Analisis Tugas Robotika Week 14

- **Analisis Learn Rust with Me**
  - **Hello World!**

    Rust adalah bahasa pemrograman yang unggul dalam keamanan memori, performa tinggi, dan pengelolaan paralelisme tanpa risiko **data race**. Instalasi Rust dapat dilakukan menggunakan **rustup**, sebuah toolchain manager yang memudahkan pengelolaan berbagai versi Rust dan alat-alat terkait.

    Setelah instalasi, kode Rust dapat dijalankan dengan mengompilasi file .rs menggunakan perintah rustc, lalu menjalankan file biner yang dihasilkan. Namun, untuk proyek yang lebih kompleks, penggunaan **Cargo** sebagai package manager dan build system bawaan Rust jauh lebih praktis. Dengan Cargo, pengguna dapat dengan mudah membuat proyek baru, mengatur dependensi, dan menjalankan kode melalui perintah sederhana seperti cargo new, cargo build, atau cargo run. Cargo tidak hanya menyederhanakan proses kompilasi tetapi juga secara otomatis mengelola paket dan versinya, menjadikannya pilihan utama untuk pengembangan aplikasi Rust.
  - **Data Type:**

    Dalam Rust, program ini memperkenalkan penggunaan struktur klasik dan tuple. Struktur Student memiliki tiga field: name, level, dan remote, sementara tuple Grades menyimpan berbagai tipe data. Program juga mendemonstrasikan variabel immutable dan mutable, serta konsep variable shadowing. Rust mendukung berbagai tipe data seperti f32, bool, char, dan String, serta cara mengakses elemen dalam struktur klasik dan tuple.

  - **If-else Statement:**

    Program ini menggunakan pernyataan if/else untuk mengevaluasi kondisi, seperti memeriksa kesamaan dua angka dan menentukan apakah perlu membawa jaket berdasarkan cuaca. Selain itu, beberapa kondisi diperiksa untuk memastikan angka berada dalam rentang yang diinginkan, dengan menggunakan beberapa pernyataan if/else berturut-turut untuk mengatur nilai variabel.

  - **Arrays and Vecto**

    Program ini membandingkan penggunaan arrays dan vectors di Rust. Arrays memiliki ukuran tetap, sedangkan vectors bersifat dinamis dan dapat berubah ukurannya. Vectors lebih fleksibel karena elemen bisa ditambah atau dihapus menggunakan metode seperti push() dan pop(). Program menunjukkan cara mengakses dan memodifikasi elemen dalam arrays dan vectors.

  - **Loops:**

    Rust menyediakan beberapa jenis loop: loop tak terbatas (loop), while loop, dan for loop. Program ini menunjukkan penggunaan loop untuk menjalankan kode berulang kali dengan kondisi tertentu, seperti menggunakan while untuk mencetak pesan sampai mencapai batas, dan for untuk iterasi melalui koleksi data.

  - **HashMap:**

    Program ini menggunakan HashMap untuk menyimpan data dalam pasangan kunci-nilai. Dengan HashMap::new(), elemen dapat ditambahkan dengan metode insert() dan diakses dengan get(). Program juga menunjukkan cara menghapus entri menggunakan remove(), serta pentingnya efisiensi dalam penyimpanan dan pengambilan data menggunakan struktur ini.

- **Analisis Kode Robotika**

  Dalam rangka merancang berbagai program robotik menggunakan Rust, masing-masing program ini menunjukkan konsep-konsep penting dalam pemrograman robotik, seperti perencanaan jalur, pengendalian gerakan, penghindaran rintangan, penjadwalan tugas, sistem berbasis event-driven, dan penggunaan model probabilistik untuk navigasi.

  - Perencanaan Jalur Sederhana menggunakan algoritma A* untuk merencanakan jalur robot dari titik awal ke tujuan, dengan mempertimbangkan rintangan dalam grid 2D. Program ini mengimplementasikan perhitungan biaya perjalanan dan jarak ke tujuan, menggunakan heuristik Manhattan dan BinaryHeap untuk efisiensi. Rust sebagai bahasa pemrograman memungkinkan pengelolaan memori yang baik, penting untuk algoritma pencarian jalur yang membutuhkan efisiensi tinggi.

  - Gerakan Robot dengan Input Pengguna memberikan pengalaman interaktif bagi pengguna untuk mengendalikan robot secara langsung melalui input terminal. Program ini memungkinkan pengguna menggerakkan robot di grid berdasarkan input tombol arah (w, s, a, d) dan memastikan robot tidak keluar dari batasan yang ditentukan.

  - Simulasi Robot Menghindari Rintangan menggunakan kembali algoritma A* untuk mensimulasikan robot yang harus menghindari rintangan dalam peta 2D. Dalam program ini, robot mencari jalur terpendek dari posisi start ke tujuan sambil menghindari area yang terblokir. Hasil dari algoritma A* ini adalah jalur yang menghindari rintangan yang ada dalam grid, dengan memanfaatkan BinaryHeap dan HashMap untuk memproses node.

  - Penjadwalan Robot dengan Prioritas menggunakan struktur BinaryHeap untuk mengelola antrean tugas robot berdasarkan prioritas. Setiap tugas memiliki tingkat prioritas yang menentukan urutan pemrosesan. Dengan cara ini, robot dapat menyelesaikan tugas-tugas penting lebih dulu, sementara tugas dengan prioritas rendah akan diproses setelahnya. Program ini sangat berguna untuk aplikasi robotik yang memerlukan pengelolaan tugas yang efisien.

  - Robotik dengan Sistem Event-Driven menggambarkan bagaimana robot dapat merespons perubahan dalam lingkungan secara real-time, seperti deteksi rintangan baru atau perubahan tujuan. Program ini menggunakan multi-threading dan komunikasi antar-thread dengan mpsc::channel() untuk menangani event yang dikirimkan oleh lingkungan dan diproses oleh robot. Pendekatan ini berguna untuk pengembangan sistem robotik yang responsif terhadap perubahan dinamis di sekitarnya.

  - Robot dengan Model Probabilistik mengatasi ketidakpastian dalam navigasi robot dengan mempertimbangkan faktor ketidakpastian dalam pergerakan robot. Setiap langkah robot memiliki sedikit variasi dalam jarak, yang mencerminkan dunia nyata yang sering kali tidak dapat diprediksi secara pasti. Program ini menggunakan BinaryHeap untuk memilih jalur terbaik berdasarkan biaya, yang juga memperhitungkan ketidakpastian yang terjadi saat robot bergerak.

  Secara keseluruhan, program-program ini menawarkan pendekatan yang beragam dalam pengembangan robotika, mulai dari pencarian jalur hingga penjadwalan tugas dan responsif terhadap perubahan lingkungan. Pemrograman dengan Rust mendukung implementasi yang efisien dan aman, yang sangat penting dalam pengembangan sistem robotik yang nyata.