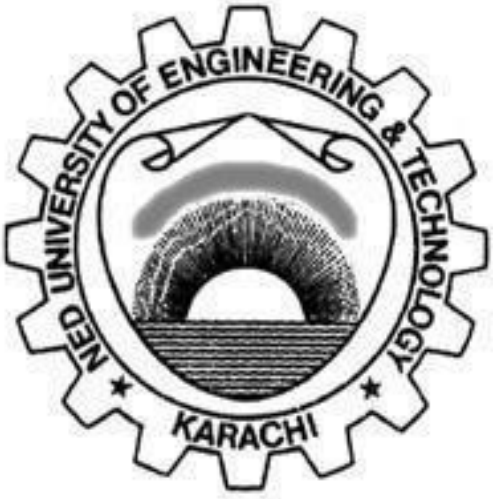
SHADOW IN THE MANSION



GROUP MEMBERS:

1. MISL E NOOR OVEIS (CT-073)

2. HAIQA ANIS KHAN (CT-060)

3. AHMAD RAZA (CT-086)

4. M KAZIM ABBAS LASHARI (CT-095)

Table of Content:

1. INTRODUCTION----------PAGE 3
2. MOTIVATION----------PAGE 4
3. PROJECT'S DESCRIPTION----------PAGE 5
4. OOP CONCEPT ----------PAGE 7
5. FEATURES ----------PAGE 8
6. UML CLASS DIAGRAM ----------PAGE 11
7. FLOAT DIAGRAM ----------PAGE 13
8. OUTLINE ----------PAGE 14
9. NEW CONCEPTS ----------PAGE 15
10. LIBRARIES ----------PAGE 17
11. CONTRIBUTIONS OF EACH MEMBER----------PAGE 18
12. CONCLUSION----------PAGE 19

INTRODUCTION:

The Ravenswood Detective Mystery Game welcomes you.

Fear takes hold of the people in the sleepy village of Ravenswood as a deadly mystery comes to light. A flood of unexplained killings has disrupted the calm and left the community feeling suspicious and uneasy.

It is your responsibility as the town's seasoned investigator to reveal the truth that is concealed in the shadows. Equipped with determination, you have to work your way through a maze of hints, solve complex puzzles, and face a cast of suspects, many of whom have secrets of their own to conceal.

As you gather proof from secret research, creepy studios, and the town hall's power corridors, the time is ticking mercilessly. Every choice you make whilst questioning suspects, solving puzzles, and finally leveling the crucial charge matters.

But take caution—the killer may be hiding in plain sight, using duplicity to get away with his crimes. Can you figure out the truth before one more person is hurt? Ravenswood's destiny is in your hands; your clever judgment and unflinching commitment will determine its outcome.

Detective, get ready. The secret to solving the Ravenwood enigma lies in the shadows.

MOTIVATION

The goal of creating this detective mystery game was to put players in the shoes of a detective and immerse them in an exciting and captivating story. The goal of the game is to test players logical reasoning and strategic thinking by combining elements of inquiry, problem solving, and decision making.

1. Entertaining Storyline: The action takes place in Ravenswood; a community where mysterious killings are a problem. Players must find clues, question suspects, and eventually identify the culprit as they become engrossed in the mystery and drama.

2. Interactive Gameplay: Players actively participate in solving the mystery with interactive gameplay elements including exploring areas, figuring out puzzles, and leveling important accusations. Their every move shapes the story, providing a dynamic and unique gameplay experience.

3. Puzzle and Riddle Challenges: Each chamber has a challenging puzzle or riddle that add s to the gameplay's depth and complexity. In order to solve each mystery, players must use thinking and logic to interpret clues, which improves their cognitive abilities and gives them a sense of success.

4. Character Development: The story is made complex by the varied group of suspects, each of whom has a unique motivation and set of alibi. Players can learn more about suspects' motivations and histories by questioning them; this information may help uncover the real offender.

5. Atmospheric Design: Room descriptions and visual depictions improve immersion and provide a tangible feeling of suspense and mystery. Every setting, from creepy studios to cunning study, is painstakingly designed to create curiosity and excitement.

6. Educational Value: By encouraging critical thinking, problem-solving, and decision-making abilities, the game provides educational advantages in addition to amusement. In order to advance in their investigation, players must carefully consider the facts and make rational decisions.

7. Exciting Conclusion: Based on their accumulated evidence, players must make an allegation in the game's explosive finish. When players correctly identify the culprit, the mystery is solved, rewarding them for their astute observation and deductive thinking.

We hope to deliver an engaging and immersive gaming experience with our detective mystery game that tests, amuses, and makes players hungry to solve the next mystery.

PROJECT DESCRIPTION

Overview: Set in the imaginary town of Ravenswood, the Ravenswood Detective Mystery Game is a text-based adventure game written in C++ that puts players in the shoes of a detective tasked with investigating a string of unexplained killings. Players move through a variety of rooms, speak with suspects, find clues, and eventually figure out who the culprit is.

Important Characteristics:

1. Interactive Gameplay: Players can select from a variety of options, including looking into rooms, questioning suspects, and leveling accusations. Every choice you make affects how the game plays out in the end.

2. Rich Storyline: The plot develops with captivating narrative components, such as thorough descriptions of the rooms, hints, suspects, and their motivations.

3. Solving Puzzles and Riddles: To find important information, players must solve the puzzles and riddles that are hidden throughout each room. These riddles provide information of the suspect. When riddles are correctly solved, hidden meanings behind the mystery become apparent.

4. A Range of Characters: The game has a range of suspects, each with their own distinct personalities, explanations for their actions, and possible goals. Players can remove suspects from their investigation and learn important information by questioning them.

5. Inventory and History Tracking: Through a thorough history record, players can examine their past acts and keep track of the clues they have gathered. This function makes gathering and organizing information easier.

6. Dynamic Endings: Depending on the choices and activities made by players during the inquiry, the game presents a number of possible

outcomes. If the killer is correctly identified, the story comes to a satisfying end, but if the charges are false, the suspect will escape resulting in more killings.

OOP CONCEPTS

In our game code, several Object-Oriented Programming (OOP) concepts are utilized effectively. Here's how each concept is applied:

1. Classes and Objects: Classes like Clue, Suspect, Room, and Game are defined to encapsulate related data and behaviors. Objects of these classes are instantiated throughout the program, such as TomHargrove Clues, TomHargrove, rooms, etc.

2. Encapsulation: Each class encapsulates related data (attributes) and methods (behaviors). For example, Clue class encapsulates attributes like description, location, analysis, riddle, and answer.

3. Inheritance: Inheritance is used with subclasses inheriting from base classes. For instance: BloodyRiddle, TinyKey, etc., inherit from Clue. TomHargrove, ElenaMarsh, JackSparrow, DrSamuelLowe inherit from Suspect. This promotes code reuse and allows specific behavior and attributes to be defined at different levels of specialization.

4. Polymorphism: Polymorphism allows the program to treat objects of different classes uniformly through inheritance and method overriding. For example, different types of clues (BloodyRiddle, PuzzlePiece) are handled uniformly through the base class Clue.

5. Abstraction: Abstraction is seen in the definition of classes and their methods, hiding complex implementation details behind well-defined interfaces. Methods like investigateRoom, interrogateSuspect, and accuseSuspect provide a high-level interface for interacting with rooms and suspects.

FEATURES

The project's key features, as per the code provided, are as follows:

A text-based detective mystery game, situated in the made-up town of Ravenswood, is the focus of this project.

Clues, Suspects, Rooms, and the game itself are all represented as classes in the object-oriented design of the game (Clue, Suspect, Room, Game).

- Interactive Modes:
  1. Investigation: In order to uncover information about the deaths, players can look through a variety of places, including Robert's Study and Tom Hargrove House.
  2. Interrogation: To learn more about suspects' alibis and motivations, players can question suspects (Tom Hargrove, Elena Marsh, etc.).
  3. Accusation: In order to identify the murderer, players are able to make allegations based on information obtained from clues and interrogations of suspects.

- Questioning Suspects: To learn more about their justifications and alibis, players can question suspects like Tom Hargrove and Elena Marsh. In an effort to identify the culprit, players are able to level

allegations against suspects based on information obtained during interrogations and gathered evidence.

- Clue System: In order to advance in the game, you must find clues. Each clue offers details that can be used to solve puzzles and find the culprit. Answers to riddles can be found in certain Clue subclasses, where players must solve the riddles by giving the right answers.

- Visual Representation: To improve the immersive experience of exploring various settings, each room includes a visual representation created with ASCII art (a visual attribute in the Room class).

- History and Inventory: The game tracks the player's actions (represented by a history vector) and stores gathered hints (represented by an inventory array).

- Dynamic gameplay: Based on the player's selections, the game will dynamically display alternatives to them, including exploring rooms, questioning suspects, looking through inventory, and leveling accusations.

- Storytelling: To improve player involvement and immersion, the project features an opening story that establishes the setting and narrative framework for the detective mystery game.

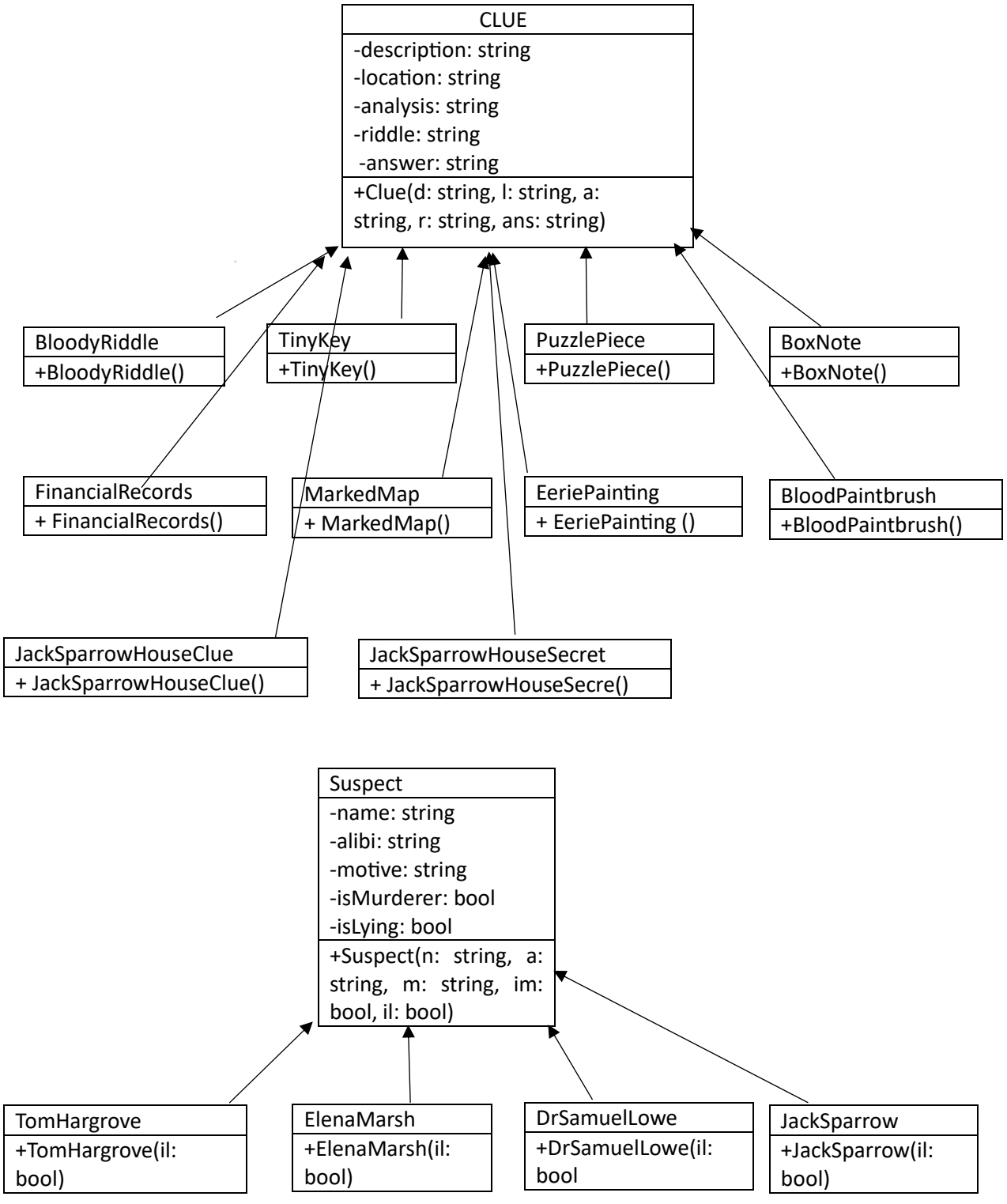
- Sound Effects: To improve the mood and gaming experience, use the sound effects (new.wav).

- Game Loop: The mystery must be solved by participants within a set number of attempts (MAX_ATTEMPTS) before the murderer makes their getaway.

- The parameter “–lwinmm” is used in Dev C++ (and similar environments) to link against the winmm library, which stands for "Windows Multimedia. “-lwinmm” is used to link your game project with the winmm library, enabling the use of multimedia functions such as PlaySound() to enhance the gaming experience with audio effects and background music.

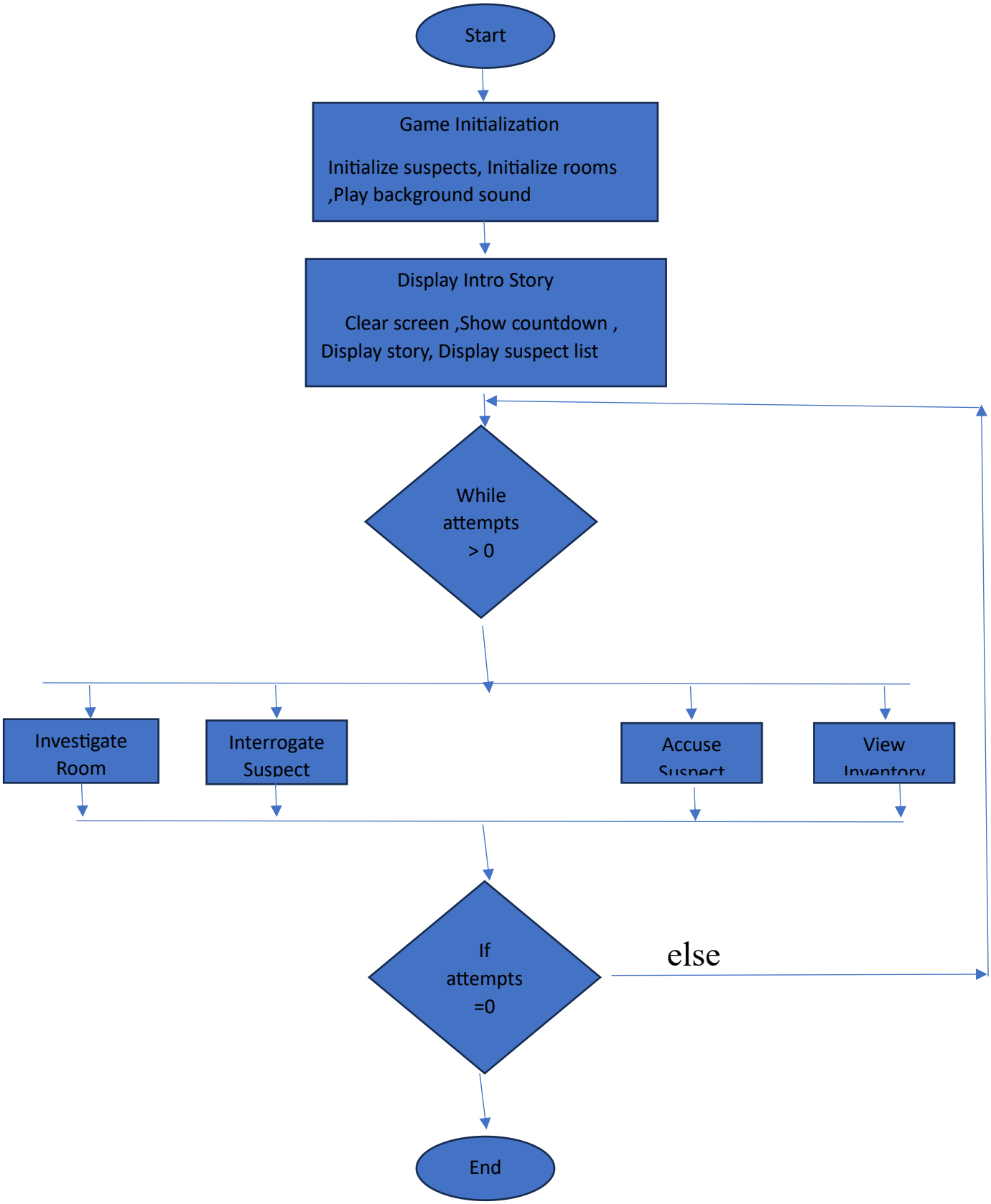
Together, these elements produce a captivating detective mystery game that tests players' ability to utilize deductive reasoning, acquire information, and make calculated choices in order to discover the truth about the killings occurring in Ravenswood.

UML CLASS DIAGRAM



Room
-name: string -clues: Clue[MAX_CLUES] -visual: string
+Room(n: string, c: Clue[], v: string) +displayRoomVisual(): void

Game
-suspects: Suspect[MAX_SUSPECTS] -rooms: Room[MAX_ROOMS] -inventory: Clue[MAX_ROOMS * MAX_CLUES] -inventorySize: int -attemptsRemaining: int -accusationAttempts: int -history: vector<string>
+Game() +start(const char* text): void -initializeSuspects(): void -initializeRooms(): void -displayCenteredText(const string& text): void -clearScreen(): void -displayTextWithDelay(const string& text, int delay): void -displayIntroductoryStory(): void -recordAction(const string& action): void -investigateRoom(int roomIndex): void -interrogateSuspect(int suspectIndex): void -accuseSuspect(): void -viewInventory(): void -displaySuspects(): void


Start
Game Initialization
Initialize suspects, Initialize rooms ,Play background sound
Display Intro Story
Clear screen ,Show countdown , Display story, Display suspect list
While attempts > 0
Investigate Room
Interrogate Suspect
Accuse Suspect
View Inventory
If attempts =0
else
End

OUTLINE

Introduction: The game begins with an intriguing introduction that sets the stage for the mystery. Players are introduced to the crime scene and the initial clues that hint at the complexity of the case.

Gameplay Mechanics

Room Exploration: Players navigate through various rooms within the crime scene, each with unique clues and puzzles. ASCII art is used to visually represent each room, enhancing immersion.

Clue Collection: Players collect clues by interacting with objects in the rooms. Clues include physical evidence, riddles, and cryptic messages that require solving.

Suspect Interrogation: Interactions with suspects are critical. Players interrogate suspects to uncover alibis, motives, and detect lies. Choices during interrogations influence the investigation's direction.

Inventory Management

Clue and Item Management: A centralized inventory system allows players to manage collected clues and items. This helps in revisiting clues, reviewing suspect profiles, and tracking progress.

Points of Decision

Making Critical Decisions: Throughout the game, players make choices that affect how the investigation turns out. The story of the game can be changed by picking the incorrect suspect or ignoring important hints.

Final Accusation: At a pivotal point in the game, players must level an allegation based on the information they have acquired. Reaching the ending of the game requires correctly identifying the culprit.

Gameplay Experience

Immersive Narrative: The game immerses players in a narrative-driven experience by fusing suspense with exploration. The course of the inquiry is shaped by each choice and finding.

Puzzles & Challenges: There are puzzles to be solved, hints to be deciphered, and red herrings to avoid. Gamers are required to apply their problem-solving and deductive thinking abilities.

NEW CONCEPTS

Utilization of <iomanip> Library: The library played an essential part in the game's text output formatting. More specifically, text on the terminal screen might be centered because of the setw() method. Text might be visually centered for improved readability and presentation by computing padding depending on the console's width, which appears to be 80 characters.

Integration of Vectors: Throughout the game, dynamic lists of actions and inventory items were managed using vectors. During the inquiry, the player handled a number of clues, suspects, and previous events, all of which required the dynamic allocation and management of memory.

Implementation of Text Delay: The Sleep() method from the Windows API was used to provide text delay features in order to produce a more immersive experience. This feature improved the storytelling element of the game and made it feasible for players to take in information gradually by adding pressure and rhythm to narrative text.

Incorporation of Sound Effects: The game's mood was greatly enhanced by the incorporation of sound effects through the use of the PlaySound() function of the Windows API. An ambient backdrop was maintained during gaming by playing background audio continually, which enhanced overall immersion and enhanced the intense story.

Usage of ANSI Escape Codes: Certain components of the game interface were given visual effects, such colorful lettering, by using ANSI escape codes. To emphasize significant hints and thematic components, for example, the usage of \033[31m and \033[0m enabled the display of text in red and its subsequent reset to default color, respectively.

LIBRARIES

1. iostream: Standard input-output stream library in C++. Used for basic input and output operations to interact with the user via the console.
2. string: Offers string handling functionality; often used for storing and modifying textual information, including descriptions, hints, suspicious details, and so on.
3. ctime and cstdlib: Used for time-related functions (time()) and general utilities (rand(), srand()), crucial for generating random numbers (e.g., for suspect lies) and managing timestamps.
4. iomanip: Offers C++ output formatting manipulation tools. helpful for adjusting the display format of numerical numbers or aligning text.
5. windows.h: Contains functions unique to the Windows operating system, such as Sleep(), which adds millisecond delays. The timing and tempo of text display and game progression are managed by this library.
6. vector: A standard template library (STL) container in C++ used for dynamic arrays. In this project, it's utilized to maintain a history of actions (vector<string> history) and manage collections of clues and suspects.
7. MMSystem.h (Multimedia System): This library provides functions to play sound (PlaySound()) which adds an auditory dimension to the game, enhancing the immersive experience for the player.

These libraries collectively support various aspects of the game's mechanics, user interaction, and sensory engagement, enriching the overall gameplay experience.

CONTRIBUTION OF EACH MEMBER

CONTRIBUTOR	TASK DESCRIPTION
Misl e Noor Oveis	Designed Base Structure of Code and function (displayTextWithDelay(), accuseSuspects() and viewInventory()).
	Generation of report
Haiqa Anis Khan	Implementation of story to code along with functions (displaySuspect(),Derived class Riddles from Clue)and shadow text.
	Generation of repor0
Ahmad Raza	Added Different functions to the code(void initializeRooms(),void InvestigateRoom())along with Visual Representation of code(added Detective Body and color to it)
	Generation of report
M Kazim Abbas Lashari	Added Sound to enhance gameplay & made draft report
	Generation of report

CONCLUSION

Using interactive gameplay, object-oriented programming, compelling storyline, and other elements of game creation, the Detective Mystery Game project is a thorough implementation. With the help of this project, gamers may pretend to be detectives, figuring out complex riddles, gathering information, and making decisions that have a bearing on how the investigation turns out.

The design of the game places a strong emphasis on the value of user interaction, logical flow, and intricate story development. The use of ASCII art improves the aesthetic appeal and draws gamers into the virtual world. Players engage in a realistic and difficult detective story by keeping track of hints and questioning suspects.

Adding extra libraries and using certain development tools, such as Dev C++'s "-lwinmm" option, guarantees stable performance and increased features, such sound control for a more engaging experience. This project offers a thorough learning and development experience, as it explores new programming concepts and libraries in addition to demonstrating the application of fundamental OOP principles.

In the end, the Detective Mystery Game project exemplifies the smooth blending of story, reasoning, and player interaction, producing an interesting and cerebral game that tests players' ability to solve a challenging mystery.