Harga Grab Same Day : ", harga_GrabSameDay)
44
45 ∨     except ProvinsiTidakTersediaError as e:
46         print("Terjadi kesalahan : ", str(e))
47 ∨     except ValueError as e:
48         print("Terjadi kesalahan : ", str(e))
49 ∨     except Exception as e:
50         print("Terjadi kesalahan : ", str(e))
51
52
53 ∨ if __name__ == "__main__":
54     main()
```

Program `main` adalah titik masuk utama dari program. Ini menggunakan fungsi `layanan_send` untuk menghitung harga pengiriman dengan berbagai layanan, provinsi asal, dan jarak pengiriman yang diinputkan.

Pada awalnya, program mencoba menjalankan blok kode dalam blok `try`. Jika terjadi pengecualian selama eksekusi, program menangani pengecualian tersebut dalam blok `except`. Fungsi `main` memanggil fungsi `get_input()` untuk mendapatkan `input_provinsi` dan `jarak` dari *user*. Selanjutnya, program memanggil fungsi `layanan_send` tiga kali dengan argumen yang berbeda, yaitu "maxim", "gojek", dan "Grab", bersama dengan `input_provinsi` dan `jarak`. Ini menghasilkan tiga variabel `harga_MaximDelivery`, `harga_GoSend`, dan `harga_GrabSameDay`, yang masing-masing berisi harga pengiriman untuk layanan yang sesuai. Setelah itu, program mencetak harga pengiriman untuk setiap layanan menggunakan perintah `print`. Kemudian, blok `except` menangani pengecualian yang mungkin terjadi selama eksekusi program.

Jika terjadi `ProvinsiTidakTersediaError`, pesan kesalahan yang dihasilkan akan dicetak. Jika terjadi `ValueError`, pesan kesalahan yang dihasilkan juga akan dicetak. Jika terjadi pengecualian lainnya, pesan kesalahan yang dihasilkan akan dicetak.

Blok `if __name__ == "__main__"` memastikan bahwa fungsi `main` hanya dijalankan jika program ini dieksekusi langsung, bukan diimpor sebagai modul

3.3 Program Utama

Bagian ini menjelaskan mengenai arti syntax program rekomendasi harga untuk layanan transportasi dan kurir ojek online di Pulau Jawa. Program ini menggunakan beberapa modul seperti modulsend, modulcardalam, modulcarluar, modulride, modulboxdalam, modulboxluar, dan sys. Selain itu, program juga menggunakan modul colorama untuk memberikan tampilan teks dengan warna yang berbeda. Ketika digunakan, program akan menampilkan daftar layanan yang tersedia seperti layanan transportasi dan layanan kurir. *User* dapat memilih jenis layanan yang mereka butuhkan dan program akan memberikan rekomendasi harga untuk layanan tersebut.

3.3.1 Syntax Import Modul

```
import modulsend as ms
import modulcardalam as mcd
import modulcarluar as mcl
import modulride as mr
import modulsend as ms
import modulboxdalam as mbd
import modulboxluar as mbl
import sys
from colorama import init, Fore

init(autoreset=True)
```

Syntax tersebut adalah impor modul-modul yang diperlukan dalam program. Modul-modul yang diimpor antara lain: "modulsend" yang diimpor dengan alias "ms", "modulcardalam" diimpor dengan alias "mcd", "modulcarluar" diimpor dengan alias "mcl", "modulride" diimpor dengan alias "mr", "modulboxdalam" diimpor dengan alias "mbd", "modulboxluar" diimpor dengan alias "mbl". Selain itu, modul "sys" dan fungsi "init" dan "Fore" dari modul "colorama" juga diimpor. Fungsi "init(autoreset=True)" menginisialisasi modul "colorama" dengan pengaturan agar warna teks diatur ulang setiap kali dipanggil. Pernyataan terakhir, "from colorama import init, Fore", mengimpor fungsi "init" dan kelas "Fore" dari modul "colorama". Fungsi "init" digunakan untuk menginisialisasi "colorama" dengan opsi otomatis ulang (autoreset=True), sedangkan kelas "Fore" digunakan untuk memberikan warna pada teks yang ditampilkan di terminal. Kode ini menyediakan akses ke berbagai modul dan fungsi yang dibutuhkan untuk menjalankan program yang menggunakan modul-modul tersebut.

3.3.2 Def Main

```
def main():
    print("=====")
    print(Fore.YELLOW + "===== PROGRAM REKOMENDASI HARGA OJEK ONLINE =====")
    print(Fore.YELLOW + "Selamat datang di layanan transportasi dan kurir ojek online!")
    print(Fore.YELLOW + "Program Ini Akan Memberikan Anda Rekomendasi \nLayanan Jasa Ojek Online Di Pulau Ja")
    print("\nSilakan pilih jenis layanan yang Anda butuhkan:")
    print(Fore.CYAN + "1. Layanan Transportasi")
    print(Fore.CYAN + "2. Layanan Kurir")
    print(Fore.RED + "0. Keluar")

    while True:
        pilihan = input(Fore.GREEN + "\nMasukkan pilihan Anda (1/2/0): ")
        print(Fore.RESET + "=====")

        if pilihan == "1":
            layanan_transportasi()
        elif pilihan == "2":
            layanan_kurir()
        elif pilihan == "0":
            print(Fore.YELLOW + "Terima kasih telah menggunakan layanan kami. Sampai jumpa!")
            sys.exit()
        else:
            print(Fore.RED + "Pilihan tidak valid. Silakan coba lagi.")
        print("=====")
```

Syntax di atas adalah sebuah fungsi utama (main) yang merupakan bagian utama dari program. Ketika fungsi “main()” dijalankan, beberapa pernyataan print akan dieksekusi dan menghasilkan output yang ditampilkan di layar. Pertama, terdapat beberapa pernyataan print yang digunakan untuk mencetak judul program, pesan selamat datang, dan deskripsi singkat tentang program rekomendasi harga ojek online. Pesan-pesan ini ditampilkan dengan menggunakan warna teks yang berbeda, seperti kuning, biru, dan merah, yang diberikan oleh kelas “Fore” dari modul “colorama”. Selanjutnya, terdapat loop “while True” yang akan berjalan terus menerus sampai ada perintah untuk keluar. Di dalam loop ini, *user* diminta untuk memasukkan pilihan (1, 2, atau 0) melalui pernyataan input. Setelah itu, terdapat kondisi if-elif-else yang akan mengevaluasi pilihan yang dimasukkan oleh *user* dan menjalankan fungsi-fungsi terkait. Jika *user* memilih 1, fungsi “layanan_transportasi()” akan dipanggil. Jika *user* memilih 2, fungsi “layanan_kurir()” akan dipanggil. Jika *user* memilih 0, program akan mengeluarkan pesan terima kasih dan kemudian keluar dari program menggunakan “sys.exit()”. Jika *user* memasukkan pilihan yang tidak valid, pesan kesalahan akan ditampilkan. Setelah setiap iterasi loop, ada beberapa pernyataan print yang mencetak garis pemisah untuk memisahkan setiap bagian output. Secara keseluruhan, syntax ini mengatur jalannya program rekomendasi harga ojek online dengan menampilkan pesan-pesan dan memproses pilihan yang dimasukkan oleh *user*.

3.3.3 Def Layanan Kurir

```
40 def layanan_kurir():
41     print(Fore.RESET + "\n=====")
42     print(Fore.YELLOW + "===== DAFTAR LAYANAN KURIR OJEK ONLINE YANG TERSEDIA =====")
43     print("\nBerikut Layanan Kurir Ojek Online Yang Tersedia")
44     print(Fore.CYAN + "1. Dalam Kota")
45     print(Fore.CYAN + "2. Luar Kota")
46     print(Fore.CYAN + "3. Kembali")
47     print(Fore.RED + "0. Exit")
48     input_kurir = input(Fore.GREEN + "\nMasukkan Layanan (1/2/3/0): ")
49     print(Fore.RESET + "=====")
50
51     while True:
52         if input_kurir == "1":
53             layanan_kurir_dalam_kota()
54         elif input_kurir == "2":
55             layanan_kurir_luar_kota()
56         elif input_kurir == "3":
57             main()
58         elif input_kurir == "0":
59             print(Fore.YELLOW + "Terima kasih telah menggunakan layanan kami. Sampai jumpa!")
60             sys.exit()
61         else:
62             print(Fore.RED + "Pilihan tidak valid. Silakan coba lagi.")
63             break
```

Syntax di atas adalah sebuah fungsi bernama “layanan_kurir()” yang akan dieksekusi ketika dipanggil. Fungsi ini mengatur tampilan dan logika program untuk layanan kurir dalam program rekomendasi harga ojek online. Pertama, terdapat beberapa pernyataan print yang akan mencetak judul dan pesan untuk daftar layanan kurir ojek online yang tersedia. Pesan-pesan ini menggunakan warna teks yang berbeda, seperti kuning dan biru, yang diberikan oleh kelas “Fore” dari modul “colorama”. Setelah itu, *user* diminta untuk memasukkan pilihan layanan kurir (1, 2, 3, atau 0) melalui pernyataan input. Setelah *user* memasukkan pilihannya, garis pemisah akan dicetak menggunakan pernyataan print. Kemudian, terdapat loop “while True” yang akan berjalan terus menerus sampai ada perintah untuk keluar. Di dalam loop ini, terdapat kondisi if-elif-else yang akan mengevaluasi pilihan yang dimasukkan oleh *user*. Jika *user* memilih 1, fungsi “layanan_kurir_dalam_kota()” akan dipanggil. Jika *user* memilih 2, fungsi “layanan_kurir_luar_kota()” akan dipanggil. Jika *user* memilih 3, fungsi “main()” akan dipanggil untuk kembali ke menu utama. Jika *user* memilih 0, program akan mengeluarkan pesan terima kasih dan kemudian keluar dari program menggunakan “sys.exit()”. Jika *user* memasukkan pilihan yang tidak valid, pesan kesalahan akan ditampilkan dan loop akan dihentikan menggunakan pernyataan “break”. Secara keseluruhan, syntax ini mengatur tampilan dan logika program untuk layanan kurir dalam program rekomendasi harga ojek online. *User* dapat memilih jenis layanan kurir yang

tersedia, dan program akan memproses pilihan mereka sesuai dengan fungsi-fungsi yang relevan atau memberikan pesan kesalahan jika pilihan tidak valid.

3.3.4 Def Layanan Kurir Dalam Kota

```
def layanan_kurir_dalam_kota():
    print(Fore.RESET + "\n=====")
    print(Fore.YELLOW + "===== LAYANAN KURIR DALAM KOTA =====")
    print("\nBerikut Layanan Kurir Dalam Kota Yang Tersedia")
    print(Fore.CYAN + "1. Reguler (Motor)")
    print(Fore.CYAN + "2. Box (Mobil)")
    print(Fore.CYAN + "3. Kembali")
    print(Fore.RED + "0. Exit")
    input_dalam_kota = input(Fore.GREEN + "\nPilih Layanan (1/2/3/0) : ")
    print(Fore.RESET + "\n=====")

    while True:
        if input_dalam_kota == "1":
            print(Fore.RESET + "\n=====")
            print(Fore.YELLOW + "===== LAYANAN KURIR REGULER =====")
            print("\nPilih Sesuai Kebutuhan Anda")
            try:
                input_provinsi, jarak = ms.get_input()
                harga_MaximDelivery = ms.layanan_send("maxim", input_provinsi.title(), jarak)
                harga_GoSend = ms.layanan_send("gojek", input_provinsi.title(), jarak)
                harga_GrabSameDay = ms.layanan_send("Grab", input_provinsi.title(), jarak)
                print("\n=====")

                print("\n=====")
                print(Fore.BLUE + "Berikut Adalah Beberapa Perbandingan Harga \nLayanan Yang Dapat Kami Berikan\n")
                print("Harga Kurir Standar dengan Maxim Delivery      : Rp", harga_MaximDelivery)
                print("Harga Kurir Standar dengan GoSend                        : Rp", harga_GoSend)
                print("Harga Kurir Standar dengan Grab Same Day                 : Rp", harga_GrabSameDay)
                print()

                if harga_MaximDelivery < harga_GoSend and harga_MaximDelivery < harga_GrabSameDay:
                    print(Fore.GREEN + "Rekomendasi Jasa Layanan Kurir Termurah Adalah \nMaxim Delivery")
                elif harga_GrabSameDay < harga_MaximDelivery and harga_GrabSameDay < harga_GoSend:
                    print(Fore.GREEN + "Rekomendasi Jasa Layanan Kurir Termurah Adalah \nGrab Same Day")
                elif harga_GoSend < harga_MaximDelivery and harga_GoSend < harga_GrabInstant:
                    print(Fore.GREEN + "Rekomendasi Jasa Layanan Kurir Termurah Adalah \nGo Send")
                print("=====")

                layananlain = input(Fore.YELLOW + "\nApakah Anda Ingin Mengecek Layanan Lain? (Ya/Tidak) : ")
                if layananlain.capitalize() == "Ya":
                    layanan_kurir_dalam_kota()
                elif layananlain.capitalize() == "Tidak":
                    print(Fore.GREEN + "Terima kasih telah menggunakan layanan kami. Sampai jumpa!")
                    sys.exit()
                else:
                    print(Fore.RED + "Pilihan tidak valid. Silakan coba lagi.")
                    break

            except ms.ProvinsiTidakTersediaError as e:
                print("Terjadi kesalahan : ", str(e))
            except ValueError as e:
                print("Terjadi kesalahan : ", str(e))
            except Exception as e:
                print("Terjadi kesalahan : ", str(e))
```

```

elif input_dalam_kota == "2":
    print(Fore.RESET + "\n=====")
    print(Fore.YELLOW + "===== LAYANAN KURIR BOX =====")
    print("\nPilih Sesuai Kebutuhan")
    try:
        input_ukuran, input_provinsi, jarak = mbd.get_input()
        harga_MaximBox = mbd.layanan_box_dalam_kota("maxim", input_ukuran.lower(), input_provinsi.title(), jarak)
        harga_GoBox = mbd.layanan_box_dalam_kota("gojek", input_ukuran.lower(), input_provinsi.title(), jarak)
        harga_GrabInstant = mbd.layanan_box_dalam_kota("Grab", input_ukuran.lower(), input_provinsi.title(), jarak)
        print("\n=====")

        print("\n=====")
        print(Fore.BLUE + "Berikut Adalah Beberapa Perbandingan Harga \nLayanan Yang Dapat Kami Berikan\n")
        print("Harga Kurir Box Dalam Kota dengan Maxim Box      : Rp", harga_MaximBox)
        print("Harga Kurir Box Dalam Kota dengan Go Box              : Rp", harga_GoBox)
        print("Harga Kurir Box Dalam Kota dengan Grab Instant         : Rp", harga_GrabInstant)
        print()

        if harga_MaximBox < harga_GoBox and harga_MaximBox < harga_GrabInstant:
            print(Fore.GREEN + "Rekomendasi Jasa Layanan Termurah Adalah \nMaxim Box")
        elif harga_GrabInstant < harga_MaximBox and harga_GrabInstant < harga_GoBox:
            print(Fore.GREEN + "Rekomendasi Jasa Layanan Termurah Adalah \nGrab Instant")
        elif harga_GoBox < harga_MaximBox and harga_GoBox < harga_GrabInstant:
            print(Fore.GREEN + "Rekomendasi Jasa Layanan Termurah Adalah \nGo Box")
        print("=====")

        layananlain = input(Fore.YELLOW + "\nApakah Anda Ingin Mengecek Layanan Lain? (Ya/Tidak) : ")
        if layananlain.capitalize() == "Ya":
            layanan_kurir_dalam_kota()
        elif layananlain.capitalize() == "Tidak":
            print(Fore.GREEN + "Terima kasih telah menggunakan layanan kami. Sampai jumpa!")
            sys.exit()
        else:
            print(Fore.RED + "Pilihan tidak valid. Silakan coba lagi.")
            break

```

```

except mbd.UkuranError as e:
    print("Terjadi kesalahan:", str(e))
except ValueError as e:
    print("Terjadi kesalahan:", str(e))
except mbd.ProvinsiTidakTersediaError as e:
    print("Terjadi kesalahan:", str(e))
except Exception as e:
    print("Terjadi kesalahan:", str(e))

elif input_dalam_kota == "3":
    layanan_kurir()
elif input_dalam_kota == "0":
    print(Fore.GREEN + "Terima kasih telah menggunakan layanan kami. Sampai jumpa!")
    sys.exit()

```

Syntax di atas adalah sebuah fungsi bernama “`layanan_kurir_dalam_kota()`” yang akan dieksekusi ketika dipanggil. Fungsi ini mengatur tampilan dan logika program untuk layanan kurir dalam kota dalam program rekomendasi harga ojek online. Pertama, terdapat beberapa pernyataan print yang akan mencetak judul dan pesan untuk daftar layanan kurir dalam kota yang tersedia. Pesan-pesan ini menggunakan warna teks yang berbeda, seperti kuning dan biru. Setelah itu, *user* diminta untuk memasukkan pilihan layanan kurir dalam kota (1, 2, 3, atau 0) melalui pernyataan input. Setelah *user* memasukkan pilihannya, garis pemisah akan dicetak menggunakan pernyataan print. Kemudian, terdapat loop “`while True`” yang akan berjalan terus menerus sampai ada

perintah untuk keluar. Di dalam loop ini, terdapat kondisi if-elif-else yang akan mengevaluasi pilihan yang dimasukkan oleh *user*. Jika *user* memilih 1, fungsi “*layanan_kurir_dalam_kota()*” akan dipanggil. Di dalamnya, terdapat beberapa pernyataan print yang mencetak informasi mengenai layanan kurir reguler dalam kota yang tersedia. Setelah itu, terdapat blok “try-except” yang akan menangani beberapa jenis kesalahan yang mungkin terjadi. Jika *user* memilih 2, fungsi “*layanan_kurir_dalam_kota()*” akan dipanggil. Di dalamnya, terdapat beberapa pernyataan print yang mencetak informasi mengenai layanan kurir box dalam kota yang tersedia. Setelah itu, terdapat blok “try-except” yang akan menangani beberapa jenis kesalahan yang mungkin terjadi. Jika *user* memilih 3, fungsi “*layanan_kurir()*” akan dipanggil untuk kembali ke menu layanan kurir. Jika *user* memilih 0, program akan mengeluarkan pesan terima kasih dan kemudian keluar dari program menggunakan “*sys.exit()*”. Jika *user* memasukkan pilihan yang tidak valid, pesan kesalahan akan ditampilkan dan loop akan dihentikan menggunakan pernyataan “*break*”. Secara keseluruhan, syntax ini mengatur tampilan dan logika program untuk layanan kurir dalam kota dalam program rekomendasi harga ojek online. *User* dapat memilih jenis layanan kurir dalam kota yang tersedia, dan program akan memproses pilihan mereka sesuai dengan fungsi-fungsi yang relevan atau memberikan pesan kesalahan jika pilihan tidak valid.

3.3.5 Def Layanan Kurir Luar Kota

```
def layanan_kurir_luar_kota():
    print(Fore.RESET + "\n=====")
    print(Fore.YELLOW + "===== LAYANAN KURIR LUAR KOTA =====")
    print("\nBerikut Layanan Kurir Dalam Kota Yang Tersedia")
    print(Fore.CYAN + "1. Box (Mobil)")
    print(Fore.CYAN + "2. Kembali")
    print(Fore.RED + "0. Exit")
    input_luar_kota = input(Fore.GREEN + "\nPilih Layanan (1/2/0): ")
    print(Fore.RESET + "=====")

    while True:
        if input_luar_kota == "1":
            try:
                print(Fore.RESET + "\n=====")
                print(Fore.YELLOW + "===== LAYANAN KURIR BOX =====")
                print("\nPilih Sesuai Kebutuhan")
                input_ukuran, input_provinsi, input_asal, input_tujuan = mbl.get_input()
                harga_MaximBox = mbl.layananboxluarkota("maxim", input_ukuran.lower(), input_provinsi.title(), input_asal.title(), input_tujuan.title())
                harga_GoBox = mbl.layananboxluarkota("gojek", input_ukuran.lower(), input_provinsi.title(), input_asal.title(), input_tujuan.title())
                harga_GrabInstant = mbl.layananboxluarkota("Grab", input_ukuran.lower(), input_provinsi.title(), input_asal.title(), input_tujuan.title())
                print("\n=====")

                print("\n=====")
                print(Fore.BLUE + "Berikut Adalah Beberapa Perbandingan Harga \nLayanan Yang Dapat Kami Berikan\n")
                print("Harga Kurir Box Luar Kota dengan Maxim Box : Rp", harga_MaximBox)
                print("Harga Kurir Box Luar Kota dengan Go Box : Rp", harga_GoBox)
                print("Harga Kurir Box Luar Kota dengan Grab Instant : Rp", harga_GrabInstant)
                print()
            except:
```

```

if harga_MaximBox<harga_GoBox and harga_MaximBox<harga_GrabInstant:
    print(Fore.GREEN + "Rekomendasi Jasa Layanan Termurah Adalah \nMaxim Box")
elif harga_GrabInstant<harga_MaximBox and harga_GrabInstant<harga_GoBox:
    print(Fore.GREEN + "Rekomendasi Jasa Layanan Termurah Adalah \nGrab Instant")
elif harga_GoBox<harga_MaximBox and harga_GoBox<harga_GrabInstant:
    print(Fore.GREEN + "Rekomendasi Jasa Layanan Termurah Adalah \nGo Box")
print("=====")

layananlain = input(Fore.YELLOW + "\nApakah Anda Ingin Mengecek Layanan Lain? (Ya/Tidak) : ")
if layananlain.capitalize() == "Ya":
    layanan_kurir()
elif layananlain.capitalize() == "Tidak":
    print(Fore.GREEN + "Terima kasih telah menggunakan layanan kami. Sampai jumpa!")
    sys.exit()
else:
    print(Fore.RED + "Pilihan tidak valid. Silakan coba lagi.")
    break

except mbl.ukuranerror as e:
    print("Terjadi kesalahan : ", str(e))
except mbl.ProvinsiTidakTersediaError as e:
    print("Terjadi kesalahan : ", str(e))
except mbl.AsalTidakTersedia as e:
    print("Terjadi kesalahan : ", str(e))
except mbl.TujuanTidakTersedia as e:
    print("Terjadi kesalahan : ", str(e))
except Exception as e:
    print("Terjadi kesalahan : ", str(e))

elif input_luar_kota == "2":
    layanan_kurir()
elif input_luar_kota == "0":
    print(Fore.YELLOW + "Terima kasih telah menggunakan layanan kami. Sampai jumpa!")
    sys.exit()

```

Syntax di atas adalah sebuah fungsi bernama “layanan_kurir_luar_kota()” yang akan dieksekusi ketika dipanggil. Fungsi ini mengatur tampilan dan logika program untuk layanan kurir luar kota dalam program rekomendasi harga ojek online. Pertama, terdapat beberapa pernyataan print yang akan mencetak judul dan pesan untuk daftar layanan kurir luar kota yang tersedia. Pesan-pesan ini menggunakan warna teks yang berbeda, seperti kuning dan biru, yang diberikan oleh kelas “Fore” dari modul “colorama”. Setelah itu, *user* diminta untuk memasukkan pilihan layanan kurir luar kota (1, 2, atau 0) melalui pernyataan input. Setelah *user* memasukkan pilihannya, garis pemisah akan dicetak menggunakan pernyataan print. Kemudian, terdapat loop “while True” yang akan berjalan terus menerus sampai ada perintah untuk keluar. Di dalam loop ini, terdapat kondisi if-elif-else yang akan mengevaluasi pilihan yang dimasukkan oleh *user*. Jika *user* memilih 1, fungsi “layanan_kurir_luar_kota()” akan dipanggil. Di dalamnya, terdapat beberapa pernyataan print yang mencetak informasi mengenai layanan kurir box luar kota yang tersedia. Setelah itu, terdapat blok “try-except” yang akan menangani beberapa jenis kesalahan yang mungkin terjadi. Di dalam blok “try”, *user* diminta untuk memasukkan input seperti ukuran, provinsi, asal, dan tujuan

menggunakan fungsi “mbl.get_input()”. Setelah itu, harga-harga untuk layanan kurir box luar kota dari beberapa penyedia, seperti Maxim Box, Go Box, dan Grab Instant, dihitung menggunakan fungsi “mbl.layananboxluarkota()”. Kemudian, informasi perbandingan harga dan rekomendasi jasa layanan termurah akan dicetak menggunakan pernyataan print. Jika *user* ingin memeriksa layanan lain, fungsi “layanan_kurir()” akan dipanggil. Jika *user* tidak ingin memeriksa layanan lain, pesan terima kasih akan dicetak dan program akan keluar menggunakan “sys.exit()”. Jika *user* memasukkan pilihan yang tidak valid, pesan kesalahan akan ditampilkan dan loop akan dihentikan menggunakan pernyataan “break”. Jika *user* memilih 2, fungsi “layanan_kurir()” akan dipanggil untuk kembali ke menu layanan kurir. Jika *user* memilih 0, program akan mengeluarkan pesan terima kasih dan kemudian keluar dari program menggunakan “sys.exit()”. Secara keseluruhan, syntax ini mengatur tampilan dan logika program untuk layanan kurir luar kota dalam program rekomendasi harga ojek online. *User* dapat memilih jenis layanan kurir luar kota yang tersedia, dan program akan memproses pilihan mereka sesuai dengan fungsi-fungsi yang relevan atau memberikan pesan kesalahan jika pilihan tidak valid.

3.3.6 Def Layanan Transportasi

```
def layanan_transportasi():
    print(Fore.RESET + "\n=====")
    print(Fore.YELLOW + "== DAFTAR LAYANAN TRANSPORTASI OJEK ONLINE YANG TERSEDIA ==")
    print("\nBerikut Layanan Transportasi Ojek Online Yang Tersedia")
    print(Fore.CYAN + "1. Mobil")
    print(Fore.CYAN + "2. Motor")
    print(Fore.CYAN + "3. Kembali")
    print(Fore.RED + "0. Exit")
    input_transportasi = input(Fore.GREEN + "\nMasukkan Layanan (1/2/3/0) : ")
    print(Fore.RESET + "\n=====")

    while True:
        if input_transportasi == "1":
            transportasi_mobil()
        elif input_transportasi == "2":
            transportasi_motor()
        elif input_transportasi == "3":
            main()
        elif input_transportasi == "0":
            print(Fore.YELLOW + "Terima kasih telah menggunakan layanan kami. Sampai jumpa!")
            sys.exit()
```

Syntax di atas adalah sebuah fungsi bernama “layanan_transportasi()” yang terkait dengan syntax sebelumnya (“layanan_kurir_luar_kota()”). Fungsi ini bertanggung jawab untuk menampilkan menu dan mengatur logika program untuk layanan transportasi ojek online dalam program rekomendasi harga. Pertama, terdapat beberapa pernyataan print yang mencetak judul dan pesan untuk daftar layanan transportasi ojek online yang

tersedia. Pesan-pesan ini menggunakan warna teks yang berbeda, seperti kuning dan biru. Setelah itu, *user* diminta untuk memasukkan pilihan layanan transportasi (1, 2, 3, atau 0) melalui pernyataan input. Setelah *user* memasukkan pilihannya, garis pemisah akan dicetak menggunakan pernyataan print. Kemudian, terdapat loop “while True” yang akan berjalan terus menerus sampai ada perintah untuk keluar. Di dalam loop ini, terdapat kondisi if-elif-else yang akan mengevaluasi pilihan yang dimasukkan oleh *user*. Jika *user* memilih 1, fungsi “transportasi_mobil()” akan dipanggil. Fungsi ini kemungkinan merupakan implementasi yang lebih rinci untuk mengatur layanan transportasi menggunakan mobil. Jika *user* memilih 2, fungsi “transportasi_motor()” akan dipanggil. Fungsi ini kemungkinan merupakan implementasi yang lebih rinci untuk mengatur layanan transportasi menggunakan motor. Jika *user* memilih 3, fungsi “main()” akan dipanggil untuk kembali ke menu utama. Jika *user* memilih 0, program akan mengeluarkan pesan terima kasih dan kemudian keluar dari program menggunakan “sys.exit()”. Syntax ini mengatur tampilan dan logika program untuk layanan transportasi ojek online dalam program rekomendasi harga. *User* dapat memilih jenis layanan transportasi yang tersedia, dan program akan memproses pilihan mereka sesuai dengan fungsi-fungsi yang relevan atau memberikan pesan kesalahan jika pilihan tidak valid.

3.3.7 Def Layanan Transportasi Mobil

```
def transportasi_mobil():
    print(Fore.RESET + "\n=====")
    print(Fore.YELLOW + "===== LAYANAN TRANSPORTASI MOBIL =====")
    print("\nPilih Sesuai Kebutuhan")
    print(Fore.CYAN + "1. Dalam Kota")
    print(Fore.CYAN + "2. Luar Kota")
    print(Fore.CYAN + "3. Kembali")
    print(Fore.RED + "0. Exit")
    input_mobil = input(Fore.GREEN + "\nPilih Layanan (1/2/3/0) : ")
    print(Fore.RESET + "=====")

    while True:
        if input_mobil == "1":
            try:
                print(Fore.RESET + "\n=====")
                print(Fore.YELLOW + "===== LAYANAN MOBIL DALAM KOTA =====")
                print("\nPilih Sesuai Kebutuhan")
                input_provinsi, jarak = mcd.get_input()
                harga_MaximCar = mcd.layanan_mobil_dalam_kota("maxim", input_provinsi.title(), jarak)
                harga_GoCar = mcd.layanan_mobil_dalam_kota("gojek", input_provinsi.title(), jarak)
                harga_GrabCar = mcd.layanan_mobil_dalam_kota("Grab", input_provinsi.title(), jarak)
                print("\n=====")

                print(Fore.BLUE + "Berikut Adalah Beberapa Perbandingan Harga \nLayanan Yang Dapat Kami Berikan\n")
                print("Harga Mobil Dalam Kota dengan Maxim Car : Rp", harga_MaximCar)
                print("Harga Mobil Dalam Kota dengan Go Car : Rp", harga_GoCar)
                print("Harga Mobil Dalam Kota dengan Grab Car : Rp", harga_GrabCar)
                print()

                if harga_MaximCar < harga_GoCar and harga_MaximCar < harga_GrabCar:
                    print(Fore.GREEN + "Rekomendasi Jasa Layanan Termurah Adalah \nMaxim Car")
                elif harga_GrabCar < harga_MaximCar and harga_GrabCar < harga_GoCar:
                    print(Fore.GREEN + "Rekomendasi Jasa Layanan Termurah Adalah \nGrab Car")
                elif harga_GoCar < harga_MaximCar and harga_GoCar < harga_GrabCar:
                    print(Fore.GREEN + "Rekomendasi Jasa Layanan Termurah Adalah \nGo Car")
                print(Fore.GREEN + "Rekomendasi Jasa Layanan Termurah Adalah \nGo Car")
                print("=====")
            except:
                print(Fore.RED + "Input tidak valid")
                continue
```

```

        if harga_MaximCar < harga_GoCar and harga_MaximCar < harga_GrabCar:
            print(Fore.GREEN + "Rekomendasi Jasa Layanan Termurah Adalah \nMaxim Car")
        elif harga_GrabCar < harga_MaximCar and harga_GrabCar < harga_GoCar:
            print(Fore.GREEN + "Rekomendasi Jasa Layanan Termurah Adalah \nGrab Car")
        elif harga_GoCar < harga_MaximCar and harga_GoCar < harga_GrabCar:
            print(Fore.GREEN + "Rekomendasi Jasa Layanan Termurah Adalah \nGo Car")
        print("=====")

    layananlain = input(Fore.YELLOW + "\nApakah Anda Ingin Mengecek Layanan Lain? (Ya/Tidak) : ")
    if layananlain.capitalize() == "Ya":
        transportasi_mobil()
    elif layananlain.capitalize() == "Tidak":
        print(Fore.GREEN + "Terima kasih telah menggunakan layanan kami. Sampai jumpa!")
        sys.exit()
    else:
        print(Fore.RED + "Pilihan tidak valid. Silakan coba lagi.")
        break

except mcd.ProvinsiTidakTersediaError as e:
    print("Terjadi kesalahan : ", str(e))
except ValueError as e:
    print("Terjadi kesalahan : ", str(e))
except Exception as e:
    print("Terjadi Kesalahan : ", str(e))

```

```

elif input_mobil == "2":
    try:
        print(Fore.RESET + "\n=====")
        print(Fore.YELLOW + "===== LAYANAN MOBIL LUAR KOTA =====")
        print("\nPilih Sesuai Kebutuhan")
        input_provinsi = mcl.get_input1()
        if input_provinsi.title() == "Yogyakarta":
            print(Fore.RED + "Mohon Maaf Layanan Tidak Tersedia Di Provinsi Ini")
        else:
            input_asal, input_tujuan = mcl.get_input2()
            harga_MaximCar = mcl.layanan_mobil_luar_kota("maxim", input_provinsi.title(), input_asal.title(), input_tujuan.title())
            harga_GoCar = mcl.layanan_mobil_luar_kota("gojek", input_provinsi.title(), input_asal.title(), input_tujuan.title())
            harga_GrabCar = mcl.layanan_mobil_luar_kota("grab", input_provinsi.title(), input_asal.title(), input_tujuan.title())
            print("\n=====")

            print("\n=====")
            print(Fore.BLUE + "Berikut Adalah Beberapa Perbandingan Harga \nLayanan Yang Dapat Kami Berikan\n")
            print("Harga Mobil Luar Kota dengan Maxim Car : Rp", harga_MaximCar)
            print("Harga Mobil Luar Kota dengan Go Car : Rp", harga_GoCar)
            print("Harga Mobil Luar Kota dengan Grab Car : Rp", harga_GrabCar)
            print()
    
```

```

        if harga_MaximCar < harga_GoCar and harga_MaximCar < harga_GrabCar:
            print(Fore.GREEN + "Rekomendasi Jasa Layanan Termurah Adalah \nMaxim Car")
        elif harga_GrabCar < harga_MaximCar and harga_GrabCar < harga_GoCar:
            print(Fore.GREEN + "Rekomendasi Jasa Layanan Termurah Adalah \nGrab Car")
        elif harga_GoCar < harga_MaximCar and harga_GoCar < harga_GrabCar:
            print(Fore.GREEN + "Rekomendasi Jasa Layanan Termurah Adalah \nGo Car")
        print("=====")

    layananlain = input(Fore.YELLOW + "\nApakah Anda Ingin Mengecek Layanan Lain? (Ya/Tidak) : ")
    if layananlain.capitalize() == "Ya":
        transportasi_mobil()
    elif layananlain.capitalize() == "Tidak":
        print(Fore.GREEN + "Terima kasih telah menggunakan layanan kami. Sampai jumpa!")
        sys.exit()
    else:
        print(Fore.RED + "Pilihan tidak valid. Silakan coba lagi.")
        break

except mcl.ProvinsiTidakTersediaError as e:
    print("Terjadi kesalahan : ", str(e))
except mcl.AsalTidakTersedia as e:
    print("Terjadi kesalahan : ", str(e))
except mcl.TujuanTidakTersedia as e:
    print("Terjadi kesalahan : ", str(e))
except Exception as e:
    print("Terjadi kesalahan : ", str(e))

elif input_mobil == "3":
    layanan_transportasi()
elif input_mobil == "0":
    print(Fore.YELLOW + "Terima kasih telah menggunakan layanan kami. Sampai jumpa!")
    sys.exit()

```

Syntax tersebut merupakan implementasi dari fungsi “transportasi_mobil()” yang merupakan bagian dari suatu program atau skrip yang berkaitan dengan layanan transportasi mobil. Deskripsi dari syntax tersebut adalah sebagai berikut: Fungsi

“transportasi_mobil()” digunakan untuk menampilkan menu dan mengatur logika layanan transportasi mobil. Ketika fungsi ini dipanggil, pertama-tama akan ditampilkan tampilan menu dengan menggunakan berbagai warna teks seperti “Fore.RESET”, “Fore.YELLOW”, “Fore.CYAN”, dan “Fore.RED”. *User* akan diminta untuk memilih layanan dengan memasukkan input melalui “input_mobil”. Selanjutnya, terdapat perulangan “while True” yang menjalankan blok kode di dalamnya selama kondisi tersebut benar. Jika *user* memilih layanan “1” untuk dalam kota, maka akan ditampilkan menu dan *user* diminta untuk memasukkan provinsi dan jarak tempuh melalui “mcd.get_input()”. Selanjutnya, harga layanan dari tiga penyedia (Maxim Car, Go Car, dan Grab Car) akan dihitung menggunakan fungsi “mcd.layanan_mobil_dalam_kota()” dengan menggunakan nilai-nilai yang telah dimasukkan *user*. Informasi harga-harga layanan tersebut akan dicetak dengan menggunakan tampilan teks dan warna yang berbeda. Selanjutnya, terdapat percabangan “if-elif-else” untuk menentukan rekomendasi jasa layanan termurah berdasarkan harga-harga yang telah dihitung sebelumnya. Setelah itu, *user* akan ditanyakan apakah ingin memeriksa layanan lain atau tidak. Jika *user* memilih untuk memeriksa layanan lain, maka fungsi “transportasi_mobil()” akan dipanggil kembali secara rekursif. Jika *user* memilih untuk tidak memeriksa layanan lain, maka program akan berakhir dengan pesan terima kasih. Terdapat juga penanganan beberapa jenis kesalahan yang mungkin terjadi selama eksekusi fungsi, seperti “mcd.ProvinsiTidakTersediaError”, “ValueError”, dan “Exception”. Selain itu, terdapat juga cabang “elif” untuk layanan “2” yang berkaitan dengan layanan mobil luar kota. Logika yang dijalankan dalam cabang tersebut mirip dengan layanan dalam kota, namun dengan implementasi yang berbeda untuk menghitung harga layanan mobil luar kota dan menampilkan informasi perbandingan harga. Terdapat juga penanganan beberapa jenis kesalahan yang mungkin terjadi selama eksekusi fungsi, seperti “mcl.ProvinsiTidakTersediaError”, “mcl.AsalTidakTersedia”, dan “mcl.TujuanTidakTersedia”. Terdapat juga cabang “elif” untuk layanan “3” yang akan memanggil fungsi “layanan_transportasi()” yang tidak terdefinisi dalam kode yang diberikan. Terakhir, terdapat cabang `elif` untuk pilihan “0” yang akan mengakhiri program dengan pesan terima kasih. Secara keseluruhan, fungsi “transportasi_mobil()” bertanggung jawab untuk menampilkan menu, mengumpulkan input *user*, menghitung harga layanan, menampilkan informasi harga, memberikan rekomendasi layanan termurah, dan mengatur alur program berdasarkan pilihan *user*.

3.3.8 Def Layanan Transportasi Motor

```
def transportasi_motor():
    print(Fore.RESET + "\n=====")
    print(Fore.YELLOW + "===== LAYANAN TRANSPORTASI MOTOR =====")
    print("\nPilih Sesuai Kebutuhan")
    print(Fore.CYAN + "1. Dalam Kota")
    print(Fore.CYAN + "2. Kembali")
    print(Fore.RED + "0. Exit")
    input_motor = input(Fore.GREEN + "\nPilih Layanan (1/2/0) : ")
    print(Fore.RESET + "\n=====")

    while True:
        if input_motor == "1":
            try:
                print(Fore.RESET + "\n=====")
                print(Fore.YELLOW + "===== LAYANAN MOTOR DALAM KOTA =====")
                print("\nPilih Sesuai Kebutuhan")
                input_provinsi, jarak = mr.get_input()
                harga_MaximBike = mr.layanan_motor("maxim", input_provinsi.title(), jarak)
                harga_GoRide = mr.layanan_motor("gojek", input_provinsi.title(), jarak)
                harga_GrabBike = mr.layanan_motor("Grab", input_provinsi.title(), jarak)
                print("\n=====")

                print(Fore.BLUE + "Berikut Adalah Beberapa Perbandingan Harga \nLayanan Yang Dapat Kami Berikan\n")
                print("Harga Motor Dalam Kota dengan Maxim Bike : Rp", harga_MaximBike)
                print("Harga Motor Dalam Kota dengan Go Ride : Rp", harga_GoRide)
                print("Harga Motor Dalam Kota dengan Grab Bike : Rp", harga_GrabBike)
                print()

                if harga_MaximBike < harga_GoRide and harga_MaximBike < harga_GrabBike:
                    print(Fore.GREEN + "Rekomendasi Jasa Layanan Termurah Adalah \nMaxim Bike")
                elif harga_GrabBike < harga_MaximBike and harga_GrabBike < harga_GoRide:
                    print(Fore.GREEN + "Rekomendasi Jasa Layanan Termurah Adalah \nGrab Bike")
                elif harga_GoRide < harga_MaximBike and harga_GoRide < harga_GrabBike:
                    print(Fore.GREEN + "Rekomendasi Jasa Layanan Termurah Adalah \nGo Ride")
                print("=====")

                layananlain = input(Fore.YELLOW + "\nApakah Anda Ingin Mengecek Layanan Lain? (Ya/Tidak) : ")
                if layananlain.capitalize() == "Ya":
                    layanan_transportasi()
                elif layananlain.capitalize() == "Tidak":
                    print(Fore.GREEN + "Terima kasih telah menggunakan layanan kami. Sampai jumpa!")
                    sys.exit()
                else:
                    print(Fore.RED + "Pilihan tidak valid. Silakan coba lagi.")
                    break

            except mr.ProvinsiTidakTersediaError as e:
                print("Terjadi kesalahan : ", str(e))
            except ValueError as e:
                print("Terjadi kesalahan : ", str(e))
            except Exception as e:
                print("Terjadi kesalahan : ", str(e))

        elif input_motor == "2":
            layanan_transportasi()
        elif input_motor == "0":
            print(Fore.YELLOW + "Terima kasih telah menggunakan layanan kami. Sampai jumpa!")
            sys.exit()

if __name__ == "__main__":
    main()
```

Syntax tersebut terdiri dari dua bagian. Pertama adalah fungsi “transportasi_motor()”, dan kedua adalah blok kode yang mengeksekusi fungsi “main()” jika script dijalankan sebagai program utama. Fungsi “transportasi_motor()” digunakan untuk menampilkan menu dan mengatur logika layanan transportasi motor. Pertama-tama, tampilan menu akan ditampilkan dengan menggunakan berbagai warna teks seperti “Fore.RESET”, “Fore.YELLOW”, “Fore.CYAN”, dan “Fore.RED”. *User* akan diminta untuk memilih layanan dengan memasukkan input melalui “input_motor”. Setelah itu, terdapat perulangan “while True” yang menjalankan blok kode di dalamnya selama kondisi tersebut benar. Jika *user* memilih layanan “1” untuk dalam kota, maka akan ditampilkan menu dan *user* diminta untuk memasukkan provinsi dan jarak tempuh

melalui `“mr.get_input()”`. Selanjutnya, harga layanan dari tiga penyedia (Maxim Bike, Go Ride, dan Grab Bike) akan dihitung menggunakan fungsi `“mr.layanan_motor()”` dengan menggunakan nilai-nilai yang telah dimasukkan *user*. Informasi harga-harga layanan tersebut akan dicetak dengan menggunakan tampilan teks dan warna yang berbeda. Selanjutnya, terdapat percabangan `“if-elif-else”` untuk menentukan rekomendasi jasa layanan termurah berdasarkan harga-harga yang telah dihitung sebelumnya. Setelah itu, *user* akan ditanyakan apakah ingin memeriksa layanan lain atau tidak. Jika *user* memilih untuk memeriksa layanan lain, maka fungsi `“layanan_transportasi()”` akan dipanggil kembali. Jika *user* memilih untuk tidak memeriksa layanan lain, maka program akan berakhir dengan pesan terima kasih. Terdapat juga penanganan beberapa jenis kesalahan yang mungkin terjadi selama eksekusi fungsi, seperti `“mr.ProvinsiTidakTersediaError”`, `“ValueError”`, dan `“Exception”`. Selanjutnya, terdapat cabang `“elif”` untuk pilihan "2" yang akan memanggil fungsi `“layanan_transportasi()”`. Terakhir, terdapat cabang `“elif”` untuk pilihan "0" yang akan mengakhiri program dengan pesan terima kasih. Blok kode terakhir menggunakan `“if __name__ == “__main__”:` untuk mengeksekusi fungsi `“main()”` saat script dijalankan sebagai program utama. Namun, tidak ada definisi atau penjelasan mengenai fungsi `“main()”` dalam kode yang diberikan, sehingga perlu diperhatikan terkait *user*an script secara keseluruhan.

BAB IV

HASIL DAN PEMBAHASAN

5.1 Hasil Running Program Python

```
===== PROGRAM REKOMENDASI HARGA OJEK ONLINE =====  
Selamat datang di layanan transportasi dan kurir ojek online!  
Program Ini Akan Memberikan Anda Rekomendasi  
Layanan Jasa Ojek Online Di Pulau Jawa  
  
Silakan pilih jenis layanan yang Anda butuhkan:  
1. Layanan Transportasi  
2. Layanan Kurir  
0. Keluar  
  
Masukkan pilihan Anda (1/2/0): 1  
=====
```

Saat menjalankan kode utama, akan muncul pesan dan dua layanan yang dapat dipilih oleh *user*, yaitu layanan transportasi dan layanan kurir. Pada kali ini akan dipilih layanan transportasi terlebih dahulu, oleh karena itu perlu memasukkan 1 untuk memilih layanan transportasi.

```
===== DAFTAR LAYANAN TRANSPORTASI OJEK ONLINE YANG TERSEDIA =====  
Berikut Layanan Transportasi Ojek Online Yang Tersedia  
1. Mobil  
2. Motor  
3. Kembali  
0. Exit  
  
Masukkan Layanan (1/2/3/0) : 1  
=====
```

Setelah itu, diberikan pilihan untuk memilih transportasi menggunakan motor atau mobil, selain itu ada pilihan untuk kembali ke layer seleksi sebelumnya atau mengakhiri layanan. Karena ingin melakukan transportasi menggunakan mobil, maka perlu dimasukkan 1 untuk memilih transportasi mobil.

```
===== LAYANAN TRANSPORTASI MOBIL =====  
  
Pilih Sesuai Kebutuhan  
1. Dalam Kota  
2. Luar Kota  
3. Kembali  
0. Exit  
  
Pilih Layanan (1/2/3/0) : 2  
=====
```

Karena ingin menuju ke luar kota, diinput angka 2.

```
===== LAYANAN MOBIL LUAR KOTA =====  
  
Pilih Sesuai Kebutuhan  
Input Provinsi Asal : Jawa Tengah  
Kota Asal : Surakarta  
Kota Tujuan : Magelang  
=====
```

Selanjutnya *user* perlu menginput provinsi asal *user* dengan mengetiknya, agar dapat berjalan dengan benar, kota asal yang dimasukkan haruslah merupakan sebuah kota yang berada di provinsi asal yang sebelumnya dimasukkan *user*. Terakhir, masukkan kota yang ingin dituju, kota harus berada di dalam provinsi yang sama

```
===== LAYANAN MOBIL LUAR KOTA =====  
  
Pilih Sesuai Kebutuhan  
Input Provinsi Asal : Jawa tengah  
Kota Asal : surakarta  
Kota Tujuan : surabaya  
Terjadi kesalahan : Tujuan tidak tersedia  
=====
```

Bila memasukkan kota tujuan yang berada diluar provinsi pilihan, maka akan ditampilkan apa kesalahan yang terjadi, dalam kasus ini pesan yang keluar adalah “Tujuan tidak tersedia”.

```

=====
Berikut Adalah Beberapa Perbandingan Harga
Layanan Yang Dapat Kami Berikan

Harga Mobil Luar Kota dengan Maxim Car : Rp 488600
Harga Mobil Luar Kota dengan Go Car    : Rp 344800
Harga Mobil Luar Kota dengan Grab Car   : Rp 275500

Rekomendasi Jasa Layanan Termurah Adalah
Grab Car
=====

Apakah Anda Ingin Mengecek Layanan Lain? (Ya/Tidak) : Ya

```

Bila *user* telah memasukkan informasi secara benar maka akan keluar tiga harga layanan transportasi dari *provider* jasa yang berbeda. Selain menampilkan harga, BestDriver juga menampilkan rekomendasi harga layanan jasa yang termurah. Setelah mendapatkan harga yang dibutuhkan, *user* dapat memilih apakah ingin mengecek layanan lain atau tidak. Jika memilih “Ya”, maka *user* akan dibawa kembali ke halaman sebelumnya.

```

=====
== DAFTAR LAYANAN TRANSPORTASI OJEK ONLINE YANG TERSEDIA ==

Berikut Layanan Transportasi Ojek Online Yang Tersedia
1. Mobil
2. Motor
3. Kembali
0. Exit

Masukkan Layanan (1/2/3/0) : 2
=====

```

Karena ingin melihat harga layanan saat menggunakan motor, masukkan angka 2.

```

=====
===== LAYANAN TRANSPORTASI MOTOR =====

Pilih Sesuai Kebutuhan
1. Dalam Kota
2. Kembali
0. Exit

Pilih Layanan (1/2/0) : 1
=====

```

Karena motor hanya memberi layanan dalam kota, input angka 1.


```

=====
===== LAYANAN MOTOR DALAM KOTA =====
Pilih Sesuai Kebutuhan
Input Provinsi Asal : jawa barat
Jarak Tempuh (km): bogor
Terjadi kesalahan : Input jarak harus berupa angka

```

Saat memasukan informasi jarak tempuh, input harus berupa angka, bila memasukkan selain angka, maka akan keluar pesan bahwa *user* telah melakukan kesalahan bertuliskan “Input jarak harus berupa angka”.

```

=====
===== LAYANAN MOTOR DALAM KOTA =====
Pilih Sesuai Kebutuhan
Input Provinsi Asal : cimahi
Jarak Tempuh (km): 20
Terjadi kesalahan : Provinsi yang anda input tidak terdapat di Pulau Jawa

```

Sedangkan, bila tidak memasukkan provinsi dengan benar atau terjadi kesalahan pengejaan, maka akan keluar pesan *exception* bahwa provinsi yang diinput tidak terdapat di Pulau Jawa.

```

=====
===== LAYANAN MOTOR DALAM KOTA =====
Pilih Sesuai Kebutuhan
Input Provinsi Asal : jawa barat
Jarak Tempuh (km): 20

=====

Berikut Adalah Beberapa Perbandingan Harga
Layanan Yang Dapat Kami Berikan

Harga Motor Dalam Kota dengan Maxim Bike : Rp 46200.0
Harga Motor Dalam Kota dengan Go Ride      : Rp 46000.0
Harga Motor Dalam Kota dengan Grab Bike    : Rp 71500.0

Rekomendasi Jasa Layanan Termurah Adalah
Go Ride

=====

Apakah Anda Ingin Mengecek Layanan Lain? (Ya/Tidak) : 

```

Agar harga yang keluar sesuai dengan kebutuhan *user*, perlu dimasukkan provinsi dari kota asal *user*. Bila *user* berasal dari Kota Bandung, maka provinsi yang dimasukkan haruslah Jawa Barat. Dalam memasukkan provinsi, *user* tidak wajib menggunakan kapital agar input dapat terbaca.

```
=====
Apakah Anda Ingin Mengecek Layanan Lain? (Ya/Tidak) : Ya
=====
== DAFTAR LAYANAN TRANSPORTASI OJEK ONLINE YANG TERSEDIA ==

Berikut Layanan Transportasi Ojek Online Yang Tersedia
1. Mobil
2. Motor
3. Kembali
0. Exit

Masukkan Layanan (1/2/3/0) : 3
=====
===== PROGRAM REKOMENDASI HARGA OJEK ONLINE =====
Selamat datang di layanan transportasi dan kurir ojek online!
Program Ini Akan Memberikan Anda Rekomendasi
Layanan Jasa Ojek Online Di Pulau Jawa

Silakan pilih jenis layanan yang Anda butuhkan:
1. Layanan Transportasi
2. Layanan Kurir
0. Keluar
```

Untuk menggunakan layanan kurir, *user* dapat kembali ke layer sebelumnya dan memilih “Layanan Kurir” dengan memasukkan angka 2.

```
=====
===== DAFTAR LAYANAN KURIR OJEK ONLINE YANG TERSEDIA =====

Berikut Layanan Kurir Ojek Online Yang Tersedia
1. Dalam Kota
2. Luar Kota
3. Kembali
0. Exit

Masukkan Layanan (1/2/3/0): 2
=====
```

Untuk kurir keluar kota masukkan 2.

```
=====
===== LAYANAN KURIR LUAR KOTA =====
Berikut Layanan Kurir Dalam Kota Yang Tersedia
1. Box (Mobil)
2. Kembali
0. Exit

Pilih Layanan (1/2/0): 1
=====
```

Layanan mobil box dapat diakses dengan memasukkan 1.

```
=====
===== LAYANAN KURIR BOX =====

Pilih Sesuai Kebutuhan
| Ukuran | Dimensi | Berat Maks. |
|-----|-----|-----|
| small | 200 x 130 x 120 cm | berat max. 1000kg |
|-----|-----|-----|
| medium | 200 x 130 x 130 cm | berat max. 2000kg |
|-----|-----|-----|
| large | 300 x 160 x 130 cm | berat max. 2000kg |
|-----|-----|-----|

Ukuran Box : medium
Input provinsi Asal : jawa barat
Kota Asal : bandung
Kota Tujuan : tangerang

=====

Berikut Adalah Beberapa Perbandingan Harga
Layanan Yang Dapat Kami Berikan

Harga Kurir Box Luar Kota dengan Maxim Box : Rp 735000
Harga Kurir Box Luar Kota dengan Go Box : Rp 2030000
Harga Kurir Box Luar Kota dengan Grab Instant : Rp 597000

Rekomendasi Jasa Layanan Termurah Adalah
Grab Instant

=====

Apakah Anda Ingin Mengecek Layanan Lain? (Ya/Tidak) : █
```

Input ukuran box, provinsi asal, kota asal, dan kota tujuan sesuai kebutuhan.

```
Apakah Anda Ingin Mengecek Layanan Lain? (Ya/Tidak) : Tidak
Terima kasih telah menggunakan layanan kami. Sampai jumpa!
```

Bila sudah selesai menggunakan layanan BestDrive, masukkan “Tidak”, maka kode utama akan selesai dijalankan.