VIETNAM NATIONAL UNIVERSITY - HO CHI MINH CITY
HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

GAME PROGRAMMING - HK241

ASSIGNMENT
RED-BLACK TOWER DEFENSE

Lecturer: Phan Tran Minh Khue

Students: Nguyen Ha Thanh Hau 2052985
Mac Ho Do Khang 2252297

HO CHI MINH CITY, NOV 2024

Contents

1 Introduction

The RedBlack Defensegame is an innovative educational tool designed to teach the fundamentals of the Red-Black Tree data structure through an engaging and interactive experience. Set in a medieval kingdom under siege, players must defend their castle from waves of siege vehicles such as battering rams, catapults, and trebuchets. The game uniquely integrates the mechanics of Red-Black Trees into its tower defense gameplay, requiring players to strategically place towers (nodes) while ensuring the tree remains balanced and adheres to its rules.

By combining the excitement of a strategy game with the rigor of data structure concepts, the game aims to provide students and learners with an immersive way to master operations like insertion, deletion, and rotations in Red-Black Trees. Whether it is connecting new towers or rebalancing after enemy destruction, players will actively apply Red-Black Tree principles to progress through the game. This innovative approach makes the RedBlack Defensegame both a compelling challenge for gamers and a powerful learning resource for computer science students.

2 Game Concept

2.1 Overview

RedBlack Defenseis a 3D tower defense game that uniquely combines strategic gameplay with educational content on the Red-Black Tree data structure. The player's objective is to defend their castle against waves of siege vehicles by strategically placing auto-shooting towers along the enemies' path. However, unlike traditional tower defense games, the towers in RedBlack Defenseare connected through a special structure based on the Red-Black Tree, a self-balancing binary search tree.

In this game, players must build and maintain a balanced Red-Black Tree by carefully placing and upgrading towers (nodes). Properly adhering to the rules of Red-Black Trees maximizes the towers' efficiency, enabling them to fire with greater speed and power.

As players navigate through relaxing moments to intense encounters with heavy enemy waves, they engage with the core principles of Red-Black Trees, including node insertion, balancing, rotations, and color properties. By integrating these data structure concepts into the core mechanics of the game, RedBlack Defenseprovides an enjoyable yet educational experience, helping players gain a deep understanding of the Red-Black Tree's workings and its practical manipulation.

2.2 Theory

2.2.1 Serious Game

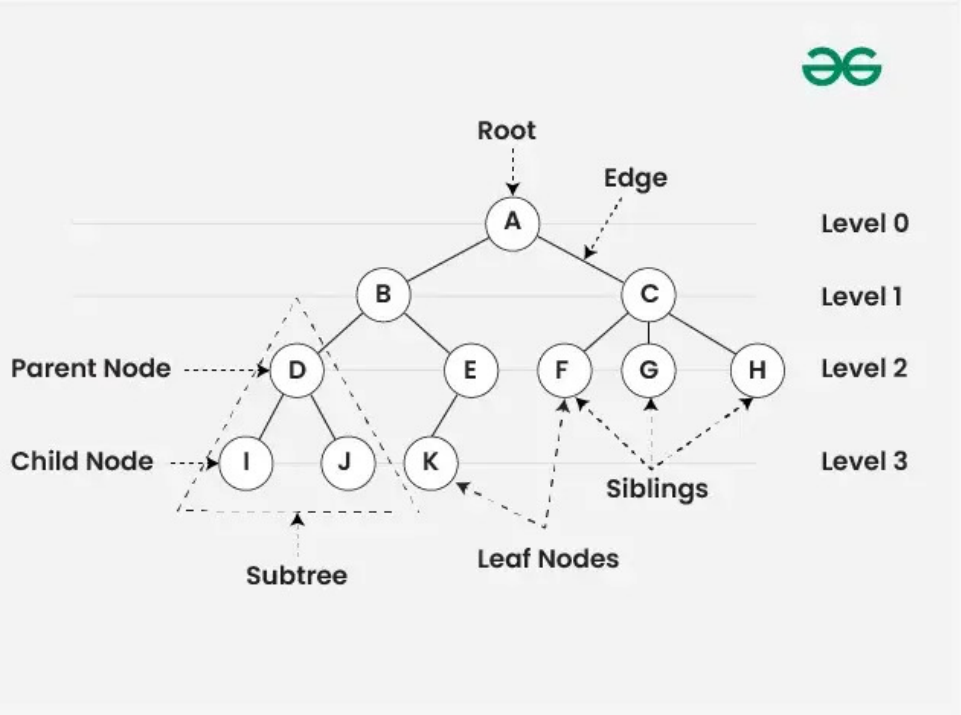
A serious game is a type of game designed not just for entertainment but with a specific educational or training purpose in mind. Unlike traditional games, serious games focus on delivering knowledge, developing skills, or influencing behavior through interactive and engaging gameplay. By incorporating elements of game design such as challenges, feedback, and rewards, serious games aim to motivate players to learn and practice real-world concepts in a fun and immersive environment. These games are used in various fields, including education, healthcare, corporate training, and social awareness, to create an impactful and memorable learning experience.

2.2.2 Data Structure

The Red-Black Tree is a crucial self-balancing binary search tree that ensures consistent performance during insertion, deletion, and lookup operations. To fully appreciate the structure and functionality of a Red-Black Tree, it is important to understand its foundational concepts: trees, binary trees, and binary search trees.

- **Tree**

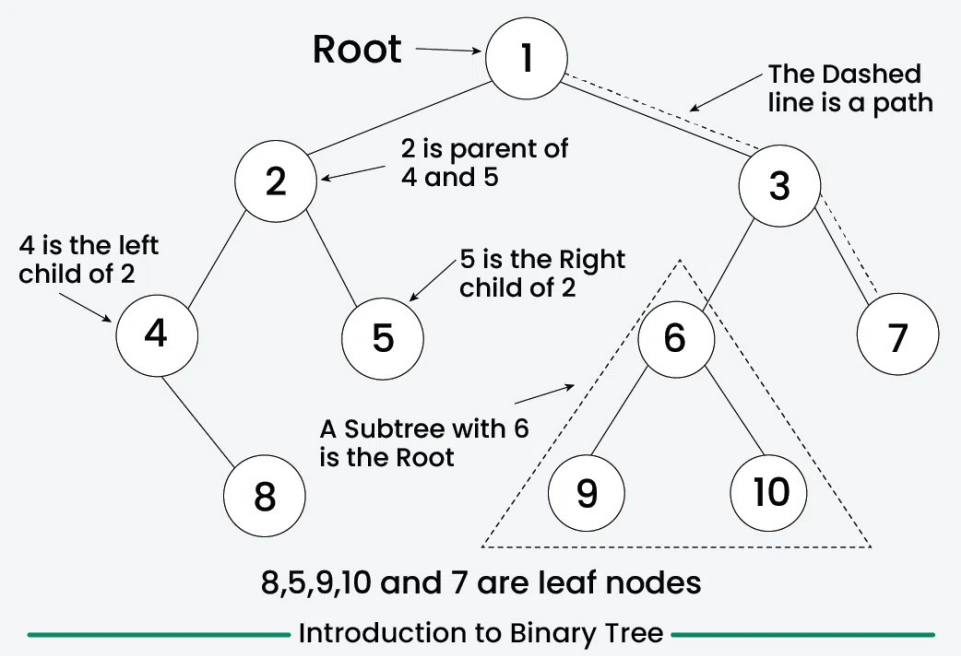
  A **tree** is a hierarchical data structure consisting of nodes, where each node contains a value and links (edges) to its child nodes. The topmost node is called the root, and nodes with no children are called leaves. Trees are widely used in computing for representing relationships, hierarchical data, and for efficient searching and sorting operations.



Source: Geeksforgeeks

- **Binary Tree**

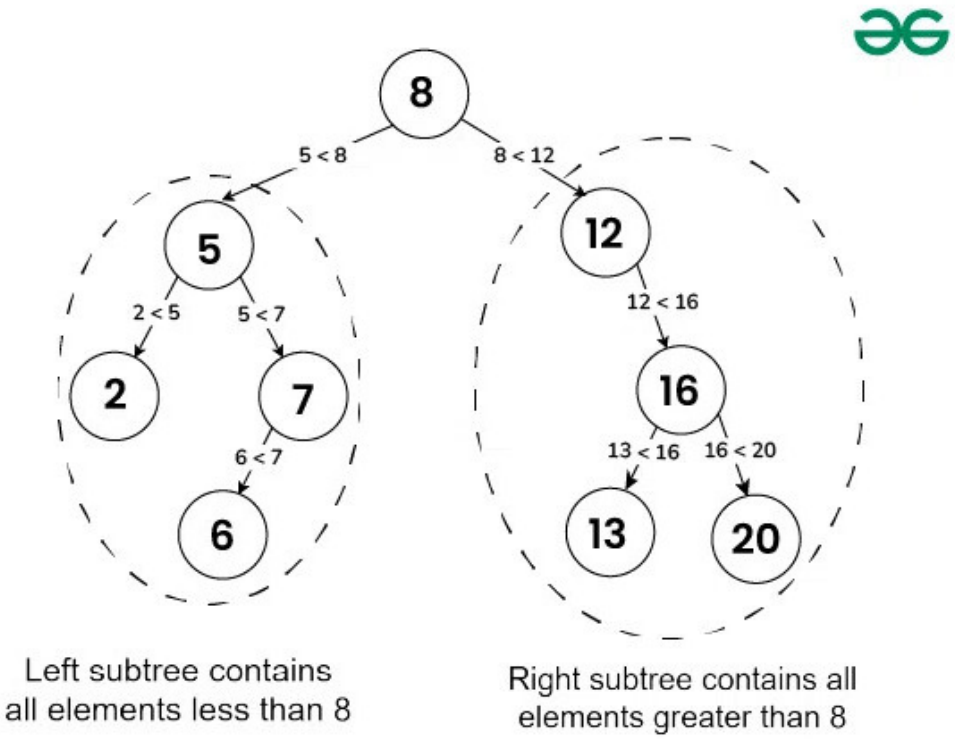
  A **binary tree** is a specialized form of a tree where each node has at most two children, referred to as the left child and the right child. Binary trees form the basis for various tree-based data structures due to their simplicity and adaptability.



Source: Geeksforgeeks

- **Binary Search Tree (BST)**

  A **binary search tree (BST)** extends the concept of a binary tree by enforcing an ordering property: for every node, the values of all nodes in its left subtree are less than its own value, and the values of all nodes in its right sub-tree are greater. This ordering makes search, insertion, and deletion operations efficient, with an average time complexity of O(logn) in balanced trees. However, traditional binary search trees may become unbalanced, leading to reduced efficiency.

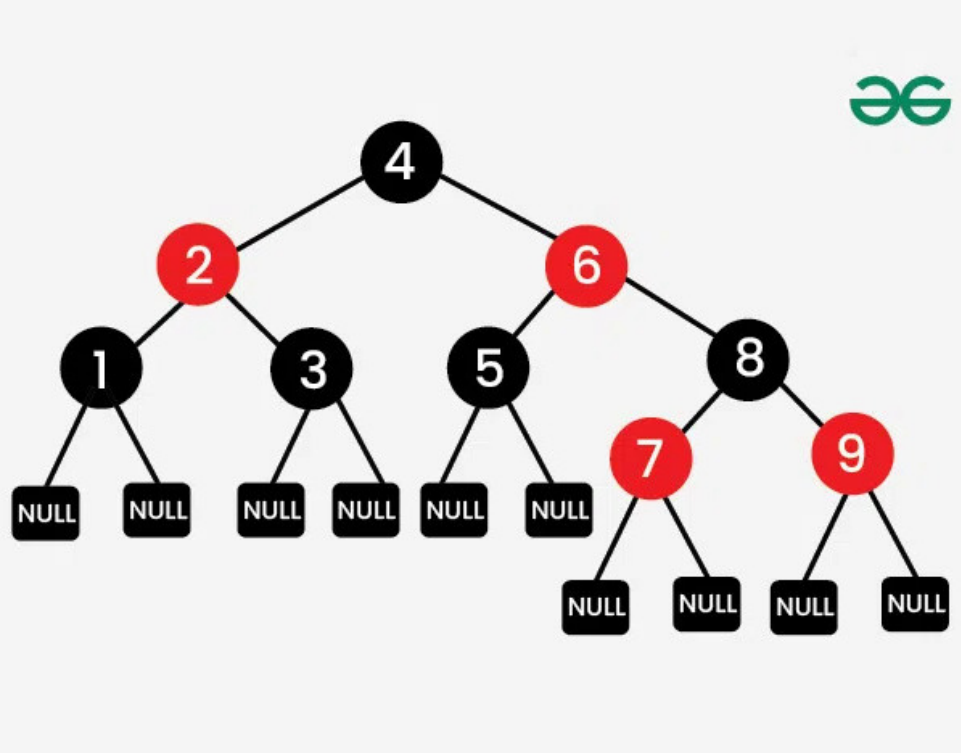


Source: Geeksforgeeks

- **Red-Black Tree**

  The **Red-Black Tree** solves this problem by maintaining a balanced structure through specific rules and properties:

  - **Node color:** Each node is either red or black.
  - **Root property:** The root of the tree is always black.
  - **Red property:** Red nodes cannot have red children (no two consecutive red nodes on any path).
  - **Black property:** Every path from a node to its descendant null nodes (leaves) has the same number of black nodes.
  - **Leaf property:** All leaves (NIL nodes) are black.



Source: Geeksforgeeks

- Compared with AVL Tree Red-black trees are more commonly used than AVL trees in the standard libraries of mainstream programming languages for associative containers due to several practical considerations related to performance, balance, and complexity. Here's why red-black trees are typically preferred:

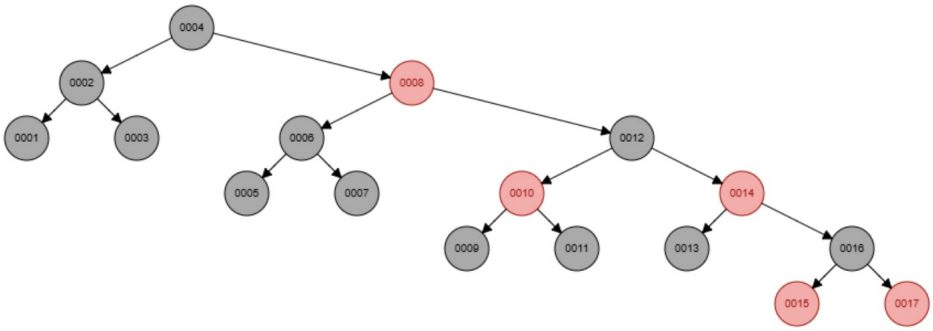
  1. **Insertion and Deletion Efficiency**

     Red-black trees are considered more efficient for insertions and deletions compared to AVL trees, particularly in terms of the number of rotations needed to maintain balance.

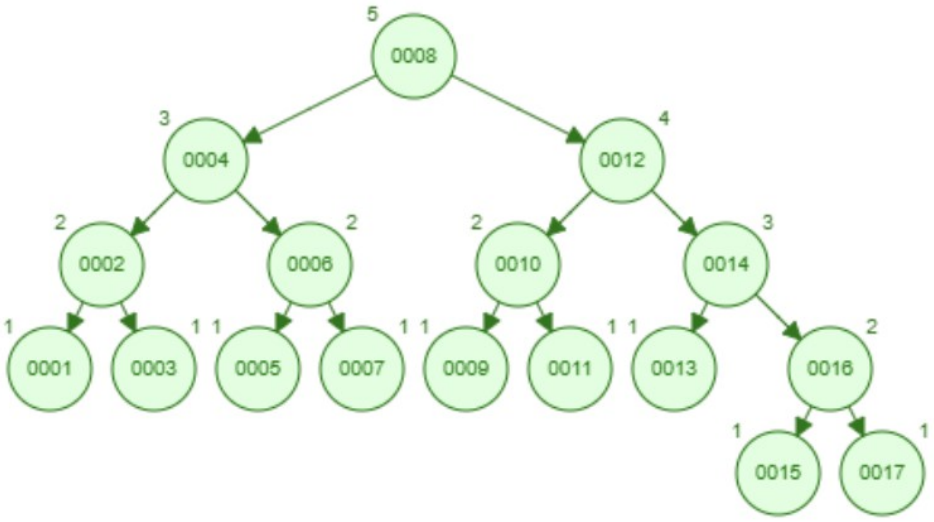
     In a red-black tree, at most one rotation is required to restore balance after an insertion or deletion.

     In contrast, AVL trees often require multiple rotations (up to two rotations) to restore balance after an insertion or deletion.

     Since insertions and deletions are common operations in associative containers, the reduced number of rotations makes red-black trees a more attractive choice in practical implementations.



Insert sequentially from 1 to 17 on Red-Black Tree



Insert sequentially from 1 to 17 on AVL Tree

2. **Relaxed Balance Condition**

   Red-black trees maintain a more relaxed balance condition compared to AVL trees. In a red-black tree, the heights of subtrees can differ by a factor of 2, whereas in an AVL tree, the height difference between subtrees of a node can be at most 1.

   This relaxed balancing in red-black trees results in fewer tree rotations during updates, which can lead to faster performance for many real-world workloads where operations are performed frequently.

3. **Search Efficiency**

   Both AVL and red-black trees provide O(log n) time complexity for search operations. However, due to AVL trees being more strictly balanced, they can theoretically provide slightly faster lookups.

   However, this difference is often negligible in practice. The real-world impact of the stricter balancing in AVL trees does not usually justify the higher costs of insertions and deletions, making red-black trees a better all-around choice.

4. **Real-World Workloads**

   In most real-world applications, the slight performance advantage that AVL trees might offer for search operations is outweighed by the more frequent insertions and deletions.

   Associative containers like map and set are often used in situations where dynamic updates are common, and the reduced cost of updates in red-black trees is a better fit for these scenarios.

As a result, red-black trees have become the favored choice in most standard libraries due to their balanced trade-offs across various operations and real-world applications.

2.3 Target Audience

RedBlack Defenseis designed primarily for students in the computer science field, especially those studying data structures and algorithms. It targets learners who seek an innovative and interactive approach to understanding complex topics like the Red-Black Tree. The game appeals to:

- **Computer Science Students:** Those enrolled in courses on data structures and algorithms who need a practical, hands-on way to grasp the Red-Black Tree's balancing rules and properties.
- **Beginners in Programming:** Newcomers who are just starting to explore data structures and want an engaging introduction. The Study Mode helps them build foundational knowledge with key concepts and interactive demos before experiencing the main gameplay.
- **Educators and Instructors:** Teachers looking for creative and interactive tools to supplement their traditional teaching methods. The game can be used in classrooms as a visual and practical aid to introduce Red-Black Trees to students in an engaging way.

Overall, RedBlack Defenseaims to make learning data structures more accessible and enjoyable for students at different levels of expertise while providing educators with an effective teaching resource.

2.4 Game Objectives

2.4.1 Educational Objectives

To help players understand the key concepts of Red-Black Trees, including their properties, insertion, deletion, and balancing mechanisms through interactive gameplay and study mode.

2.4.2 Gameplay Objective

Players must strategically place towers (nodes) along the path to defend against waves of enemies. The challenge is to maintain the Red-Black Tree's properties, such as node color and tree balance, to maximize the towers' efficiency and successfully fend off increasingly difficult waves of enemies.

2.5 Conclusion

RedBlack Defensecombines engaging tower defense gameplay with educational content, helping players understand the Red-Black Tree data structure. Through strategic tower placement and interactive challenges, players learn concepts like insertion, deletion, and balancing in a dynamic, medieval-themed environment.

3 Design

3.1 Backstory

In the distant kingdom of **BKU**, the land thrived under the protection of an ancient magical artifact known as the **Tree of Balance**. This mystical tree, blessed by the sages, maintained order and stability in the kingdom through its perfect structure and harmony. Each of its nodes was imbued with a unique power, represented by its color—red for swift attack and black for unwavering defense. Together, these nodes worked in perfect unison to ensure the prosperity of BKU.

However, peace was shattered when a rogue faction of siege engineers from the neighboring kingdom of **Gilipo** sought to destroy the Tree of Balance. Envious of BKU's prosperity, they unleashed waves of relentless siege vehicles—battering rams, catapults, and trebuchets—designed to breach the castle and bring ruin to the land. These enemies are cunning, adapting their paths and targeting weak points in the Tree of Balance to exploit any instability.

As the kingdom's last line of defense, you, a **keeper of the tree**, have been entrusted with its protection. Armed with the knowledge of the **Red-Black Tree's ancient rules**, you must strategically place towers imbued with the tree's power, ensuring they remain properly connected and balanced. Each decision you make—whether adding new nodes, rotating the tree, or repairing its structure after enemy attacks—will determine the survival of BKU.

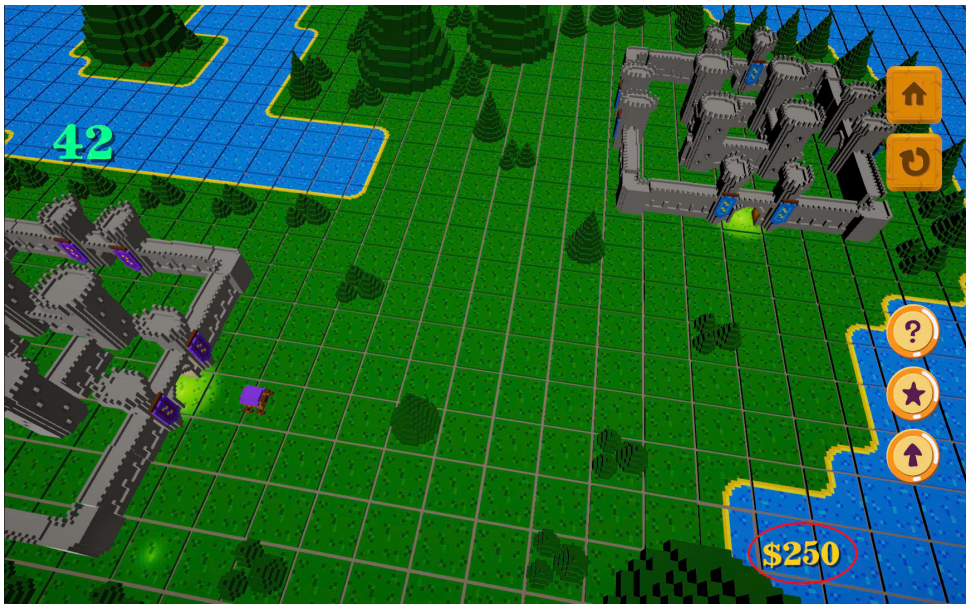
The battle is not just about brute strength; it is a war of knowledge and strategy. As waves of siege engines grow stronger, so must your understanding of the Tree of Balance. The people of BKU look to you for salvation. Will you rise to the challenge, master the art of the Red-Black Tree, and protect your kingdom from destruction? The fate of BKU lies in your hands.

3.2 Gameplay Mechanics

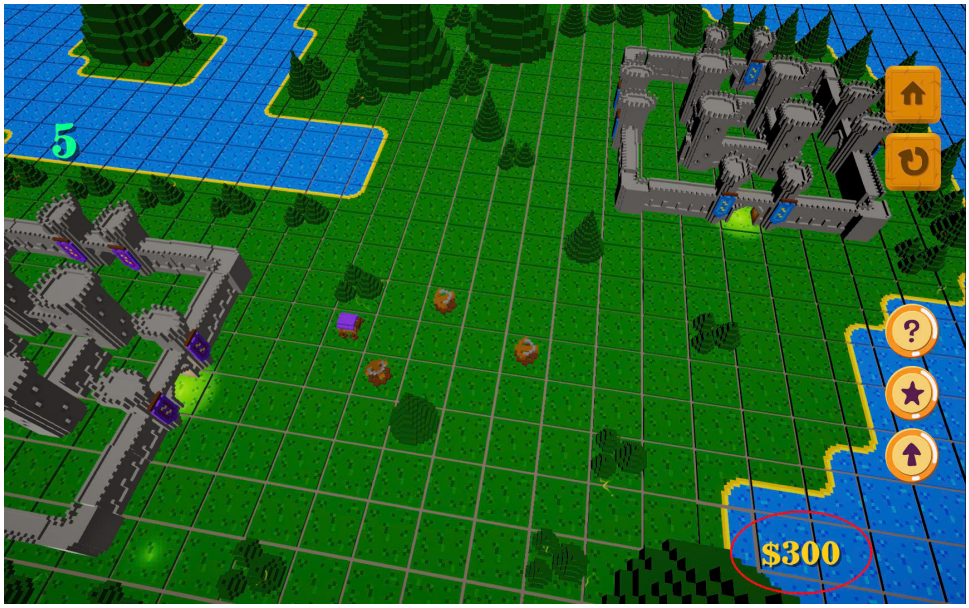
3.2.1 Tower Addition and Structure Maintenance

At the start of the game, the player is given an initial amount of gold to purchase Ballista Towers, which serve as nodes in the Red-Black Tree. Each time a siege minion is eliminated,

the player earns additional gold, allowing them to buy more Ballista Towers.



Initial money amount ($250 in default)



After defeating some sieges

Each tower comes with a randomly assigned value, but the player can choose its color — red in default, double-click the tower to change to black - when placing it on the map.

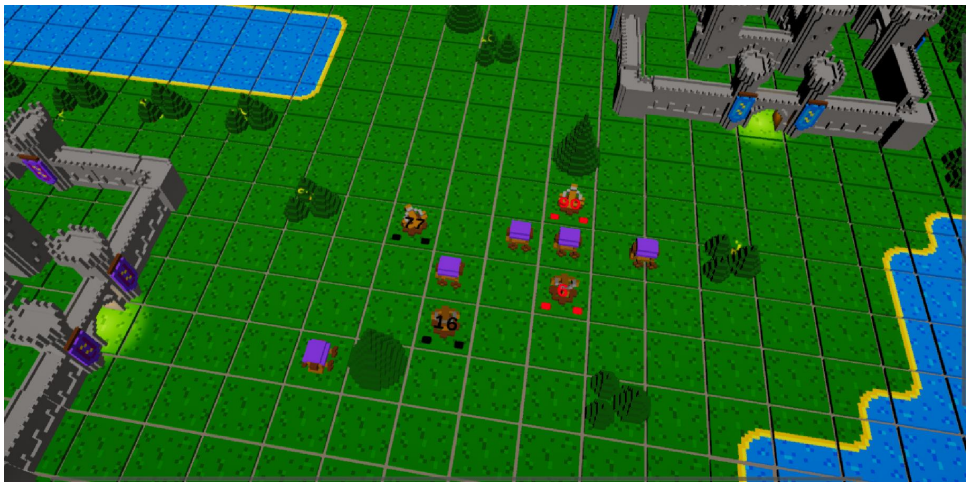
Red in default, double-click to black

To maximize the effectiveness of the towers, the player must connect the Ballista Towers into a Red-Black Tree structure that adheres to the tree's balancing rules by a drag and drop movement to connect the towers. Correctly placing and maintaining the structure provides buffs such as increased attack speed, enhanced firepower, or expanded range for the towers.

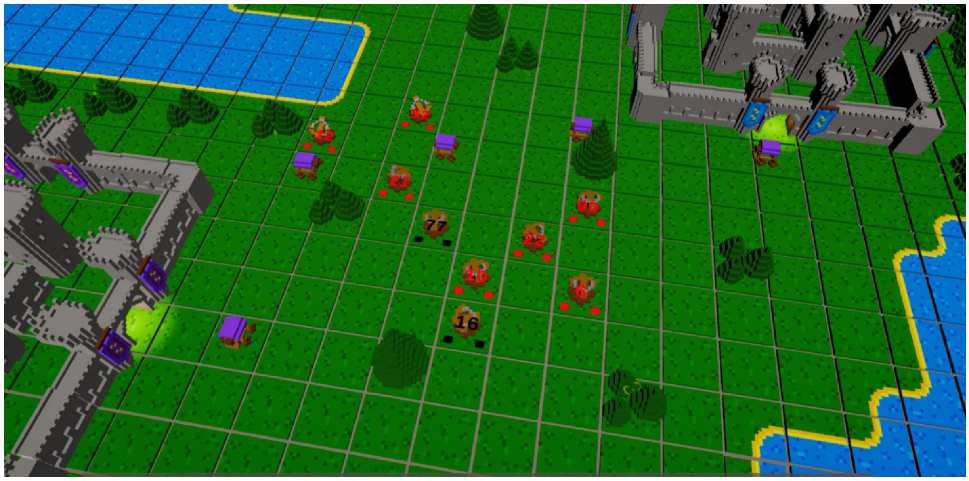
The system rewards strategic thinking, as the placement of towers not only determines the defensive strength but also impacts the siege engines' paths toward the castle. Properly managing gold, choosing tower attributes wisely, and maintaining a balanced tree are essential to surviving increasingly challenging waves of enemies.

3.2.2 Dynamic Enemy Path-finding

The waves of enemies automatically adjust their paths to find the shortest route to the castle (destination). The player's tower placement directly influences and alters the path-finding of the enemies, adding a layer of strategic depth to the game.



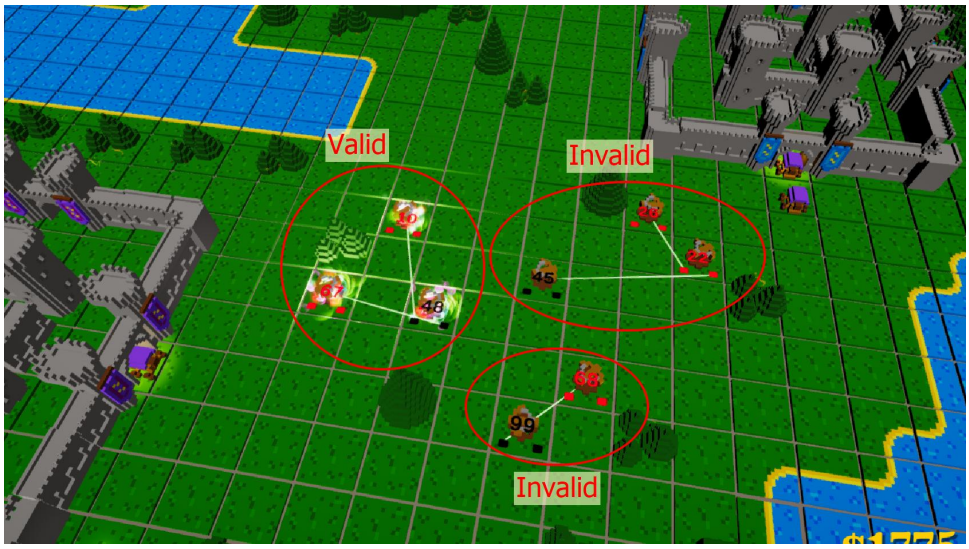
Enemy initial path



Enemy self-modified path

3.2.3 Wave Rewards and Upgrades

Players can create and manage multiple independent Red-Black Trees on the map, each functioning as a distinct structure. Only trees have from 3 nodes (3 connected towers) and above are considered a valid tree structure. The size of a Red-Black Tree determines the upgrades available for the towers within it. Larger, well-balanced trees grant greater rewards, enhancing the towers in that tree with increased firepower, faster firing rates, and extended firing range.



Valid and invalid Red-Black Tree

The strategic placement and connection of towers into correctly maintained Red-Black Trees not only ensure effective defense but also maximize the potential upgrades for each tower, encouraging players to expand and optimize their structures for long-term survival.

3.2.4 Winning and Losing Conditions

The game has no defined winning condition; players continue playing indefinitely, aiming for higher scores as the waves of siege vehicles grow progressively larger and more challenging.

The losing condition occurs when enemies successfully breach the castle. Each time an enemy reaches the base, a portion of the player's gold is deducted. If the player's gold is entirely depleted, the game ends, and the player must restart from the beginning.

Game over

3.3 Educational Integration

The Study Mode in RedBlack Defenseis designed to provide players with a comprehensive understanding of Red-Black Trees through interactive learning. It combines theoretical explanations with visual demonstrations to help players grasp the core concepts before engaging in the main gameplay. This mode includes:

3.3.1 Theoretical Guide and Visualization

Players are introduced to the key principles of Red-Black Trees, including properties, insertion, deletion, balancing, and rotations. The game uses circles and lines to represent nodes and tree structures, with animations demonstrating each operation step by step, making abstract concepts more concrete and easier to understand.

Players can enter Simulation, a feature designed to help them understand and interact with a Red-Black Tree in a hands-on way. In this mode, players can perform various operations, including inserting values, deleting nodes, and searching for specific values within the tree. Once a command is given, the game automatically executes the operation, including rebalancing the tree and determining the correct color of each node.

This interactive demonstration visually illustrates how Red-Black Tree properties are main-

tained during each operation, providing a clear understanding of complex concepts like rotations, balancing, and color adjustments. Simulation mode serves as an educational tool, allowing players to practice and internalize the rules of Red-Black Trees before applying their knowledge in the main gameplay.

Red-Black Tree Simulation

3.4 User Interface

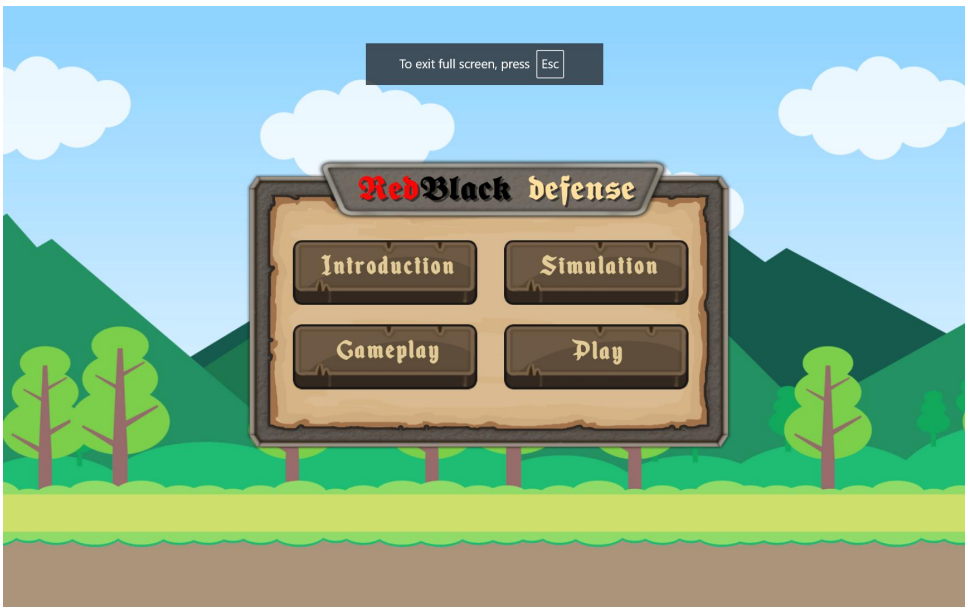
3.4.1 Visual and Aesthetic style

- **Overall**

  The game adopts a 3D pixel-style graphics with a cartoonish medieval theme, immersing players in a vibrant world filled with castles, wooden structures, lush forests designed with blocky, pixelated textures, resembling a handcrafted world built from medieval-themed voxel art.The atmosphere is lighthearted yet engaging, with colorful animations and whimsical designs bringing the environment to life.

- **Main Menu**

  The main menu is styled as a leather-bound box containing several options for navigating the game. Each option is presented as a button with a medieval font, and the background features a horizontally sliding forest scene, creating a dynamic and immersive atmosphere. When a button is pressed, it depresses with a smooth animation accompanied by a soft medieval sound effect, wooden click sound, enhancing the tactile feedback for the player.



Main Menu

- **Game Introduction**

  This mode allows the player to read an introduction detailing the backstory of the game and its educational content. Topics include the theory of trees, binary trees, binary

search trees, and Red-Black Trees, presented in an easily readable format with scrollable text. The window design matches the main menu, retaining the leather box style and sliding forest background. At the end of the introduction, a button labeled **"Got it"** is displayed. Clicking this button transitions the player back to the main menu, with a subtle animation and sound effect for a cohesive experience.

Game Introduction

- **Gameplay Instruction**

  This mode provides detailed instructions on how to play the game, such as how to purchase and place towers, manage Red-Black Trees, and deal with enemy waves. The design of this window mirrors the Game Introduction, maintaining consistency in the interface. The instructions are visually supported with icons or pixel-art illustrations to clarify gameplay mechanics. Once the player has reviewed the instructions, they can return to the main menu by clicking **"Return"** button or go straight to the main gameplay by **"Play!"** button .

Gameplay Instruction

- **Main Gameplay**

  Selecting the **Play** option transitions the player to the main gameplay window. This window introduces the active game world, where players can begin building and managing their Red-Black Tree structures while defending against siege waves.

  - **Exit Button**

    Located at the top-right corner, the square-shaped button features a house symbol to represent exiting to the main menu. When the player clicks this button, a pop-up prompt appears, asking, "Do you want to exit the game?". The player can select Yes to return to the main menu or No to continue playing.

  - **Restart Button**

    Positioned directly beneath the exit button, this button has the same square shape but features a restart arrow symbol. Clicking this button triggers a pop-up prompt that asks, "Do you want to restart the game?". Choosing Yes resets the game to its initial state, including waves and towers, while selecting No resumes the current session.

  - **Review Instruction Button**

    Found below the restart button, this circular button displays a question mark symbol. Clicking it opens the gameplay instructions window, allowing the player to quickly review how to play without leaving the game.

– **Tower Value Display Switch**

Positioned beneath the review button, this circular button has a star symbol. Clicking it toggles the display of the values of each tower on the map, making it easier for players to plan their strategy based on tower values.

– **Tower Connection Display Switch**

Located below the tower value button, this circular button displays an up arrow symbol. Clicking it toggles the visibility of the nodes and edges connecting the towers, visually representing the Red-Black Tree structure. In this mode, players can drag and drop to connect towers, ensuring the tree maintains its properties.

– **Current Money Display**

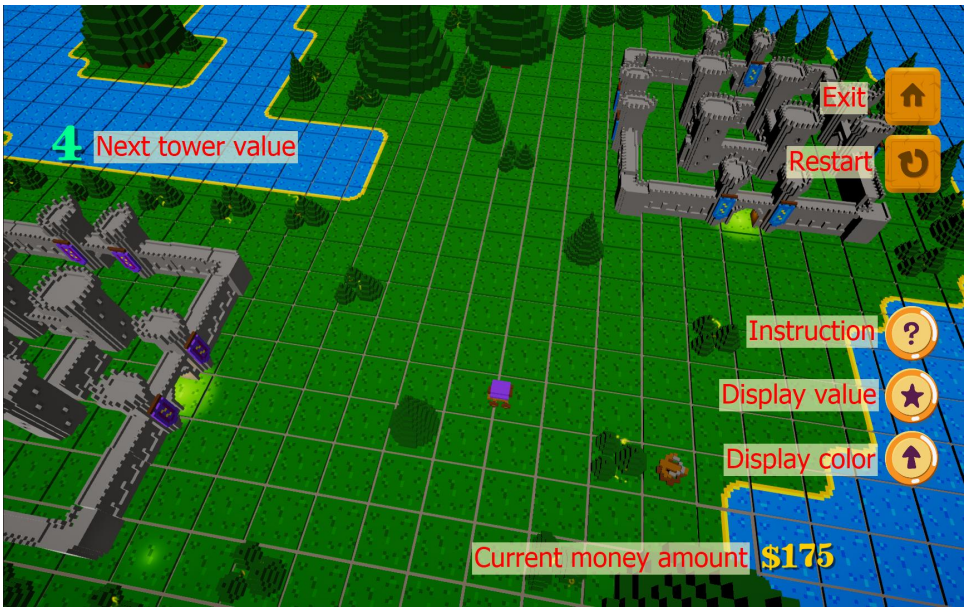
Positioned at the bottom center of the screen, this interface element continuously updates to display the player's current money. It reflects changes immediately whenever a siege vehicle is eliminated (earning gold) or breaches the base (deducting gold).

– **Next Tower Value Display**

Located at the top-left corner, this interface shows the value of the next tower available for placement on the map. This allows players to plan their tree structure in advance, taking into account the upcoming tower's value.

– **Buffed Tower**

Buffed towers are displayed with flaming surrounding them and glowing effect. They have more rapid firing animation.



Gameplay Interface

– **Game over**

When the player is defeated, a pop-up prompt appears to indicate the end of the game. This prompt displays the player's final score, providing feedback on their performance.

Below the score, two buttons are presented:

* **Restart:** Clicking this button resets the game to the initial state, including towers, waves, and resources, allowing the player to start fresh immediately.

* **Exit:** Clicking this button returns the player to the main menu, where they can explore other options or start a new session.

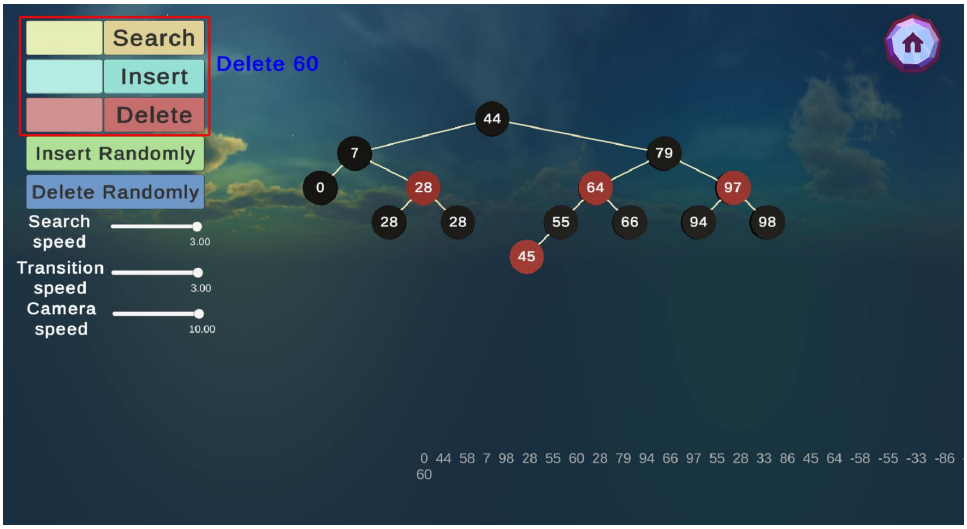
Enemy self-modified path

- **Simulation**

  For the simulation mode, the graphics are intentionally minimalistic and focus on clarity. Nodes are represented as simple colored circles, connections as straight lines, and tree operations as smooth, animated transitions. Numerical values are displayed on each node, and operations like insertion, deletion, and rebalancing are demonstrated with clean text annotations and polygonal representations of the tree's structure. This visual approach ensures the educational aspect of the game remains straightforward and easily understandable.

  – **Search/Insert/Delete**

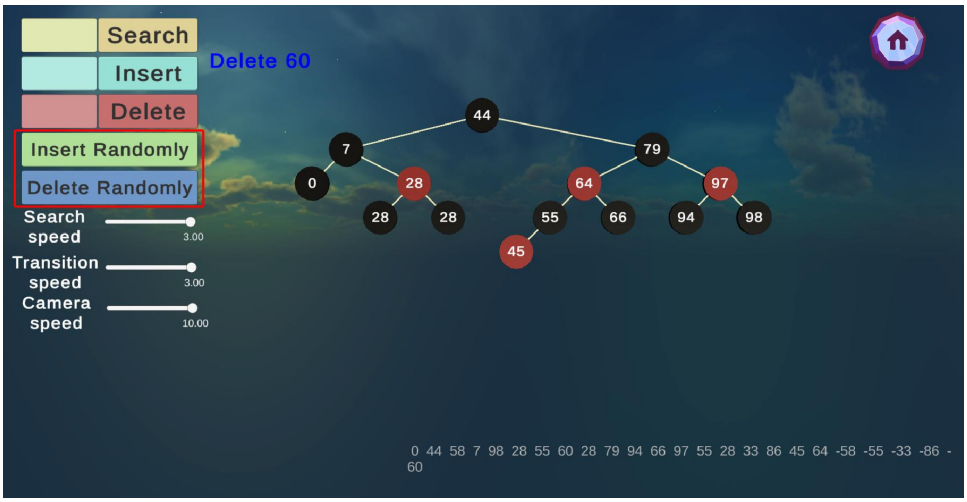
    In this mode, the player can manually enter a number into the corresponding box to perform the operations of insert, delete, or search on the Red-Black Tree. The game automatically carries out the selected operation, visually demonstrating the result, including tree balancing, node color changes, and adjustments to the tree structure. This allows the player to actively interact with the tree and understand the underlying concepts.



Search/Insert/Delete

  – **Insert/Delete Randomly**

    This option functions similarly to the manual search/insert/delete, but the game randomly selects a number for either insertion or deletion. The player is then able to observe how the game handles these random operations, learning how the Red-Black Tree reacts to unpredictable changes. This feature enhances the player's understanding of tree balancing and the importance of maintaining the tree's properties during dynamic operations.



Insert/Delete Randomly

– **Search Speed Bar**

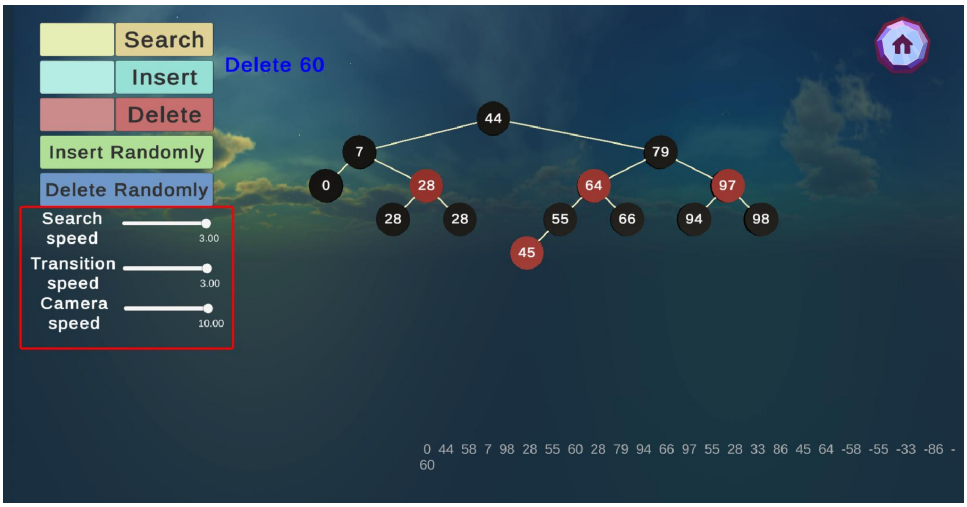
The search speed bar allows the player to adjust the speed at which the game performs the searching operation. Moving the bar allows the player to slow down or speed up the tree traversal when searching for a specific value, offering a customizable pace for learning and understanding the search process.

– **Transition Speed**

The transition speed bar controls the speed of animations that occur during tree changes, such as when inserting, deleting, or balancing nodes. By adjusting this bar, the player can make the animation faster or slower, which helps when they want to watch the transitions in more detail or speed through repetitive operations.

– **Camera Speed**

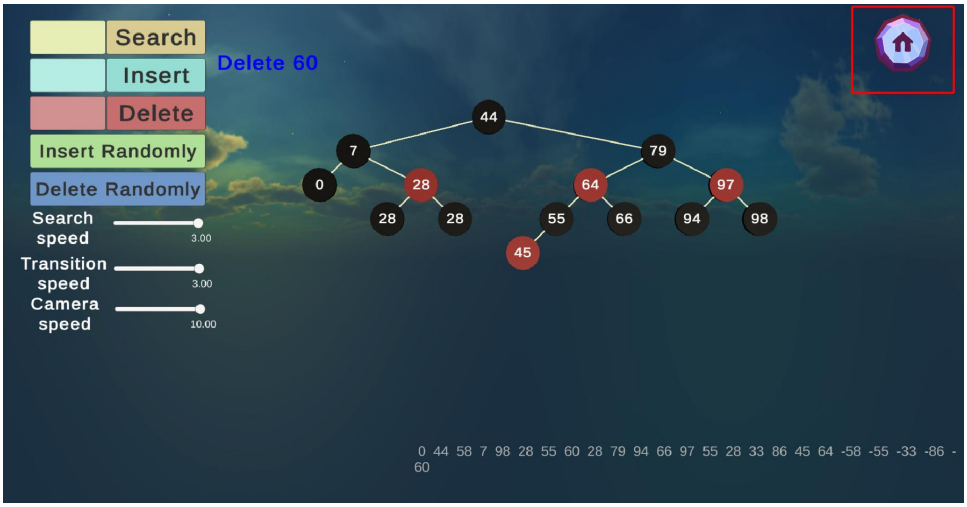
The camera speed control adjusts the speed of the zoom feature when viewing the tree. The player can zoom in or out on the tree at varying speeds, allowing them to get a closer look at specific nodes or view the entire tree structure from a higher level. This feature supports better exploration and interaction with the tree structure.



Enemy self-modified path

– **Exit**

Exit to main menu



Enemy self-modified path

3.5 System Requirements

- **Platform:** Windows (7/10/11)
- **Programming Language:** C# for logic and Unity for game development.
- **Graphics Library:** Unity's built-in tools for rendering.
- **Data Structure Integration:** Implement Red-Black Tree using custom data structures.

4 Testing

The following configurations are crucial for the gameplay experience. These values were carefully chosen after extensive playtesting to ensure an optimal balance:

- **Initial gold amount:** $150
- **Gold reward for each defeated enemy:** $10
- **Gold loss for each escaped enemy:** $25
- **Initial enemy health:** 1 Hp
- **Proportional enemy health per wave:** 1.2
- **Enemy speed:** 1
- **Enemy pool size:** 12
- **Time between two enemies:** 2 seconds
- **Tower cost:** $40
- **Tower's original shooting rate:** 0.75
- **Tower's original shooting range:** 20
- **Tower's enhanced damage:** Tower's original damage multiplied by the enhancement factor.
- **Tower's enhanced shooting rate:** Tower's original shooting rate multiplied by the enhancement factor.
- **Tower's enhanced shooting range:** Tower's original shooting range multiplied by the enhancement factor.
- **Enhance factor:** The enhancement factor is determined by the formula:

$$\text{Enhance factor} = 1.2^{N-2}$$

N: Number of nodes of the valid tree in which the tower belongs
Ex: If the tree has 4 nodes, the Enhance factor will be $1.2^{4-2} = 1.2^2 = 1.44$

5 Future Improvements

To enhance the overall quality, engagement, and educational value of the game, the following future improvements are proposed:

5.1 Development

5.1.1 Configuration Analysis

To maintain player motivation and avoid frustration, the game should implement a dynamic difficulty adjustment mechanism. This system would adapt the game's challenge level based on the player's performance, making it easier or harder to ensure an engaging and balanced experience for all skill levels.

5.2 In-game

5.2.1 Display

- **Game Logo At Starting Screen**

  Currently, the game lacks a logo at the main menu. Adding a visually appealing logo will create a strong first impression and improve the overall aesthetics of the main menu.

- **Credits**

  Including a credits section will acknowledge the team members and their contributions, offering transparency and appreciation for the development process.

- **Patch Notes**

  Introducing patch notes will inform players about the game version and any recent updates, encouraging them to stay up-to-date and understand new features or fixes.

- **Button size**

  The button sizes and positions are currently static. Adopting a dynamic resizing mechanism will ensure that buttons adjust to different screen sizes or container dimensions, improving accessibility and usability across devices.

5.2.2 Simulation Mode Enhancements

- **Auto-Adjusted Camera Angle and Zooming**

The camera should automatically adjust to display the entire tree structure during operations like insertion, deletion, or search. This feature will provide a clearer view of the tree's changes while allowing manual camera control for focused exploration.

- **Step-by-Step Explanations**

  Adding text-based explanations for tree operations will enhance the educational aspect, showing players why and how nodes are inserted, deleted, or searched, and explaining the balancing process.

5.2.3 Gameplay Enhancements

- **Pause Button**

  Adding a pause button will allow players to temporarily halt the game, providing the option to return to the main menu or restart their session.

- **Selling Tower**

  Players should have the ability to sell placed towers to reclaim a portion of their investment, enabling more flexible strategies.

- **Tower's position modification**

  After placing a tower, players should be allowed to relocate it, adding strategic depth to gameplay.

- **Gameplay Diversity**

  Introducing a variety of enemy types (e.g., faster units, units with special abilities) and tower designs (e.g., area-effect towers, long-range snipers) will enrich gameplay. Additionally, new maps with unique terrains can provide diverse challenges and replayability.

- **Wave delay**

  A delay between waves will allow players time to analyze the situation and adjust their strategies, improving gameplay pacing.

- **Scoreboard and Rankings**

  Implementing a scoreboard will track and display high scores, motivating players to aim for better results. In the future, this feature could evolve into an online ranking system, adding a competitive edge to the game.

By incorporating these improvements, the game can achieve higher educational value, increased replayability, and better engagement for its players, ultimately offering a more comprehensive and enjoyable experience.

6 Conclusion

Red-Black Defense offers a unique fusion of education and entertainment, teaching players the intricacies of Red-Black Trees through engaging tower defense mechanics. With its interactive simulations and dynamic gameplay, the game provides an innovative approach to mastering data structures while delivering an enjoyable gaming experience. Designed with room for future enhancements, Red-Black Defense promises to be a valuable tool for students, educators, and gaming enthusiasts seeking to combine learning with fun.

7 External Resources

7.1 User Interface

1. **fantasy wooden gui free 103811**
2. **game gui buttons 96277**
3. **fantasy skybox free 18353**
4. **free 2d cartoon parallax background 205812**

7.2 VFX

1. **48 particle effect pack 13998**

7.3 Sound Effects

1. **Cartoon Game Theme Loop 4**
2. **Tower Defense Simulator OST - It's Getting Frosty**
3. **Battle Cats Music: Battle Theme 1 Default**
4. **Bloons TD Main Theme**

7.4 Fonts

1. **Alice In Wonderland**
2. **Enchanted Land**
3. **Fette National Fraktur**

——————————END——————————