

CESS
(Journal of Computer Engineering, System and Science)

Available online: <https://jurnal.unimed.ac.id/2012/index.php/cess>

ISSN: 2502-714x (Print) | ISSN: 2502-7131 (Online)



**Implementation of Data Structures in an Educational Detective Game:
Catch U Bro**

**Rafi Rachman^{1*}, Fikri Alfathir Solehudin², Ikhwan Rizki Ramadhan³, Pascalis Alvin Futari
Ardi⁴, Muhammad Taufik Dwi Putra⁵**

^{1,2,3,4,5}Prodi Teknik Komputer, Universitas Pendidikan Indonesia Kampus UPI di Cibiru, Indonesia

Jl. Pendidikan No.15, Cibiru Wetan, Kec. Cileunyi, Kabupaten Bandung, Jawa Barat 40625

Email: ¹rafirachman@upi.edu, ²alfathir@upi.edu, ³ikhwanrizki_20@upi.edu, ⁴pascalisalvinfa@upi.edu,
⁵tdputra@upi.edu

**Corresponding Author*

ABSTRACT

Data Structures and Algorithms (DSA) are essential topics in computer science education, but many students experience difficulties in understanding abstract concepts through conventional learning methods. This study presents *Catch U Bro*, an educational detective game designed to introduce DSA concepts through interactive gameplay. In the game, players investigate suspects using multiple information brokers that represent different data structures, including stacks, queues, and hash maps. Each broker applies a different data retrieval mechanism, requiring players to interact with the data according to the implemented structure. The system was developed as a desktop-based game using the Godot engine with randomized suspect data and clue generation to create varied gameplay experiences. The implementation demonstrates how DSA concepts can be integrated into detective-style game mechanics to provide a more engaging and practical learning medium. Functional (black-box) testing confirmed that all broker mechanisms and the investigation workflow operated as intended, indicating that the proposed integration is technically feasible as a learning medium. Empirical evaluation of its effect on students' conceptual understanding is identified as a direction for future work.

Keywords: *Data Structures; Educational Game; Detective Game; Algorithms.*

1. INTRODUCTION

Data Structures and Algorithms (DSA) are fundamental subjects in computer science education and are widely applied in software development and modern computing systems. Concepts such as stacks, queues, and hash maps are commonly used in various applications. However, many students still experience difficulties in understanding DSA concepts because data structures are inherently abstract and dynamic, and students often struggle with



This open access article is distributed under a Creative Commons Attribution (CC-BY) 4.0 license

misconceptions and insufficient prerequisite programming knowledge [1]. To address these challenges, previous studies have explored visualization-based learning approaches that help students understand abstract data structures and algorithms more effectively [2].

The development of educational games has become a promising approach to improve interactive learning experiences. Educational games can increase student engagement and motivation while enabling students to understand abstract concepts through interactive activities and visualizations. In recent years, game-based learning has been increasingly utilized in computer science education because it allows students to learn through direct interaction and problem-solving activities. Through interactive gameplay and visualizations, abstract concepts can be represented more clearly, helping students understand DSA concepts more effectively compared to conventional learning methods [3].

To further understand the potential of educational games in computer science education, several studies have explored the implementation of educational games and gamification techniques in learning systems. Research on gamification shows that interactive learning environments can improve motivation and student participation [4]. Other studies also demonstrate that simulation-based learning can help students understand technical concepts more effectively through visualization and interaction [4]. However, many existing educational games in computer science primarily focus on programming concepts, problem-solving activities, or individual data structures. Previous studies indicate that educational games related to data structures are still less common than those focusing on fundamental programming concepts, and existing games often concentrate on a single data structure, such as stacks [5], [6].

Table 1 summarizes this gap by comparing Catch U Bro with related educational games and studies on data structures, based on data structure coverage, platform, and the specific limitations each leaves unaddressed.

Table 1. Comparison of Catch U Bro with Related Educational Games and Studies

Study	Data Structures	Platform / Context	Gap Addressed by Catch U Bro
Su et al. [3], 2021	Algorithms & data structures (general)	Visualization-based teaching tool	Focuses on visualization rather than a narrative-driven, multi-structure investigation
Rojas-Salazar [4], 2022	Data structures via analogies	Analogy-based game-based learning (thesis)	Analogy-based rather than direct operational interaction (e.g., push/pop, enqueue/dequeue)
Bedreski et al. [6], 2025 (DELIMITER)	Stack only	Serious game (delimiter/bracket-matching)	Single data structure; not integrated with other structures
Pratiwi & Setiawan [17], 2023 (STANQUE)	Stack, Queue	Android application, separate modules	Two structures taught as separate modules, not embedded in one cohesive scenario; no Hash Map

Catch U Bro (this study)	Stack, Queue, Hash Map	Godot desktop application, detective investigation narrative	Integrates three data structures into one cohesive gameplay loop with a unified narrative and consistent operation-based interaction
--------------------------	------------------------	--	--

Consequently, there is an opportunity to develop educational games that integrate multiple data structures as interconnected gameplay mechanics within a contextualized learning environment. This gap motivates the development of *Catch U Bro*, which incorporates stack, queue, and hash map concepts into a detective-based investigation scenario.

To address these limitations, this study proposes *Catch U Bro*, an educational detective game designed to introduce DSA concepts through interactive investigation gameplay. Previous studies have shown that game-based learning approaches combined with interactive visualizations can support the learning of data structures and algorithms by increasing student engagement and helping learners understand abstract concepts more effectively [3]. In the game, players act as detectives who must identify suspects using information obtained from multiple data brokers, where each broker represents a different data structure. For example, document archives implement stack behavior using a Last In First Out (LIFO) mechanism, interview systems apply queue structures through a First In First Out (FIFO) process, and digital databases utilize hash maps for efficient data retrieval based on unique identities. Each broker provides distinct interaction mechanics, allowing players to experience contextualized implementations of data structures directly through gameplay and problem-solving activities. Educational games have also been shown to increase student engagement and support active learning within immersive learning environments [7].

The proposed game was developed as a desktop-based application using the Godot game engine with a data-driven investigation system and interactive broker-based mechanics. The main contribution of this research is the integration of multiple data structures into detective-style gameplay mechanics as an interactive learning medium for DSA. Unlike traditional educational systems that mainly focus on theory and coding exercises, this approach provides a practical simulation environment where players can learn data structures through exploration, investigation, and decision-making activities [8]. In the game, players interact with different information brokers representing stack, queue, and hash map structures to obtain clues and identify suspects. Unlike existing educational games that mainly focus on isolated visualization or coding exercises, *Catch U Bro* integrates multiple data structures as interconnected gameplay mechanics within a detective investigation scenario. Therefore, this study aims to provide an engaging alternative learning medium for DSA concepts that is designed to support students' understanding, logical thinking, and problem-solving skills, consistent with findings from prior game-based learning research [9].

2. RESEARCH METHOD

2.1. Educational Game Design

Catch U Bro was designed as an educational detective game that integrates data structure learning into an investigation-based gameplay experience. Educational games have

been shown to provide active and engaging learning environments that support critical thinking, decision-making, and problem-solving through interactive gameplay [10], [11].

The game places players in the role of a newly appointed detective who is assigned to identify a criminal suspect using fragmented information obtained from multiple information brokers. The detective theme was selected to provide a contextual learning environment in which data structure concepts can be represented through investigation activities, exploration, and clue collection, allowing players to interact directly with learning mechanics while solving investigative challenges [11].

Unlike conventional learning systems that introduce data structures through theoretical explanations or programming exercises, Catch U Bro employs a guided learning approach that combines narrative elements, tutorial dialogues, and interactive gameplay mechanics. Narrative-based game design has been recognized as an important component of digital game-based learning (DGBL) and may positively influence learner engagement, motivation, and learning outcomes [12]. Furthermore, game-based learning environments support experiential learning by allowing learners to acquire knowledge through active participation and learning-by-doing activities [13]. Before interacting with the main gameplay features, players are introduced to the game environment through a story-driven tutorial involving several non-playable characters (NPCs), including a senior detective, police officers, and information brokers. These characters provide explanations regarding the investigation process and the data structure concepts implemented within the game.

The tutorial system gradually introduces the operations and behaviors of each data structure before players perform the corresponding actions. This design follows the principle that game tutorials are most effective when players receive guidance while being given opportunities to practice the introduced mechanics and obtain timely feedback [14], [15]. Through dialogues and guided instructions, players learn how stack operations such as `push()`, `pop()`, and `peek()` are implemented through the Stack Broker mechanism, which follows a Last-In First-Out (LIFO) information retrieval process. Similarly, queue operations such as `enqueue()` and `dequeue()` are introduced through the Queue Broker mechanism, where suspect information is processed according to a First-In First-Out (FIFO) sequence. In addition, the Hash Map Broker demonstrates direct information retrieval using suspect identification numbers as keys, reflecting the behavior of hash map data structures. After completing the tutorial phase, players interact directly with various broker systems to collect suspect information. Information considered relevant can be stored in a suspect list and later compared with official police records to determine the correct suspect. By combining narrative guidance, character-based explanations, and practical interaction, the game aims to reduce learning difficulties associated with abstract data structure concepts while maintaining player engagement throughout the investigation process.

Figure 1 illustrates the overall gameplay flow of Catch U Bro. Players begin by completing the tutorial phase before interacting with information brokers representing different data structures. Information obtained from the brokers can be stored in the Suspect List through the Keep mechanism. Players may continue collecting additional information by revisiting the broker selection stage until sufficient evidence has been gathered. The collected suspect information is then used during the final investigation stage to identify the correct suspect.

The game concludes with either a success or failure ending depending on the player's final decision.

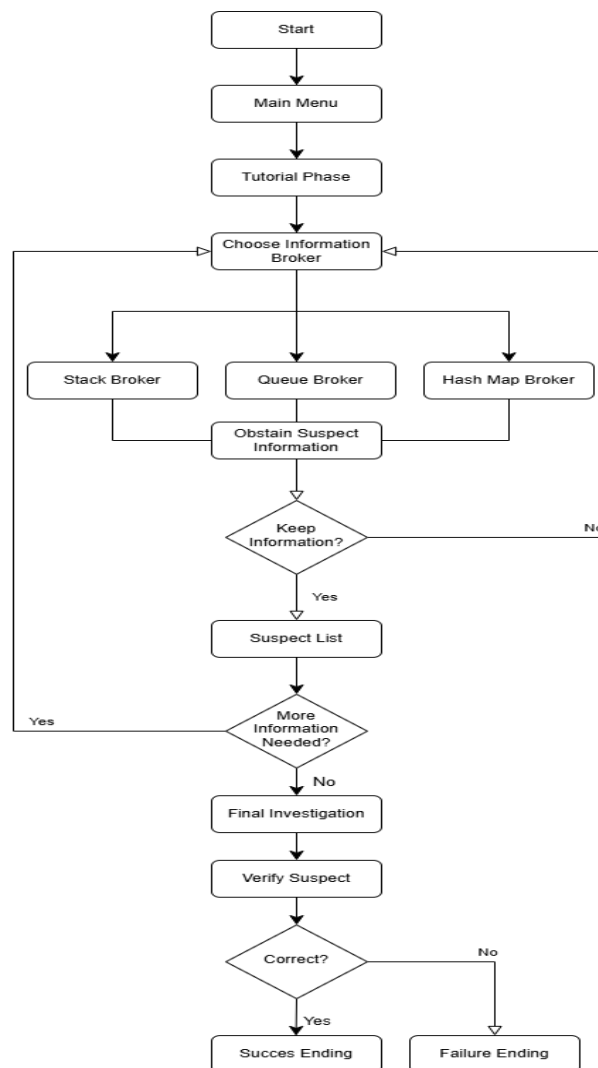


Figure 1. Investigation Flow of Catch U Bro

Figure 1 illustrates the overall gameplay flow of Catch U Bro. Players begin by completing the tutorial phase before interacting with information brokers representing different data structures. Information obtained from the brokers can be stored in the Suspect List through the Keep mechanism and later used during the final investigation stage to identify the correct suspect. The game concludes with either a success or failure ending depending on the player's final decision.

2.2 Data Structure Integration

Catch U Bro integrates several data structures into its gameplay mechanics to help players understand their practical applications within a detective investigation scenario. Each data structure is represented through a different information broker and interaction system, allowing players to learn data structure operations while collecting clues and identifying suspects.

2.2.1. Stack Broker

The Stack data structure is implemented through the Stack Broker, an information broker responsible for managing investigation documents. Investigation information is organized as a stack of document records, where the most recently added document is placed at the top of the collection. Players interact with the broker through three primary operations: `peek()`, `pop()`, and `push()`. The `peek()` operation allows players to inspect the top document without removing it from the stack. The `pop()` operation removes the top document and reveals the next document underneath. The `push()` operation allows players to request additional investigation information, which is then added to the top of the document stack. Through this mechanic, players directly experience the Last In First Out (LIFO) behavior of the Stack data structure while searching for investigation clues. The use of visual and interactive representations of data structures can support learners in understanding the behavior of data structures and related algorithmic concepts more effectively [16].

2.2.2. Queue Broker

The Queue data structure is implemented through the Witness Queue Broker. Both systems utilize the First In First Out (FIFO) principle but represent different forms of information processing. The Witness Queue Broker manages a queue of witnesses waiting to provide testimonies. Players can inspect the first witness using the `peek()` operation and remove a witness from the front of the queue using `dequeue()` after receiving the testimony. Additional witnesses may be added through the `enqueue()` operation.

Through these mechanics, players learn how Queue structures process information sequentially according to the First In First Out (FIFO) principle, where the earliest inserted element is the first element to be processed or removed from the queue [17].

2.2.3. Hash Map Broker

The Hash Map data structure is implemented through the Hash Map Broker, a specialized information broker that provides direct access to a criminal database. Unlike the Stack and Queue brokers, which require sequential access to information, the Hash Map Broker performs direct searches using unique identification numbers. Players enter a suspect identification number obtained from previous clues into the search system. The identification number acts as a key, while the corresponding suspect profile stored in the database represents the value. After the search operation is executed, the system immediately retrieves the associated suspect information if a matching record exists.

The search process produces two possible outcomes: FOUND when the identification number exists in the database and NOT FOUND when no matching record is available. This mechanic demonstrates how Hash Maps utilize key-value pairs to support efficient data retrieval without sequential searching. Information collected from the Stack, Queue, and Hash Map brokers can be stored by the player through the Keep mechanism and later reviewed in the Suspect List. The collected suspect information is then used during the final investigation stage, where players compare evidence and determine the most likely suspect.

2.3 System Architecture

The system architecture of Catch U Bro was designed using a scene-based architecture provided by the Godot Engine. The architecture integrates dialogue-driven tutorials, broker

interaction systems, suspect management, and data-driven content management to support both gameplay and educational objectives. Consistent with modular software architecture principles, which organize system functionality into specialized and interconnected components, the system is divided into four primary components: the Dialogue System, Broker System, Suspect Management System, and Data Management System [18], [19].

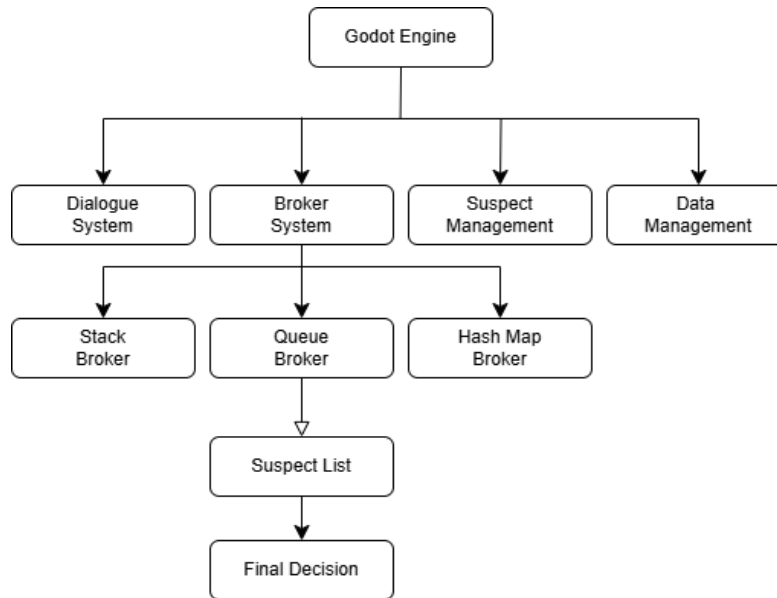


Figure 2. System Architecture of Catch U Bro

2.3.1 Dialogue System

The Dialogue System serves as the primary instructional component of the game. Before interacting with gameplay mechanics, players receive guidance through conversations with non-playable characters (NPCs), including Senior Bram, police officers, and information brokers. The dialogue content is stored separately in a structured JSON file and is dynamically loaded according to the current game state. A dedicated dialogue interface is responsible for displaying tutorial instructions and story progression throughout the game. This approach enables educational content to be delivered gradually before players interact with data structure mechanics. In addition, the game implements a progressive tutorial mechanism that tracks tutorial completion status. Tutorials are displayed only when required, preventing repetitive explanations while maintaining a smooth gameplay experience.

2.3.2 Broker System

The Broker System represents the core learning environment of Catch U Bro. Players obtain investigation clues by interacting with multiple brokers, where each broker represents a specific data structure implementation. The Stack Broker simulates document archive management using stack operations such as `peek()`, `pop()`, and `push()`. The Queue Broker implements witness and report processing using queue operations such as `peek()`, `dequeue()`, and `enqueue()`. The Hash Map Broker provides direct suspect information retrieval using unique identification numbers as search keys. Each broker is implemented as an independent game scene with its own interaction logic while sharing a common objective of providing information required for the investigation process.

2.3.3 Suspect Management System

The Suspect Management System is responsible for storing and organizing suspect information collected throughout the investigation. During gameplay, players may decide to preserve important information using the Keep mechanism. Selected information is then stored and displayed in the Suspect List interface.

The Suspect List functions as a centralized workspace where players review collected evidence and compare suspect profiles before making a final decision. This system supports investigative reasoning by allowing players to revisit previously obtained information at any time during the game. The final outcome of the investigation depends on the information accumulated within this system and the player's ability to identify the correct suspect based on available evidence.

2.3.4 Data Management System

Catch U Bro adopts a data-driven approach for managing game content. Core game information, including dialogue content, suspect profiles, criminal identification records, broker-related information, and investigation progress data, is stored in external JSON files and loaded dynamically during runtime. This approach separates game content from gameplay logic, allowing game data to be maintained and updated without modifying the core system implementation.

The Data Management System provides information to other components of the game, including the Dialogue System, Broker System, and Suspect Management System. Whenever players interact with brokers, progress through tutorial dialogues, retrieve suspect information, or review collected evidence, the required data is obtained through this management system. By centralizing data storage and retrieval, the architecture supports both educational content delivery and detective-style gameplay while maintaining modularity, consistency, and ease of future development, which is consistent with data-centered system integration approaches commonly used in game-based and learning-oriented software architectures [20].

2.4 Development Method

The development of *Catch U Bro* followed a software engineering process consisting of analysis, design, implementation, and testing activities. This process was adopted to systematically transform educational objectives and gameplay requirements into a functional educational game that integrates data structure concepts within a detective investigation scenario [21].

2.4.1 Analysis

The analysis stage focused on identifying learning difficulties associated with Data Structures and Algorithms (DSA) and determining suitable gameplay mechanics for educational purposes. Educational objectives were established to help players understand the practical implementation of Stack, Queue, and Hash Map structures through interactive investigation activities. During this stage, learning requirements, gameplay requirements, tutorial requirements, and suspect investigation workflows were identified. The analysis also defined the role of information brokers as learning agents responsible for introducing different data structure concepts.

2.4.2 Design

The design stage involved the development of tutorial dialogues, broker interaction mechanics, learning activities for Stack, Queue, and Hash Map concepts, the Suspect List system, and the overall user interface. These components were designed based on the investigation workflow defined during the analysis stage and served as the foundation for game implementation.

2.4.3 Implementation

During the implementation stage, several core components were developed, including the Dialogue System, Stack Broker System, Queue Broker System, Hash Map Broker System, Suspect Management System, and Data Management System. These components were implemented as independent scenes within the Godot Engine and connected through the scene management architecture. Game data, including dialogues, suspect information, and investigation content, were stored in external JSON files to support a data-driven architecture.

2.4.4 Testing

Black-Box testing was conducted to verify the correct execution of Stack, Queue, and Hash Map operations. The testing process also evaluated the storage and presentation of suspect information, dialogue progression, tutorial delivery, and scene transitions. The results confirmed that all major gameplay components operated according to their intended functionality.

The evaluation in this study was limited to functional (black-box) testing of system components, as described above. No empirical assessment of learning outcomes, such as pre-test/post-test measures or standardized usability instruments (e.g., SUS or UEQ), was conducted at this stage. This scope was chosen to first establish that the system functions as intended before proceeding to learning-outcome evaluation, which is identified as future work (see Section 4).

3. RESULT AND DISCUSSION

3.1 Game Implementation Results

The implementation resulted in a fully functional educational game containing tutorial interactions, broker-based learning activities, suspect management, and investigation mechanics. The game was developed using the Godot Engine and consists of several interconnected modules, including the Dialogue System, Broker System, Suspect Management System, and Data Management System. The Dialogue System provides tutorial guidance and narrative progression through conversations with non-playable characters (NPCs). Tutorial content is dynamically loaded from external JSON files and displayed according to the player's progress. To prevent repetitive explanations, tutorial completion status is recorded and checked before a dialogue sequence is displayed. The Broker System serves as the primary learning environment in which players interact with data structure implementations represented by different information brokers. Each broker is implemented as an independent game scene with its own interaction logic and educational objectives. Through these interactions, players collect suspect information required for the investigation process.



Figure 3. Main Gameplay Interface

Information obtained from the brokers can be preserved using the Keep mechanism and stored in the Suspect List. The Suspect Management System allows players to review collected evidence and compare suspect profiles before making a final decision. The game concludes when the player identifies a suspect during the final investigation stage, resulting in either a success or failure ending depending on the correctness of the selected suspect.

3.2 Data Structure Learning Implementation

3.2.1 Stack Broker Results

The Stack Broker was implemented as a document archive system that demonstrates the Last In First Out (LIFO) behavior of a stack data structure. Suspect information is represented as investigation documents arranged in a stack, where the most recently added document is positioned at the top of the collection. Players interact with the Stack Broker through three primary operations: peek(), pop(), and push(). The peek() operation allows players to inspect the document currently located at the top of the stack without removing it. The pop() operation removes the top document and exposes the next document beneath it. The push() operation adds a new document containing additional suspect information to the top of the stack. Through these interactions, players directly observe how stack structures process information according to the LIFO principle.

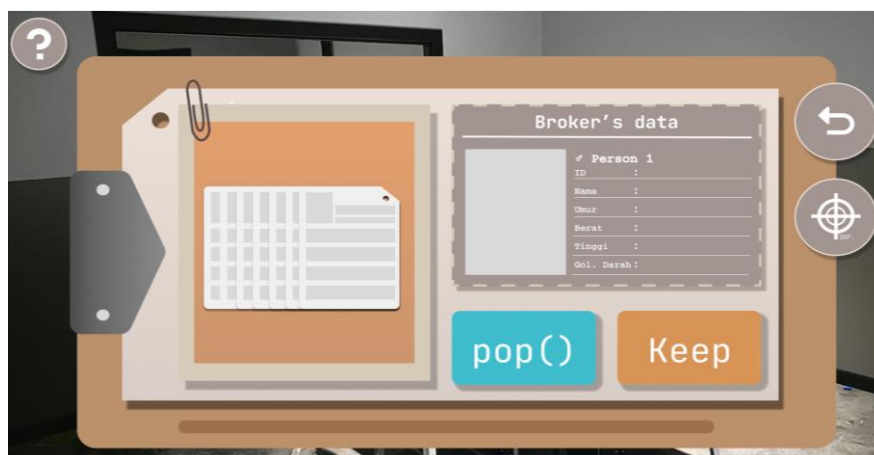


Figure 4. Stack Broker Interface

The implementation successfully transforms abstract stack operations into a visual investigation activity, enabling players to learn data structure behavior while collecting evidence related to criminal suspects.

3.2.2 Queue Broker Results

The Queue Broker was implemented as a witness-processing system that follows the First In First Out (FIFO) principle. Suspect information is obtained from a sequence of witnesses arranged in a queue, where the first witness entering the queue is always processed first. Players can inspect the witness at the front of the queue through the peek() operation before deciding whether to keep the information. After the information has been processed, the dequeue() operation removes the witness from the front of the queue. Additional witnesses may be added to the rear of the queue through the enqueue() operation, allowing the queue to continue processing information in sequential order.

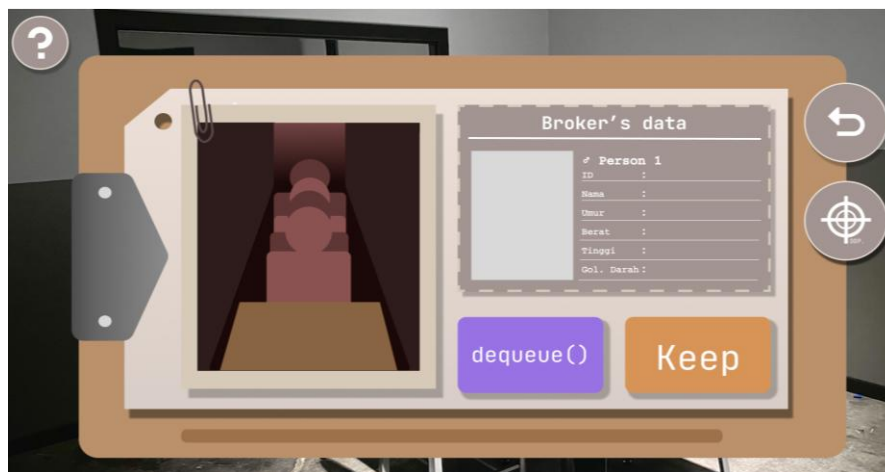


Figure 5. Queue Broker Interface

The broker visually displays front and rear positions within the queue, helping players understand how queue structures organize and process data. This implementation provides a practical demonstration of FIFO behavior within the context of an investigation workflow.

3.2.3 Hash Map Broker Results

The Hash Map Broker was implemented as a suspect database search system. Unlike the Stack Broker and Queue Broker, which require sequential access to information, the Hash Map Broker enables direct retrieval of suspect records using identification numbers. Players enter a suspect identification number obtained from previous clues into the search system. The identification number acts as a key, while the corresponding suspect profile stored in the database represents the value. Hash Maps are widely used for efficient data storage and retrieval, allowing information to be accessed directly through a key-value mapping rather than sequential searching [22].



Figure 6. Hash Map Broker Interface

The implementation demonstrates how hash maps utilize key-value relationships to support efficient data retrieval. By associating identification numbers with suspect records, players experience direct access to information without traversing multiple entries sequentially. The information collected from the Stack Broker, Queue Broker, and Hash Map Broker can be stored through the Keep mechanism and transferred to the Suspect List. This integration allows educational interactions involving different data structures to contribute directly to the investigation process.

3.3 Investigation Gameplay Results

The gameplay of Catch U Bro follows an investigation-oriented learning process in which players progressively gather evidence before identifying a criminal suspect. The investigation begins with a tutorial phase that introduces gameplay mechanics, learning objectives, and data structure concepts through narrative dialogues. After completing the tutorial, players interact with the available information brokers. Each broker provides suspect information through a different data structure implementation. During the investigation, players may choose to preserve important information using the Keep mechanism. Stored information is then transferred to the Suspect List, where collected evidence can be reviewed and compared. The investigation process continues until players determine that sufficient evidence has been gathered. At this stage, players proceed to the final investigation phase and select the suspect considered most likely to be responsible for the crime. The selected suspect is then verified against the predefined solution stored within the game.

The gameplay produces two possible outcomes. If the selected suspect matches the correct criminal record, the player receives a success ending. Otherwise, a failure ending is displayed. This investigation workflow integrates educational content with gameplay progression by requiring players to repeatedly apply data structure operations while collecting and evaluating evidence.

3.4 Educational Contribution

The discussion in this section reflects the design rationale underlying Catch U Bro, informed by findings from prior game-based learning literature. As the evaluation in this study was limited to functional testing (Section 3.5), the following points describe the intended educational mechanisms of the system rather than empirically confirmed learning outcomes for this specific implementation.

Catch U Bro demonstrates how fundamental data structure concepts can be integrated into an interactive investigation-based learning environment [23], [24]. Rather than presenting Stack, Queue, and Hash Map concepts through conventional lectures or programming exercises, the game introduces these structures as gameplay mechanics that directly support the investigation process [23], [24].

Each broker represents a different data structure and requires players to interact with its operations while collecting suspect information. The Stack Broker introduces Last In First Out (LIFO) behavior through document archive management, the Queue Broker demonstrates First In First Out (FIFO) processing through witness information handling, and the Hash Map Broker provides direct information retrieval using identification numbers as search keys. This design is intended to help players learn data structure behavior through practical interaction rather than memorizing theoretical definitions alone [25], [26].

The integration of data structure operations into gameplay also supports contextual learning [25], [27]. Information obtained from the brokers contributes directly to the investigation objective, allowing educational activities to remain connected to meaningful gameplay tasks [25], [28]. Players repeatedly apply data structure concepts while searching for evidence, organizing suspect information, and making investigative decisions.

In addition, the narrative-driven tutorial system helps reduce the complexity often associated with introductory data structure learning [26], [29]. Educational content is delivered gradually through dialogues and guided interactions before players perform the corresponding operations independently [25], [26]. This approach is intended to enable learners to understand both the purpose and practical usage of each data structure within a problem-solving context.

The implementation of Catch U Bro demonstrates the potential of educational game mechanics to transform abstract data structure concepts into interactive learning experiences [23], [28], [30]. By combining narrative guidance, investigation activities, and data structure operations within a single environment, Catch U Bro provides an alternative approach for introducing fundamental Data Structures and Algorithms concepts to beginner learners.

3.5 Functional Testing Results

Functional testing was conducted to verify that the main gameplay components of Catch U Bro operated according to their intended functionality. The testing process focused on validating dialogue delivery, data structure operations, suspect information management, and final investigation mechanisms. Each component was tested through predefined gameplay scenarios and evaluated based on whether the expected behavior was successfully executed.

Table 2. Functional Testing Results

Component	Test Scenario	Result
Dialogue System	Display tutorial dialogues and story progression	Pass
Stack Broker	Execute push(), pop(), and peek() operations	Pass
Queue Broker	Execute enqueue(), dequeue(), and peek() operations	Pass
Hash Map Broker	Search suspect information using identification number	Pass

Suspect List	Store and display selected suspect information	Pass
Data Management System	Load dialogue and suspect data from JSON files	Pass
Scene Navigation	Transition between game scenes correctly	Pass
Final Investigation	Verify selected suspect and display ending result	Pass

The testing results indicate that all major components operated successfully during gameplay. Dialogue content was displayed correctly, broker systems executed their respective data structure operations as expected, and suspect information was properly stored and retrieved throughout the investigation process. In addition, scene transitions and final suspect verification functions performed without errors. These results confirm that the implemented systems support the intended educational and gameplay objectives of Catch U Bro.

4. CONCLUSION AND RECOMMENDATIONS

This study presented Catch U Bro, an educational detective game designed to introduce Data Structures and Algorithms (DSA) concepts through interactive investigation-based gameplay. The game integrates multiple data structures, including Stack, Queue, and Hash Map, into detective-themed information broker systems that require players to interact with data according to the behavior of each structure. The system was successfully developed using the Godot Engine with a modular architecture consisting of the Dialogue System, Broker System, Suspect Management System, and Data Management System.

The implementation results demonstrate that data structure operations can be effectively transformed into meaningful gameplay mechanics. The Stack Broker successfully represents Last In First Out (LIFO) behavior through document archive management, the Queue Broker demonstrates First In First Out (FIFO) processing through witness and report handling, and the Hash Map Broker enables direct information retrieval using identification numbers as keys. These mechanics are designed to allow players to learn data structure concepts through practical interaction rather than relying solely on theoretical explanations, though this learning effect has not yet been empirically measured. The investigation workflow, supported by narrative tutorials and evidence collection activities, also provides a contextual learning environment that encourages logical thinking and problem-solving skills.

Overall, the implementation demonstrates that the objectives stated in the introduction were achieved. Catch U Bro successfully integrates multiple data structures into a single educational game environment while providing an engaging alternative learning medium for introductory DSA education. The game demonstrates the potential of game-based learning approaches to improve learners' understanding of abstract computer science concepts through interactive and contextualized experiences.

For future work, a priority is to conduct controlled experimental evaluations involving students, using instruments such as pre-test/post-test assessments and standardized usability questionnaires (e.g., SUS or UEQ), to empirically measure learning effectiveness, engagement, and knowledge retention compared with conventional teaching methods. In addition, the scope of the game could be expanded to incorporate additional data structures such as trees, graphs, linked lists, and heaps, and adaptive learning features or performance analytics could

be implemented to support personalized learning experiences. Such developments would strengthen the educational value of Catch U Bro and support its application as a complementary learning tool in computer science education.

REFERENCES

- [1] A. Tilanterä, "Students' topic-specific difficulties in learning data structures and algorithms," Aalto University, Helsinki, Finland, 2026. [Online]. Available: <https://urn.fi/URN:ISBN:978-952-64-2916-8>
- [2] H. Li, Y. Leng, and J. Peng, "Research On Teaching, Management And Learning Difficulties And Solutions In 'Data Structure And Algorithm' Of Applied Undergraduate," In 2021 International Conference On Advanced Education And Information Management (*AEIM 2021*), Destech Publications, Inc., Mar. 2021. doi: 10.12783/dtssehs/aeim2021/35961.
- [3] S. Su, E. Zhang, P. Denny, and N. Giacaman, "A Game-Based Approach for Teaching Algorithms and Data Structures using Visualizations," in *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*, Virtual Event USA: ACM, Mar. 2021, pp. 1128–1134. doi: 10.1145/3408877.3432520.
- [4] A. J. Rojas-Salazar, "Game-Based Learning of Data Structures Based on Analogies: Learning Gains and Intrinsic Motivation in Higher Education Environments." Trinity College Dublin, Dublin, Ireland, Mar. 2022.
- [5] C. Rademan, "Literature Review of Game-Based Learning in Computer Science Education," in *SAICSIT '21: Proceedings of the South African Institute of Computer Scientists and Information Technologists 2021*, Cape Town, Western Cape, South Africa: Association for Computing Machinery (ACM), 2021, pp. 1–5. doi: 10.1145/3476267.3488545.
- [6] F. C. Bedreski, I. W. Soares, and L. T. W. Agner, "Delimiter: A Game For Stack Data Structure Teaching," *RETEC - Rev. Tecnol.*, vol. 18, no. 2, pp. 65–76, Jan. 2025.
- [7] M. Grizioti and M.-S. Nikolaou, "Playing, Moving and Designing with Data: Exploring Young Students' Data Literacy Skills in Embodied Classification Games," in *Proceedings of the 3rd International Conference of the ACM Greek SIGCHI Chapter*, Syros Greece: ACM, Sep. 2025, pp. 197–202. doi: 10.1145/3749012.3749080.
- [8] D. Zhao, C. H. Muntean, A. E. Chis, G. Rozinaj, and G.-M. Muntean, "Game-Based Learning: Enhancing Student Experience, Knowledge Gain, and Usability in Higher Education Programming Courses," *IEEE Trans. Educ.*, vol. 65, no. 4, pp. 502–513, Nov. 2022, doi: 10.1109/TE.2021.3136914.
- [9] O. Barzilai, S. Sherman, M. Leiba, and H. Spiegel, "Cognitive Aspects in Problem Solving: The Case of a Data Structures Course for IS Students," *J. Inf. Syst. Educ.*, vol. 35, no. 2, pp. 175–188, 2024, doi: 10.62273/JJUB4136.
- [10] S. C. Gurbuz and M. Celik, "Serious games in future skills development: A systematic review of the design approaches," *Comput. Appl. Eng. Educ.*, vol. 30, no. 5, pp. 1591–1612, Sep. 2022, doi: 10.1002/cae.22557.
- [11] Y. Gui, Z. Cai, Y. Yang, L. Kong, X. Fan, and R. H. Tai, "Effectiveness of digital educational game and game design in STEM learning: a meta-analytic review," *Int. J. STEM Educ.*, vol. 10, no. 1, p. 36, May 2023, doi: 10.1186/s40594-023-00424-9.

- [12] F. S. Breien and B. Wasson, "Narrative categorization in digital game-based learning: Engagement, motivation & learning," *Br. J. Educ. Technol.*, vol. 52, no. 1, pp. 91–111, Jan. 2021, doi: 10.1111/bjet.13004.
- [13] E. Jääskä, J. Lehtinen, J. Kujala, and O. Kauppila, "Game-based learning and students' motivation in project management education," *Proj. Leadersh. Soc.*, vol. 3, p. 100055, Dec. 2022, doi: 10.1016/j.plas.2022.100055.
- [14] S. Cao and F. Liu, "Learning to play: understanding in-game tutorials with a pilot study on implicit tutorials," *Heliyon*, vol. 8, no. 11, p. e11482, Nov. 2022, doi: 10.1016/j.heliyon.2022.e11482.
- [15] D. Kao, A. J. Magana, and C. Mousas, "Evaluating Tutorial-Based Instructions for Controllers in Virtual Reality Games," *Proc. ACM Hum.-Comput. Interact.*, vol. 5, no. CHI PLAY, pp. 1–28, Oct. 2021, doi: 10.1145/3474661.
- [16] I. Bende, "Data Visualization in Programming Education," *Acta Didact. Napoc.*, vol. 15, no. 1, pp. 52–60, Aug. 2022, doi: 10.24193/adn.15.1.5.
- [17] I. A. Pratiwi and A. Setiawan, "STANQUE: Android-based stack and queue learning media for vocational students," presented at the 5Th International Conference on Innovative Design, Analysis & Development Practices in Aerospace & Automotive Engineering: I-DAD'22, Chennai, India, 2023, p. 020094. doi: 10.1063/5.0141569.
- [18] N. Pereira, A. Rowe, M. W. Farb, I. Liang, E. Lu, and E. Riebling, "ARENA: The Augmented Reality Edge Networking Architecture," in *2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, Bari, Italy: IEEE, Oct. 2021, pp. 479–488. doi: 10.1109/ISMAR52148.2021.00065.
- [19] V. Kumaran, J. Rowe, B. Mott, and J. Lester, "SceneCraft: Automating Interactive Narrative Scene Generation in Digital Games with Large Language Models," *Proc. AAAI Conf. Artif. Intell. Interact. Digit. Entertain.*, vol. 19, no. 1, pp. 86–96, Oct. 2023, doi: 10.1609/aiide.v19i1.27504.
- [20] R. Horst, D. Fenchel, R. Retz, L. Rau, W. Retz, and R. Dörner, "Integration of game engine based mobile augmented reality into a learning management system for online continuing medical education," in *Proc. INFORMATIK 2020*, R. H. Reussner, A. Koziolk, and R. Heinrich, Eds. Bonn, Germany: Gesellschaft für Informatik, 2021, pp. 955–962, doi: 10.18420/inf2020_88.
- [21] E. C. Foster and B. Towle Jr., *Software engineering: a methodical approach*, 2nd ed. Boca Raton: CRC Press, 2021.
- [22] V. A. Kolesnikov, "Visualization of Hash Map Data Structure," *Comput. Sci. Appl. Math.*, no. 1, pp. 66–74, 2024, doi: 10.26661/2786-6254-2024-1-08.
- [23] M. Videnovik, T. Vold, L. Kiøng, A. Madevska Bogdanova, and V. Trajkovik, "Game-based learning in computer science education: a scoping literature review," *Int. J. STEM Educ.*, vol. 10, no. 1, p. 54, Sep. 2023, doi: 10.1186/s40594-023-00447-2.
- [24] Z. Zhan, L. He, Y. Tong, X. Liang, S. Guo, and X. Lan, "The effectiveness of gamification in programming education: Evidence from a meta-analysis," *Comput. Educ. Artif. Intell.*, vol. 3, p. 100096, 2022, doi: 10.1016/j.caeai.2022.100096.

- [25] V. Shute *et al.*, "Maximizing learning without sacrificing the fun: Stealth assessment, adaptivity and learning supports in educational games," *J. Comput. Assist. Learn.*, vol. 37, no. 1, pp. 127–141, Feb. 2021, doi: 10.1111/jcal.12473.
- [26] M. S. Alotaibi, "Game-based learning in early childhood education: a systematic review and meta-analysis," *Front. Psychol.*, vol. 15, p. 1307881, Apr. 2024, doi: 10.3389/fpsyg.2024.1307881.
- [27] J. Xie, M. Wang, and D. Hooshyar, "Student, parent, and teacher perceptions towards digital educational games: How they differ and influence each other," *Knowl. Manag. E-Learn. Int. J.*, vol. 13, pp. 142–160, Jun. 2021, doi: 10.34105/j.kmel.2021.13.008.
- [28] M. Li, S. Ma, and Y. Shi, "Examining the effectiveness of gamification as a tool promoting teaching and learning in educational settings: a meta-analysis," *Front. Psychol.*, vol. 14, p. 1253549, Oct. 2023, doi: 10.3389/fpsyg.2023.1253549.
- [29] M. Xu, Y. Luo, Y. Zhang, R. Xia, H. Qian, and X. Zou, "Game-based learning in medical education," *Front. Public Health*, vol. 11, p. 1113682, Mar. 2023, doi: 10.3389/fpubh.2023.1113682.
- [30] S. Y. Cheung and K. Y. Ng, "Application of the Educational Game to Enhance Student Learning," *Front. Educ.*, vol. 6, p. 623793, Mar. 2021, doi: 10.3389/feduc.2021.623793.