Funded by
the European Union

Beat the bomb - Report

Module 5

Team members:

Leader: Dumitrescu Bianca Alexandra – 313AC

Joița Fabian Gabriel – 313AC

Jugulete George Marius Alexandru – 313AC

Mănescu Daria Ioana – 313AC

Sandu Andreea Diana – 313AC

Coordonating teachers: Mihail Caramihai
Daniel Chis

Funded by the European Union

I. Project objectives

Through the game "Beat the Bomb," we, the team members, ensured a high level of engagement and involvement. This is reflected in the captivating experience the game offers, incorporating interactive elements and time-based rewards designed to motivate players to continue and stay engaged. With this approach, we aim to encourage not only fun but also learning and the discovery of new aspects of biodiversity, contributing to the cultural enrichment of the players.

Technical objectives:

1. Design and implementation of a gamification solution (Beat the Bomb) :

Defining and developing a captivating and educational game experience for users, using specific gamification elements within the game "Beat the Bomb."

2. Using a solution with graphs:

The integration and efficient use of graph data structures within the solution facilitates smooth and fast management of connections between game elements

3. Application development:

Creating and actually implementing the 'Beat the Bomb' app, ensuring it works and provides a pleasant experience for users.

Funded by the European Union

The objectives of the game:

1. <u>Educating the public about the importance of biodiversity:</u>

Creating an interactive and fun environment for players, giving them the opportunity to learn about the environment and biodiversity through 15 statements divided into 4 categories.

2. <u>Strengthening knowledge about biodiversity:</u>

Improving players' knowledge and understanding of biodiversity and ecosystems by providing accurate and detailed information on various aspects of these areas.

3. <u>Encouraging public engagement and participation:</u>

Using the game "Beat de Bomb" to encourage active participation of players in actions to protect the environment, through a variety of statements to challenge them to act in favor of biodiversity conservation.

4. <u>Evaluating the impact on knowledge and attitudes:</u>

Assessing the game's impact on players' knowledge, attitudes and behavior in relation to biodiversity and the environment through feedback.

II. Documentation ("state of the art")

In the initial stage of documentation, a deep understanding of the principles behind the Beat the Bomb game was essential. This understanding led to the conclusion that the app must provide an interactive experience to attract users and educate them in an engaging way. Thus, the final goal of the project was defined as the creation of an application that provides not only entertainment, but also the opportunity to learn and test users' knowledge in an interactive and interesting way.

To achieve these objectives, a number of main development directions have been synthesized, ensuring that each stage of the development process contributes to their achievement. This strategic and planned approach is crucial to ensure the long-term success and effectiveness of the project.

Choosing the C programming language for game development was a deliberate decision. This language offers a balance between performance and control, making it ideal for game development that requires a fast and responsive interface. In addition, familiarity with this language can facilitate further development and maintenance of the application.

Through the game, users are challenged to quickly decide whether statements about biodiversity are true or false, thus having the opportunity to test their own knowledge against time. This approach not only stimulates decision-making ability and quick reactions, but also encourages hands-on learning and tests users' cognitive skills, encouraging them to test and reinforce their knowledge in an interactive and fun way.

Funded by
the European Union

The statements that can be found in the application were created based on Module 5 on Biodiversity, a topic that belongs to the "Gender, Digitalization, Green: Ensuring a Sustainable Future for all in Europe" project. There are 15 of these statements, divided into 4 relevant and suggestive categories. This theme not only adds relevance and timeliness to the game, but also helps promote a deeper awareness of the importance of biodiversity and sustainability.

The development process of the application was based on the implementation of a gamification solution. This approach was chosen to increase user engagement by giving them rewards and goals to motivate them to keep playing and learning.

The use of graphs in the code made in the C programming language and the addition of the raylib library were technical choices. Graphics are a crucial element of the game experience and using a library like raylib, a standard library for creating video games through programming, can simplify and speed up the development process, while providing a wide range of functionality and practical examples to facilitate the process of learning and implementation.

In conclusion, every aspect of the development of the game "Beat the Bomb" was planned and executed with care and attention to ensure an engaging, educational and interactive experience for users. From the selection of the programming language to the choice of content and the implementation of technologies, every decision was oriented towards achieving the set objectives and satisfying the needs and expectations of the users.

Funded by
the European Union

III. Justification of the solution

1. *Part of the backend logic or game engine:*

Code Description:

This implementation provides functionalities for working with graphs in the C language, considering their use within the game "Beat the Bomb." The game is based on gamification and involves navigating through four of knowledge categories, each containing four distinct questions. To efficiently organize and manage these questions, we have adopted the concept of graphs.

Basic Data Structure:

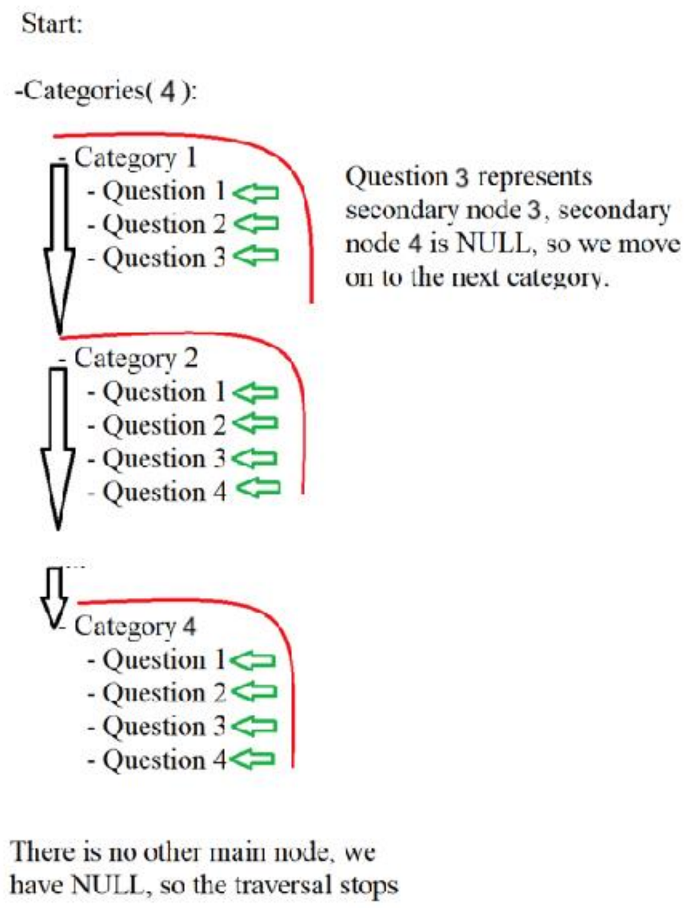
To implement a "content -> category (4) -> 4 questions" structure, we created a main graph containing 4 primary nodes. Each primary node is associated with a secondary graph, which in turn contains four secondary nodes representing the respective questions.

Graph Characteristics:

The graphs are directed, as the traversal of questions is done in a straightforward manner without randomization. This direction ensures that the user always receives an expected output and that the navigation through categories and questions is executed in an orderly and coherent manner.

The benefits of using graphs:

The use of graphs provides an efficient and elegant structure for organizing and managing questions within the "Beat the Bomb" game. Graphs allow for quick and easy access to categories and questions, and their orientation ensures that navigating the game unfolds in a logical and fluent manner. By integrating this implementation of graphs into the game's code, we ensure that the user experience is enjoyable and engaging, providing them with the opportunity to explore and learn in an interactive and stimulating manner.



2. *Graphical User Interface (GUI):*

Funded by the European Union

Regarding the implementation of the graphical interface in C, there are various ways to do this, each with its own advantages and disadvantages. Some of these methods include:

- Using the SDL (Simple DirectMedia Layer) library: SDL is a low-level development library that can be used to create simple graphical interfaces in C. Although initially intended for game development, SDL provides functionality for managing windows, user events, and drawing graphic elements on the screen.

- Using OpenGL or Vulkan: If you want to create a more advanced graphical interface or have full control over the drawing process on the screen, you can use OpenGL or Vulkan to work with 3D and 2D graphics in C. These libraries provide access to low-level graphics functionality, allowing you to draw directly to the screen's display buffer.

- Using the Raylib library: Raylib is a simple and easy-to-use library for developing 2D applications and games in the C language. It provides functionality for creating windows, drawing shapes and text, managing user input, and much more.

- Using a scripting language or a Rapid Application Development (RAD) environment: You can use a scripting language such as Lua or Python along with a graphical interface library like Tkinter or PyQt to swiftly create graphical interfaces. This allows you to leverage the power and flexibility of scripting languages while benefiting from pre-made graphical components.

Funded by the European Union

Among the methods of constructing a graphical interface presented above, the following conclusion has been reached: Raylib represents one of the best libraries in C for creating a UI according to the requirements of our application.

The choice of Raylib over other graphical interface development technologies has been motivated by several important factors:

1. **Ease of Use and Accessibility**: Raylib provides a simple and intuitive programming interface, making it easy to develop and implement graphical interface elements, even for those with limited experience in computer graphics. This feature reduces the time required to learn and use the library and allows developers to focus more on the application's functionality.

2. **Optimized Performance and Efficiency**: Raylib is known for its performance and efficiency in handling graphics, ensuring a smooth and responsive experience for the user. This aspect is particularly important in a game or interactive application, where the responsiveness of the graphical interface is essential for a pleasant and satisfying experience.

3. **Comprehensive Documentation and Active Community**: Raylib comes with detailed and well-structured documentation, providing extensive information about the library's usage and functionalities. Additionally, we benefit from an active community of users and developers who can provide support and solutions to issues encountered during development.

Funded by the European Union

4. **Flexibility and Versatility**: Raylib offers a wide range of functionalities for drawing and managing graphic elements, as well as interacting with the user. This versatility allows us to implement a variety of graphical interface elements in a flexible and efficient manner, depending on the specific needs and requirements of the application.

5. **Multi-Platform Compatibility and Free of Charge**: Raylib is an open-source and free library that is compatible with multiple platforms, including Windows, Linux, and macOS. This compatibility allows us to develop graphical interface applications using the same code base for different operating systems, reducing development efforts and facilitating the distribution of the application across various platforms.

In conclusion, integrating the Raylib library into this project was a strategic and rational choice, enabling the development of a high-quality and superior-performing graphical.

IV. Description of Implementation

Data Structures

`struct Edge`

- Represents an edge between two nodes in a graph.
- Members:
    * `int src`: The source node of the edge.
    * `int dest`: The destination node of the edge.

`struct node`

- Represents a node in the main graph.

- Members:

  * `int dest`: The destination node of the edge.

  * `char continut_nod_graf_principal [MAX_LINE_LENGTH]`: The content of the node, representing the associated knowledge category.

  * `struct node* next`: Pointer to the next node in the list.

  * `struct node_secund* next_secundar``: Pointer to the secondary nodes associated with this main node.

`struct node_secund`

- Represents a node in the secondary graph associated with a main node.

- Members:

  * `int dest`: The destination node of the edge.

  * `char answer[3]`: The correct answer to the question associated with the secondary node.

  * `char char continut_nod_graf_secundar [MAX_LINE_LENGTH]`: The content of the node, representing the associated question.

  * `struct node_secund* next`: Pointer to the next secondary node in the list.

`struct Graph`

- Represents a main graph.

- Members:

  * `struct node* head[MAX_VERTICES_GRAPH]`: Array of pointers to the heads of adjacency lists of nodes.

`struct Graph_secund`

- Represents a secondary graph associated with a main node.

- Members:

 * `struct node_secund* head[MAX_VERTICES_SUBGRAPH]`: Array of pointers to the heads of adjacency lists of secondary nodes.

Functions

- `struct Graph* createGraph(struct Edge edges[], int num_edges, int num_vertices)`

- **Description:** Creates a main graph using an array of edges and a given number of vertices.

- **Parameters:**

 1. `edges[]`: Array of edges.
 2. `num_edges`: The total number of edges.
 3. `num_vertices`: The total number of vertices.

- **Returns:** A pointer to the created main graph.

- `struct Graph_secund* createGraphSecund(struct Edge edges[], int num_edges, int num_vertices, struct node* node1, struct Graph* graph_principal, int nr_intrebari_utilizate)`

- **Description:** Creates a secondary graph associated with a main node.

- **Parameters:**

 1. `edges[]`: Array of edges.
 2. `num_edges`: The total number of edges.
 3. `num_vertices`: The total number of vertices.
 4. `node1`: Pointer to the main node associated with the secondary graph.
 5. `main_graph`: Pointer to the main graph.

6. `used_question_count`: The total number of questions used.

- **Returns:** A pointer to the created secondary graph.

- `void printGraph(struct Graph* graph, int num_vertices)`

  - **Description:** Displays the main graph and the secondary graph associated with each main node.

  - **Parameters:**

  1. `graph`: Pointer to the main graph.
  2. `num_vertices`: The total number of nodes in the graph.

This is an overview of the functionalities and data structures in the `graph.c` code.

<u>The description of the Raylib library implementation includes the following aspects:</u>

1. Initialization and Configuration of the Graphic Interface: To create and manage the graphical interface, functions provided by the Raylib library are used. Within the code, this process begins with initializing the window using the InitWindow() function. This function receives parameters that define the dimensions of the window and its title. For example, InitWindow(screenWidth, screenHeight, "Window Name") initializes the window with the specified dimensions and title.

2. Drawing Graphic Elements: Drawing graphic elements such as buttons and loading bars is done using functions like DrawRectangleRec() and DrawText(). These functions allow drawing geometric shapes and text on the screen in a flexible and efficient manner. Additionally, images can be added using functions like LoadImage() and

Funded by the European Union

LoadTextureFromImage() to load the image from a file and transform it into a texture, and DrawTexture() to draw it on the screen.

3. Handling User Input: To enable user interaction with the graphical interface, functions for detecting input events such as key presses and mouse clicks are used. For example, using functions like IsKeyPressed() and IsMouseButtonPressed(), it detects when the user presses answer buttons (True and False) or other interactive elements in the interface.

4. Time Management: To measure and manage time within the application, functions for obtaining the current time and calculating time differences between events are used. This is useful in the context of games and interactive applications for implementing functionalities such as loading bars and game timers.

5. Sounds and Music: The Raylib library also provides functionalities for managing sounds and music in applications. Sounds can be loaded and played using functions like LoadSound() and PlaySound(), while music can be loaded and played using functions like LoadMusicStream() and PlayMusicStream().

6. Finalization and Resource Cleanup: Upon application termination or window closure, operations to clean up and release resources used by the graphical interface are performed. This includes calling the CloseWindow() function to close the window, releasing textures using UnloadTexture(), releasing sounds using UnloadSound(), releasing music using UnloadMusicStream(), and releasing other resources used in the drawing and interaction process.

7. In order to assure portability on all devices the Raylib’s Notepad++ application was used to create the executable file attached to the Application zip file.

Therefore, the implementation of the graphical interface within the application relies on the efficient use of functions and methods provided by the Raylib library, allowing the developer to create attractive and functional interactive visual interfaces in the C language.

V. Functionality

Interface and navigation:

When the application is opened, the user is greeted by a clear and concise message, "Press SPACE to start", which invites the initiation of the gaming experience. The first window also displays credentials and a welcome to the project „Beat The Bomb”. The simple and intuitive interface makes it easy to navigate and quickly access the game.

Presentation of statements and answers:

Statements about biodiversity are presented sequentially, and users are challenged to choose between two answer options: TRUE or FALSE. Options are visually represented by colored boxes, providing a clear and efficient interaction with the game. The statements are also grouped by four into a specific category, and this is present next to each individual utterance. Through this structure, users can better understand the context of each statement and identify the relationship between questions within the same category, thus facilitating the process of navigation and understanding of the game's theme.

User feedback:

Funded by the European Union

The game provides immediate and visible feedback after each response given by the user. The messages "Correct!" or "Incorrect!" are displayed above, using distinct colors, green for correct answers and red for incorrect ones, emphasizing the clarity and consistency of the feedback.

Time Management:

A crucial element of the game is the visible timer, which puts pressure on the user and encourages quick and efficient responses. The timer goes on for wrong answers, but for correct answers the timer goes back by a few seconds, gaining more time.

Progress Monitoring:

The user can monitor their progress and performance in real time by seeing the number of correct and incorrect answers displayed in the upper left corner of the screen. This functionality allows for continuous self-assessment and provides opportunities for adjusting game strategy.

Consequences of Losing:

Implementing an automatic game loss condition in case of exceeding the time limit or answering incorrectly multiple times adds finality, encouraging users to improve their speed and accuracy in their answers.

The game also includes sound effects. When the user answers a question, an explosion sound is played. When the user runs out of time, a bomb explosion sound is played.The game is designed to create a sense of urgency and pressure. The loading bar depletes quickly, and the user must answer questions as quickly as possible to recover time. The bomb explosion sound adds to the pressure, reminding the user that time is running out. The game is intended to be challenging and exciting, providing the user with a sense of accomplishment when they win.

Other "Beat the Bomb" Game Solutions on the Market:

In the gaming market, there is a wide range of solutions, each with its own principles and distinct characteristics. Each type of game is differentiated by unique gameplay mechanics, themes, and levels of interactivity, yet all contribute positively by stimulating cognition, developing social skills, and providing a fun way to relax and entertain. In the "Beat the Bomb" game, the emphasis is on quickly combining knowledge and skills to solve challenges or puzzles within a limited time.

A similar example is the game "Trivia Crack," which focuses on testing players' general knowledge, covering a wide range of fields such as history, science, general culture, and entertainment. Players are challenged to quickly and correctly answer questions to earn points and progress in the game. A distinctive feature of this game is its presentation of questions in the form of a spinning mechanism, adding an element of anticipation and competitiveness.

On the other hand, "QuizUp" offers a social and competitive gaming experience, allowing players to compete directly against each other in real-time. Players can choose from a variety of themes and topics and test their knowledge against other players from around the world. A unique feature of this game is its ranking and scoring system, which encourages competition and rivalry.

The "Beat the Bomb" game brings multiple benefits to users through its interactive and challenging experience. Firstly, it promotes the development of cognitive skills such

Funded by
the European Union

as quick thinking, problem-solving, and decision-making within a limited time. These skills are fundamental in many aspects of life, and practicing them in a fun and engaging environment can strengthen understanding and effectiveness. Additionally, the game stimulates teamwork and communication as players must work together to solve challenges and puzzles. This social aspect not only strengthens interpersonal relationships but also encourages the development of teamwork and leadership skills. Furthermore, the game provides a fun and relaxing experience, helping to reduce stress and improve users' mood. Through all these benefits, "Beat the Bomb" is not just a form of entertainment but also a valuable tool for the personal and social development of players.

How can we improve players' experience?

1. Multiple Difficulty Modes: Adding multiple difficulty options could allow players to choose the appropriate level of challenge for them. For example, an "easy" mode could offer a slower pace and fewer obstacles, while an "advanced" mode could ramp up the pace and introduce more challenges.

2. Progression and Rewards System: Implementing a progression and rewards system could motivate players to keep coming back and playing. For example, players could earn points or rewards for completing each level or task, which could be used to unlock additional content or customize the gaming experience.

3. Multiplayer and Cooperative Modes: Introducing multiplayer and cooperative modes could bring a new level of interaction and competition to the game. For example, players could collaborate to solve puzzles or achieve common objectives, or they could compete against each other in different challenges and mini-games.

4. Level Editor and User-Generated Content: Allowing players to create and share their own levels and content could significantly extend the game's lifespan and create an active community around it. For example, an intuitive level editor would enable players to create and share their own challenges and puzzles, encouraging creativity and innovation.

5. Alternative Game Modes: Introducing alternative game modes, such as Endless or Time Attack modes, could offer even more variety and fun for players. These modes could be tailored to provide different experiences and to challenge players' skills and strategies in new and interesting ways.

6. Support for Controllers and Alternative Interfaces: Expanding support for different types of controllers and interfaces could enhance the game's accessibility and provide players with more options to experience and interact with the game. For example, support for motion controllers or virtual reality devices could offer more interactive and immersive gaming experiences.

VI. Conclusion

Developing Beat the Bomb was a complex and immersive experience where we were able to combine technical skills and creativity to provide users with an interactive and educational experience. The collective effort of the team was essential in turning a simple idea into an outstanding final product.

During the development process, we encountered various technical and conceptual challenges, but we managed to stay focused on our main goals: creating an engaging game and providing a valuable educational experience. The choice of the C

Funded by
the European Union

programming language and using graphs were key decisions that helped implement solid and efficient logic within the game.

The raylib library was a key partner in the development of the game's graphical aspects, providing us with the tools and resources needed to create an attractive and dynamic visual interface. The integration of a timer and biodiversity statements added complexity to the gaming experience, thus turning the app into a learning tool in addition to its entertainment function.

In conclusion, the game "Beat the Bomb" is not only an entertainment application, but also an education platform in the field of biodiversity and environmental conservation. Through this application, we want to inspire and encourage users to explore and understand the importance of protecting nature and biodiversity for a sustainable future.