Real world Application of Queue Data structure using Minecraft

MINI PROJECT REPORT

By

Aman Varma (RA2211026010172)
Manoj P (RA2211026010174)
Aadhithya M (RA2211026010180)
Hariharan G Mudaliar (RA2211026010183)

Under the guidance of

Dr. Prithi S

In partial fulfilment for the Course

of

21CSC201J – Data Structures and Algorithms

in the Department of Computational Intelligence

FACULTY OF ENGINEERING AND TECHNOLOGY

SCHOOL OF COMPUTING

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR

NOVEMBER 2023

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that this minor project report for the course **21CSC201J – Data Structures and Algorithms** entitled in "**Real world Application of Queue Data structure using Minecraft**" is the bonafide work of Aman Varma (RA2211026010172), Manoj.P (RA2211026010174) and Aadhithya.M (RA2211026010180), Hariharan G Mudaliar (RA2211026010183) who carried out the work under my supervision.

SIGNATURE

Dr. Prithi S

Data Structures and Algorithms – Course Faculty

Assistant Professor

Department of Computational Intelligence

SRM Institute of Science and Technology

Kattankulathur

SIGNATURE

Dr. Annie Uthra R

Head of the Department

Professor

Department of Computational Intelligence

SRM Institute of Science and Technology

Kattankulathur

Objective :

The Rail Track Delivery System in Minecraft is an innovative application of the queue data structure, providing a practical solution for efficient item transportation within the game world. This project aims to enhance the automation and organization of item delivery systems, offering a reliable and orderly means of moving items from one location to another.

The primary objective of this project is to create a system that operates based on the First-In-First-Out (FIFO) principle, where items are processed in the same order they are received. Leveraging the inherent properties of hoppers, minecarts, and redstone logic, the Rail Track Delivery System ensures that items collected from chests are delivered to their destination chest in a structured and predictable manner.

Through a detailed technical implementation and strategic use of Minecraft's mechanics, this system enables players to streamline their resource management and transportation processes, making it particularly useful for large-scale mining operations, farms, or any scenario where item transfer efficiency is essential.

The project report provides an in-depth exploration of the system's architecture, the queue data structure's application in Minecraft, and the technical intricacies involved in its design. It also discusses challenges encountered during development and the system's performance results.

Abstract :

The Rail Track Delivery System in Minecraft is a creative and practical application of the queue data structure, designed to enhance item transportation and management within the Minecraft gaming environment. This project addresses the challenge of creating a system that ensures items are processed in a First-In-First-Out (FIFO) order, closely resembling the operation of real-world queue systems.

The main objective of this project is to develop a system that leverages Minecraft's game mechanics, including hoppers, minecarts, and Redstone components, to guarantee the orderly movement of items from their source chest to a destination chest. This innovative system streamlines resource management and simplifies item delivery, making it highly valuable for various in-game scenarios.

This project report provides a comprehensive overview of the Rail Track Delivery System, including its architecture, functionality, and the technical implementation of the queue data structure within the Minecraft environment. It also discusses the challenges faced during the project's development and presents performance results, highlighting the effectiveness of the system.

The Rail Track Delivery System not only demonstrates the versatility of Minecraft as an educational and problem-solving platform but also offers a practical and creative solution for players seeking an organized and efficient approach to item transportation within the game.

Introduction :

Minecraft, a world of endless creativity and ingenuity, provides an ideal environment for crafting solutions to complex problems using a virtual, block-based universe. In this realm, players often face the challenge of efficiently transporting items from one location to another within the game, a task that mirrors real-world logistics and automation challenges. The Rail Track Delivery System in Minecraft is a creative and practical response to this challenge, combining elements of computer science and game mechanics to offer an innovative solution.

At its core, this project revolves around the concept of a queue data structure, a fundamental concept in computer science used for orderly data processing. By applying this concept in the context of Minecraft, we aim to create a system that mirrors the real-world behavior of a queue, ensuring that items are processed in a First-In-First-Out (FIFO) order. The Rail Track Delivery System represents a marriage of gaming and computer science, demonstrating how knowledge and skills can be applied to solve real-world problems, even in virtual landscapes.

The primary objective of this project is to design and implement a system that collects items from a source chest, organizes them in a queue, and delivers them to a destination chest while maintaining their order. To achieve this, we leverage the inherent properties of Minecraft's game mechanics, such as hoppers, minecarts, and Redstone components, to build an intricate yet intuitive system that guarantees the orderly movement of items. The result is a system that not only enhances the gameplay experience but also simplifies and streamlines resource management, making it especially valuable for various in-game scenarios.

In this project report, we will delve into the details of the Rail Track Delivery System, covering its architecture, functionalities, and the technical implementation of the queue data structure within the Minecraft environment. We will also discuss the challenges encountered during development and present the performance results, showcasing the system's efficiency in handling item transportation.

The Rail Track Delivery System represents the ingenuity and adaptability of players within the Minecraft community, highlighting the platform's potential as a tool for learning and problem-solving. This project offers a unique blend of creativity, technical skill, and real-world application, demonstrating how a virtual world can serve as a testing ground for innovative solutions.

Through this report, we aim to provide insights into the development process, share the outcomes achieved, and convey the valuable lessons learned through the implementation of this innovative Minecraft automation system.

Hardware requirements:

Computer:

Operating System: Windows 10, macOS, or Linux.
Processor: Dual-core CPU, such as an Intel Core i3 or AMD equivalent.
Memory (RAM): 8 GB or more.

Graphics Card:

Integrated graphics are generally sufficient for Minecraft, but a dedicated graphics card can provide a smoother experience, especially when using resource-intensive shaders or high-resolution texture packs.

Storage:

Storage Space: A few gigabytes of free storage space for the Minecraft installation and world saves.
Solid-State Drive (SSD): While not mandatory, using an SSD can significantly reduce loading times.

Internet Connection:

A stable internet connection for multiplayer gameplay and updates.

Peripherals:

Mouse: A mouse with a scroll wheel for easy item selection and building.
Keyboard: A standard keyboard for controls.
Monitor: A display with a resolution of at least 1920x1080 for a better visual experience.

Redstone Knowledge:

Understanding of Redstone mechanics and automation in Minecraft. This is more of a skill requirement, but it's essential for setting up complex systems like the Rail Track Delivery System.

Minecraft Version:

Ensure you are running a supported version of Minecraft that is compatible with the Redstone mechanics and features needed for your project.

Optional: Dedicated Server

If you plan to run your Rail Track Delivery System on a multiplayer server, you may need a more powerful server with better CPU and memory resources, depending on the number of concurrent players and the complexity of the world.

Concepts/Working Principle:

Source Chest: This is the initial chest where items are stored before they are transported. Items are collected from this chest and processed by the system.

Hoppers: Hoppers are used to transport items between containers. In this system, hoppers are employed to move items from the source chest to the minecart.

Redstone Logic: This block represents the intricate Redstone logic that controls the entire system. It includes various Redstone components like Redstone torches, repeaters, and comparators that ensure the correct timing and actions.

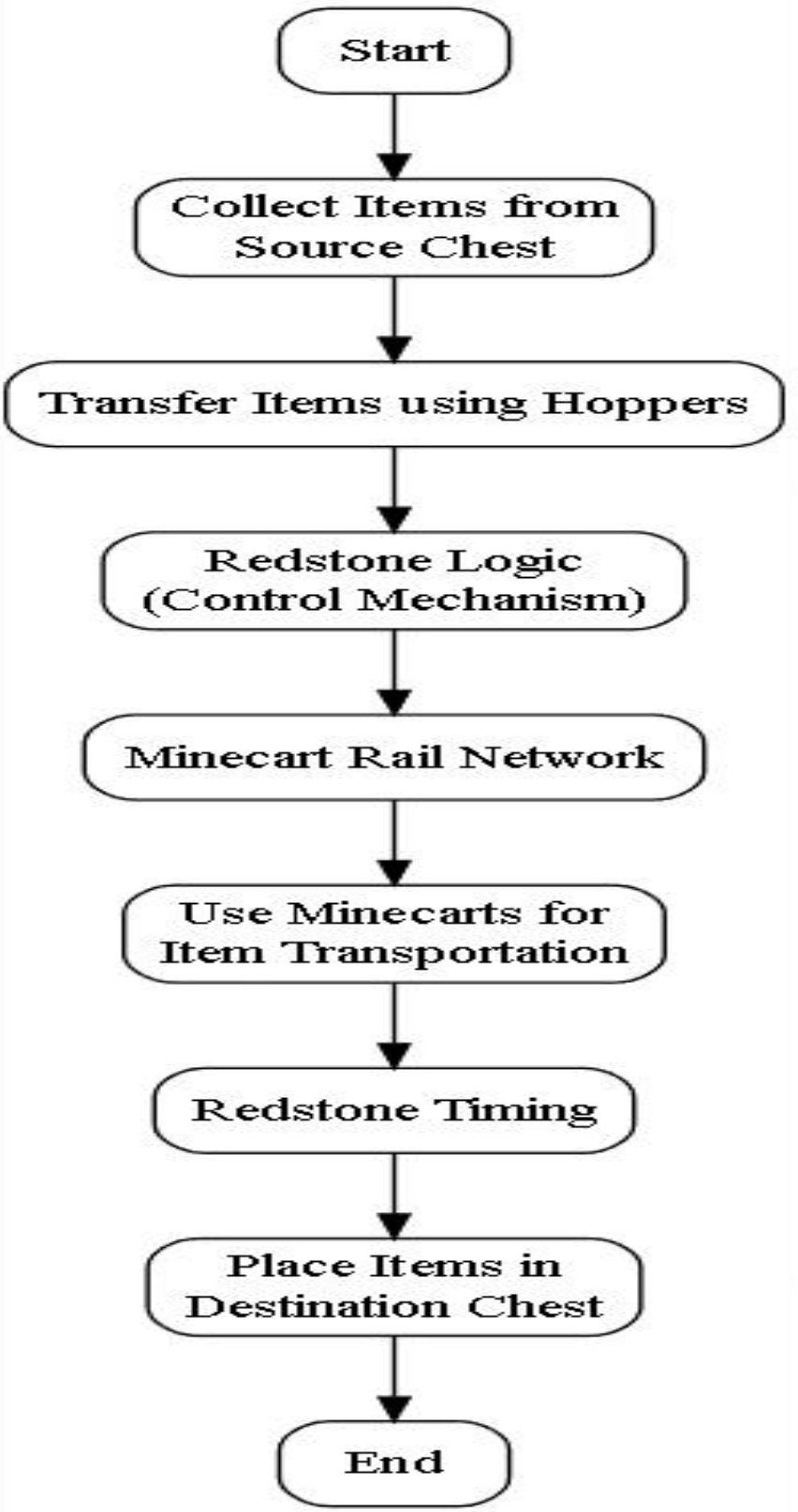
Minecart Rail: The minecart rail network facilitates the movement of minecarts carrying items. It includes powered rails and regular rails to control the speed and direction of the minecarts.

Minecarts: Minecarts are used to transport items from the source chest to the destination chest. Items are loaded into minecarts and sent along the rail network.

Redstone Timing: This block represents the timing mechanisms used to control the movement of minecarts and the release of items. It ensures that items are delivered in the correct order.

Destination Chest: The destination chest is where items are ultimately placed after being transported by the minecarts. It's the final point in the delivery process

Flowchart :


Start
Collect Items from Source Chest
Transfer Items using Hoppers
Redstone Logic (Control Mechanism)
Minecart Rail Network
Use Minecarts for Item Transportation
Redstone Timing
Place Items in Destination Chest
End

Working / Output :

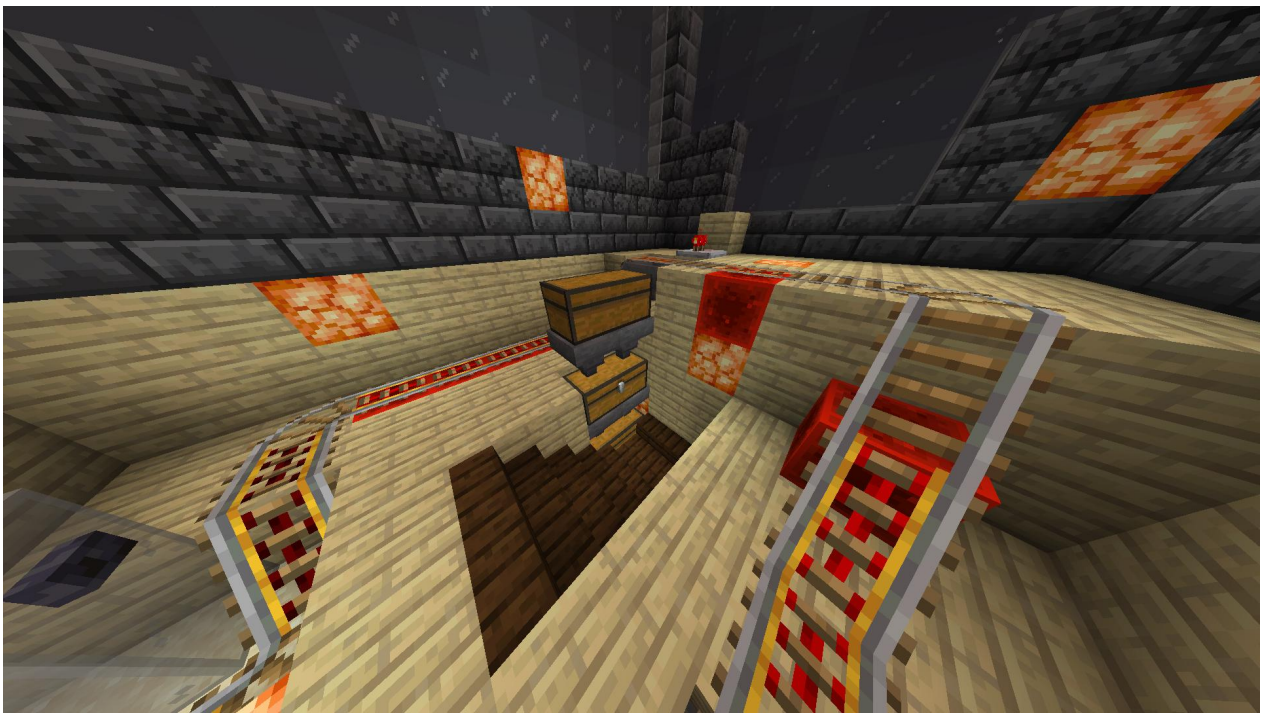
Entire Overiew :

Start Side :

Starting Chest (with item inside) :

Cart going to deposit :

Deposit Area :

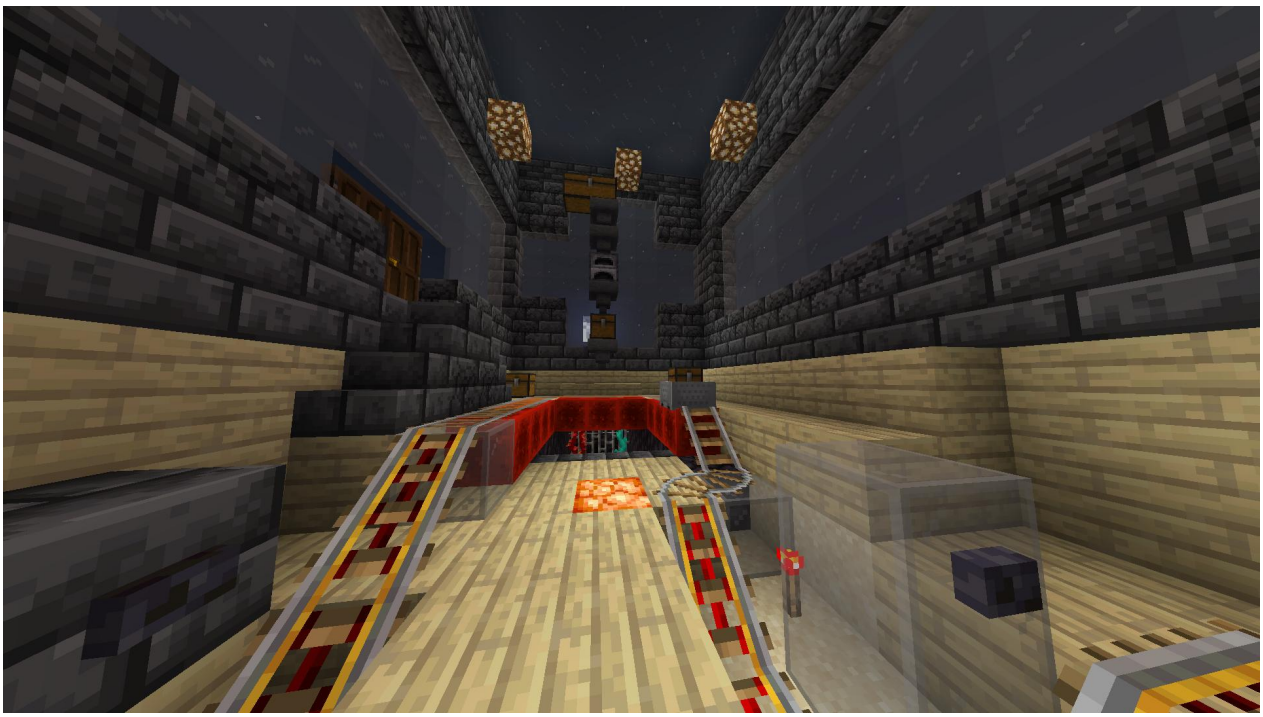


Deposit Side Chest (with item delivered) :

Redstone Logic :

Cart Stop (after delivering) :

Cart going to collect (new item) :



Collecting Side :

Conclusion :

Queue Data Structure Analogy:

In DSA, a queue is a linear data structure where elements are stored in a linear order. The first element inserted into the queue is considered the front, and the last element is the rear. Items are enqueued (added) at the rear and dequeued (removed) from the front, following the FIFO principle.

Source Chest as an Enqueue Point:

In your Rail Track Delivery System, the "Source Chest" serves as an analogy to the front of a queue. Items are collected and enqueued into the system via this chest. Just like in a queue, the first item to be collected (enqueued) corresponds to the front of the queue.

Hoppers as the Queue's Structure:

The use of hoppers in your system represents the structure that preserves the order of items, similar to how a queue preserves the order of elements. As items are collected from the source chest and transferred through hoppers, they maintain their order within the system.

Redstone Logic Control as Queue Operations:

The Redstone logic and components are responsible for controlling the operations of your Rail Track Delivery System. These components simulate the queue operations of enqueue and dequeue in a traditional data structure. When items are "stopped" and processed, this corresponds to dequeuing items from the front of the queue.

Minecart as the Processing Element:

The minecart in your system represents the processing element that dequeues items from the "front" (analogous to the source chest). The minecart ensures that the items are removed and processed in the same order in which they were enqueued.

Hopper and Deposit Process as Rear of the Queue:

The hopper and deposit process correspond to the rear of the queue. Once items are dequeued by the minecart and processed, they are deposited into the destination chest. This step is akin to enqueuing items at the rear of a queue.

.

References :

Minecraft Wiki (Redstone Circuitry):

The Minecraft Wiki's Redstone Circuitry section provides comprehensive information about Redstone mechanics, components, and circuits.

Minecraft Community Forums:

Websites like Minecraft Forum and Planet Minecraft have active Redstone and automation subforums where you can find discussions, ideas, and tutorials.

Minecraft Books:

There are several books available that delve into Redstone and automation in Minecraft, such as "The Redstone Handbook" by Scholastic.

Data Structures and Algorithms Textbooks:

Textbooks like "Introduction to Algorithms" by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein provide a comprehensive introduction to data structures, including FIFO-based structures like queues.