**RESPONSI 2**

**PRAKTIKUM STRUKTUR DATA DAN ALGORITMA**

**2024 / 2025**

|  |  |  |
| --- | --- | --- |
| **IDENTITAS** | | |
| Nama | : | Syaikhasril Maulana Firdaus |
| NIM | : | L0124077 |
| Kelas | : | D |
| Judul Program | : | GOJEK |
| Deskripsi Program | : | Program ini merupakan sebuah simulasi aplikasi Gojek yang berbasis konsol, dirancang untuk mendemonstrasikan penerapan berbagai struktur data dan algoritma fundamental. Aplikasi ini mencakup implementasi layanan-layanan utama seperti GoRide yang menggunakan algoritma Dijkstra untuk pencarian rute terpendek, GoCar dengan aplikasi Minimum Spanning Tree, GoFood yang memanfaatkan Queue dan Sorting, GoMart dengan Binary Search Tree dan Stack, serta layanan lain yang masing-masing mengilustrasikan konsep-konsep spesifik dalam struktur data secara praktis. |

*Tabel penggunaan materi:*

|  |  |  |
| --- | --- | --- |
| **NO** | **MATERI** | **PERAN** |
| 1. | List | Dalam program Gojek ini, struktur data List memegang peranan kunci dalam fitur GoFood. Secara spesifik, List digunakan untuk mengelola dan menampilkan daftar restoran yang tersedia serta menu makanan dari setiap restoran. Penggunaan ArrayList, salah satu implementasi dari List, memungkinkan penyimpanan data restoran dan menu secara dinamis. Artinya, kita bisa dengan mudah menambah atau mengurangi jumlah restoran dan item menu tanpa harus mengubah struktur kode secara signifikan.  Pada GoFoodService, List<Resto> digunakan untuk menyimpan semua objek restoran yang tersedia untuk dipilih oleh pengguna. Saat pengguna memilih sebuah restoran, program akan mengambil menu makanan restoran tersebut, yang juga disimpan dalam sebuah List<Makanan>, dan menampilkannya kepada pengguna. Selain itu, daftar pesanan pengguna juga dikelola menggunakan List<Pesanan>, yang memungkinkan program untuk menyimpan beberapa item makanan yang dipilih oleh pengguna sebelum proses checkout. Fleksibilitas List ini sangat mendukung fungsionalitas inti dari layanan GoFood, yaitu memesan makanan dari berbagai pilihan restoran. |
| 2. | Stack | Pada fitur GoMart, struktur data Stack digunakan untuk mengelola keranjang belanja pengguna. Implementasi Stack<Pesanan> berfungsi sebagai tumpukan di mana setiap item yang dipilih oleh pengguna akan "didorong" (di-push) ke dalam tumpukan. Keunikan dari Stack adalah sifatnya yang LIFO (Last-In, First-Out), yang berarti item terakhir yang dimasukkan ke dalam keranjang akan menjadi item pertama yang dikeluarkan.  Peran ini menjadi jelas saat proses checkout. Program akan mengeluarkan (melakukan pop) setiap pesanan dari tumpukan satu per satu hingga tumpukan kosong. Hal ini mensimulasikan bagaimana seseorang akan mengeluarkan barang-barang dari keranjang belanja fisik, di mana barang yang paling atas (terakhir dimasukkan) akan diambil terlebih dahulu. Penggunaan Stack di sini tidak hanya memenuhi kebutuhan fungsional untuk menyimpan pesanan sementara, tetapi juga memberikan demonstrasi klasik tentang bagaimana struktur data LIFO dapat diterapkan dalam skenario dunia nyata seperti manajemen keranjang belanja. |
| 3. | Queue | Dalam fitur GoFood, struktur data Queue (Antrean) diimplementasikan menggunakan LinkedList untuk menyimulasikan antrean pesanan di restoran. Setiap objek Resto memiliki Queue<Integer> sendiri yang merepresentasikan urutan pelanggan yang sedang menunggu pesanan mereka disiapkan. Penggunaan Queue di sini sangat penting karena mencerminkan sistem antrean di dunia nyata yang bersifat FIFO (First-In, First-Out). Artinya, pesanan yang masuk lebih dulu akan diproses lebih dulu, menciptakan sistem yang adil bagi semua pelanggan.  Ketika seorang pengguna berhasil melakukan pemesanan melalui GoFood, nomor antrean baru akan ditambahkan ke akhir Queue menggunakan metode add(). Program kemudian memberi tahu pengguna nomor antrean mereka. Meskipun program ini tidak menyimulasikan proses penyelesaian pesanan (mengeluarkan dari antrean), penggunaan Queue secara efektif mendemonstrasikan bagaimana sebuah layanan digital dapat mengelola dan memberikan informasi urutan layanan kepada penggunanya secara transparan dan terstruktur. |
| 4. | Set | Dalam program Gojek ini, Set digunakan untuk melakukan validasi input dari pengguna secara efisien. Secara spesifik, sebuah HashSet bernama validMenuOptions diinisialisasi dengan semua pilihan menu yang valid (angka 0 hingga 8). Set memiliki keunggulan utama yaitu kemampuannya untuk memeriksa keberadaan suatu elemen dengan sangat cepat (rata-rata dalam waktu konstan) menggunakan metode contains().  Peran ini sangat penting dalam metode run() di kelas GojekApp. Sebelum menjalankan switch-case untuk layanan yang dipilih, program terlebih dahulu memeriksa apakah input pilihanMenu dari pengguna ada di dalam validMenuOptions. Jika tidak ada, program akan langsung memberitahu pengguna bahwa input tidak valid dan meminta untuk mencoba lagi, tanpa harus menjalankan logika program yang lebih kompleks. Penggunaan Set di sini memastikan bahwa hanya input yang sesuai yang akan diproses, sehingga membuat program lebih robust dan efisien dalam menangani interaksi pengguna. |
| 5. | Map | Struktur data Map, yang diimplementasikan sebagai HashMap, memegang dua peran penting dalam aplikasi ini. Peran utamanya adalah sebagai penyimpanan riwayat transaksi pengguna. Dalam kelas GojekApp, Map<Integer, Riwayat> riwayatPesanan digunakan untuk menyimpan setiap transaksi yang berhasil dilakukan. Integer berfungsi sebagai key unik untuk setiap transaksi (ID transaksi), dan Riwayat adalah value yang berisi objek dengan detail pesanan seperti jenis layanan dan total biaya. Penggunaan Map memungkinkan program untuk mengambil data transaksi secara efisien hanya dengan menggunakan ID-nya, yang sangat berguna saat menampilkan riwayat kepada pengguna.  Peran kedua Map terdapat pada fitur GoSend, di mana Map<String, String> parent digunakan untuk mendukung implementasi struktur data Disjoint Set. Di sini, Map berfungsi untuk memetakan setiap lokasi (sebagai key) ke "induk" atau perwakilannya dalam sebuah himpunan (sebagai value). Ini adalah cara yang efisien untuk melacak keterhubungan antar lokasi dalam simulasi jaringan pengiriman. Dengan demikian, Map terbukti sangat fleksibel untuk mengelola data berbasis pasangan kunci-nilai, baik untuk keperluan level aplikasi seperti riwayat transaksi maupun untuk keperluan struktural yang lebih mendasar seperti pada algoritma Disjoint Set. |
| 6. | Tree | Dalam program ini, struktur data Tree (Pohon) digunakan secara spesifik pada fitur GoRide untuk merepresentasikan dan menampilkan lokasi penjemputan dalam format hierarkis. Kelas LokasiTree dirancang sebagai general tree, di mana setiap node (lokasi) dapat memiliki banyak children (sub-lokasi). Hal ini sangat cocok untuk menggambarkan hubungan antar wilayah, misalnya sebuah kecamatan sebagai parent dan kelurahan-kelurahan di dalamnya sebagai children.  Peran utamanya adalah untuk meningkatkan pengalaman pengguna. Daripada menampilkan daftar lokasi yang panjang dan tidak terstruktur, LokasiTree memungkinkan program untuk menyajikan pilihan lokasi secara berjenjang melalui metode displayPreOrder. Pengguna dapat dengan mudah memahami hubungan antar lokasi (misalnya, "Puncang Sawit" adalah bagian dari area "Jebres"). Implementasi ini menunjukkan bagaimana Tree dapat digunakan untuk mengorganisir data yang secara alami memiliki struktur hierarkis, sehingga membuat antarmuka pengguna menjadi lebih intuitif dan teratur. |
| 7. | Graph | Graph adalah struktur data fundamental yang digunakan untuk memodelkan jaringan lokasi dan hubungan antar lokasi tersebut dalam aplikasi ini. Secara spesifik, graph direpresentasikan menggunakan adjacency matrix (matriks ketetanggaan), di mana setiap baris dan kolom mewakili sebuah lokasi, dan nilai di dalam matriks menunjukkan jarak (atau "bobot") antar dua lokasi. Jika nilainya 0, artinya tidak ada jalur langsung di antara keduanya. Peran Graph sangat krusial pada dua fitur utama:  GoRide: Graph digunakan sebagai dasar untuk menjalankan algoritma Dijkstra. Algoritma ini menjelajahi graph untuk menemukan jalur terpendek dari lokasi penjemputan ke semua lokasi tujuan lainnya. Dengan demikian, graph memungkinkan program untuk mengkalkulasi jarak tempuh yang paling efisien dan menentukan biaya perjalanan secara akurat.  GoCar: Di sini, graph menjadi input untuk algoritma Prim yang bertujuan mencari Minimum Spanning Tree (MST). MST digunakan untuk mengidentifikasi sekumpulan rute paling efisien yang dapat menghubungkan semua titik lokasi utama dengan total jarak minimum. Ini menyimulasikan bagaimana layanan transportasi dapat merancang rute yang paling hemat biaya untuk operasionalnya.  Secara keseluruhan, Graph berfungsi sebagai "peta digital" yang menjadi landasan bagi fungsionalitas inti terkait navigasi dan kalkulasi rute di dalam aplikasi. |
| 8. | Disjoint Set | Dalam fitur GoSend, struktur data Disjoint Set digunakan untuk menyimulasikan jaringan logistik pengiriman paket. Setiap lokasi pengiriman pada awalnya dianggap sebagai himpunan (set) yang terpisah. Struktur data ini kemudian digunakan untuk mengelompokkan beberapa lokasi ke dalam "hub" pengiriman regional (misalnya, "HUB-UTARA" dan "HUB-SELATAN"), dan selanjutnya menghubungkan semua hub ke "PUSAT-SORTIR".  Perannya adalah sebagai berikut:  union: Operasi ini digunakan untuk menggabungkan lokasi-lokasi ke dalam satu jaringan. Misalnya, union("Jebres", "HUB-UTARA") menandakan bahwa semua paket dari Jebres akan melewati HUB-UTARA. Ini secara efisien membangun model konektivitas jaringan.  find: Operasi ini digunakan untuk menentukan jalur atau kelompok dari suatu lokasi. Ketika program ingin menunjukkan rute pengiriman, ia menggunakan operasi find secara rekursif untuk melacak jalur dari lokasi asal paket hingga ke pusat sortir.  Secara efektif, Disjoint Set memungkinkan program untuk mengelola dan memvisualisasikan bagaimana paket dari berbagai area yang berbeda dikonsolidasikan dan dialihkan melalui titik-titik pusat. Ini adalah contoh bagus tentang bagaimana struktur data ini dapat memodelkan masalah pengelompokan dan konektivitas dalam sistem jaringan. |
| 9. | Sorting | Fungsi Sorting (Pengurutan) diimplementasikan pada layanan GoFood untuk memberikan kemudahan bagi pengguna dalam melihat menu makanan. Secara spesifik, program ini menggunakan algoritma Bubble Sort untuk mengurutkan daftar makanan (List<Makanan>) berdasarkan harganya, dari yang termurah hingga yang termahal.  Peran utamanya adalah untuk meningkatkan pengalaman pengguna (UX). Ketika pengguna dihadapkan dengan banyak pilihan menu, fitur untuk mengurutkan menu berdasarkan harga sangatlah membantu. Pengguna dapat dengan cepat menemukan makanan yang sesuai dengan anggaran mereka tanpa harus memeriksa setiap item satu per satu. Ketika pengguna mengetik "sortir", metode bubbleSort akan dipanggil untuk menata ulang urutan menu yang ditampilkan. Ini adalah contoh penerapan praktis algoritma sorting untuk memberikan fungsionalitas yang bermanfaat dalam sebuah aplikasi. |
| 10. | Binary Tree | Dalam fitur GoLife, struktur data Binary Tree digunakan sebagai pohon keputusan (decision tree) untuk memandu pengguna dalam memilih layanan. Setiap node dalam pohon merepresentasikan sebuah pertanyaan atau pilihan dengan dua kemungkinan jawaban (misalnya, "Pilih Tipe Layanan: Kebersihan (1) atau Pijat (2)?"). Pilihan pengguna akan menentukan navigasi ke node anak kiri atau kanan.  Peran utamanya adalah untuk menyederhanakan proses pemilihan layanan yang memiliki beberapa tahap. Dengan mengikuti alur pertanyaan dari root (akar) hingga ke leaf (daun), pengguna secara bertahap menyaring pilihan hingga tiba pada satu layanan spesifik yang diinginkan (contoh: "GoMassage: Pijat 1 Jam (Rp 100.000)"). Struktur Binary Tree sangat cocok untuk skenario ini karena secara alami memodelkan serangkaian pilihan biner (ya/tidak, kiri/kanan, opsi 1/opsi 2). Ini membuat alur pemilihan layanan menjadi logis, terstruktur, dan mudah diikuti oleh pengguna. |
| 11. | Binary Search Tree | Binary Search Tree (BST) digunakan dalam layanan GoMart untuk mengelola data minimarket secara efisien. Dalam implementasi ini, setiap node pada BST menyimpan informasi tentang sebuah minimarket, dan jarak minimarket tersebut dari pengguna dijadikan sebagai key untuk penyisipan dan pengurutan. Sifat utama BST, di mana semua node di sub-pohon kiri memiliki kunci yang lebih kecil dan semua node di sub-pohon kanan memiliki kunci yang lebih besar, dimanfaatkan sepenuhnya di sini.  Peran kuncinya adalah untuk menyediakan daftar minimarket yang otomatis terurut berdasarkan jarak. Ketika GoMartService perlu menampilkan daftar minimarket, ia memanggil metode getNodesInOrder(). Metode ini melakukan in-order traversal pada BST, yang secara inheren akan mengembalikan data dalam urutan menaik (dari jarak terpendek ke terjauh). Hal ini membuat program dapat menyajikan informasi yang paling relevan bagi pengguna (minimarket terdekat) dengan sangat efisien, tanpa perlu melakukan operasi sorting terpisah setiap kali daftar akan ditampilkan. |
| 12. | MST | Dalam fitur GoCar, algoritma MST (Minimum Spanning Tree) digunakan untuk menyimulasikan perencanaan rute operasional yang paling efisien. Dengan menggunakan Algoritma Prim, program menganalisis graph (peta koneksi antar lokasi) untuk menemukan sebuah "pohon" yang menghubungkan semua titik lokasi utama dengan total jarak (bobot) seminimal mungkin.  Peran utamanya adalah untuk demonstrasi optimisasi jaringan. Sebelum pengguna memesan, program akan menghitung dan menampilkan rute-rute efisien yang menjadi "tulang punggung" (backbone) jaringan layanan GoCar. Ini menunjukkan kepada pengguna bagaimana layanan ini, secara hipotetis, dapat menghubungkan semua area layanannya dengan biaya infrastruktur (dalam hal ini, total jarak) yang paling rendah. Jadi, MST di sini tidak digunakan untuk menghitung rute pesanan spesifik pengguna, melainkan untuk merancang fondasi jaringan rute yang paling optimal secara keseluruhan. |
| 13. | Shortest Path | Algoritma pencarian Shortest Path (Jalur Terpendek), yang diimplementasikan dengan Algoritma Dijkstra, adalah komponen paling krusial dalam layanan GoRide. Peran utamanya adalah untuk menghitung jarak tempuh terpendek dari satu lokasi penjemputan yang dipilih pengguna ke semua kemungkinan lokasi tujuan lainnya dalam jaringan Graph yang telah didefinisikan.  Ketika pengguna memilih titik jemput, metode dijkstra akan dieksekusi. Metode ini mengambil Graph (peta) dan titik awal sebagai input, lalu secara sistematis menjelajahi semua kemungkinan jalur untuk menemukan rute dengan total bobot (jarak) paling minimum. Hasilnya adalah sebuah array yang berisi jarak terpendek dari titik jemput ke semua titik lainnya. Informasi ini kemudian digunakan secara langsung untuk dua hal: menampilkan pilihan tujuan beserta estimasi jaraknya, dan yang terpenting, mengkalkulasi total biaya perjalanan. Tanpa algoritma ini, layanan GoRide tidak akan bisa memberikan informasi jarak dan harga yang akurat kepada pengguna. |

Hasil Praktikum :

*(jelaskan bagaimana program berjalan beserta gambar screenshot jalannya program)*