CT-260

OBJECT-ORIENTED PROGRAMMING

LAB PROJECT REPORT

GAMEMANIA

GROUP MEMBERS:

- YUMNA IRFAN (CT-22004)
- ALEESHBA (CT-22005)
- HADIYA KASHIF (CT-22008)
- YASHA ALI (CT-22010)

COURSE INSTRUCTORS: DR. MURK MARVI

LAB INSTRUCTORS: SIR. ROHAIL QAMAR

MS. NUDRAT NAVED

TABLE OF CONTENTS

//The UML Diagram of our Code is sent separately as a pdf.

1. INTRODUCTION & HISTORY OF ARCADE GAMING:

Arcade gaming has been a prominent form of entertainment since the early 20th century, captivating players of all ages with its interactive and immersive experiences. The concept of arcade gaming emerged with the advent of mechanical gaming machines, which were initially found in amusement parks, bars, and restaurants.

It was during the 1970s and 1980s that arcade gaming experienced a true golden age. The introduction of breakthrough technologies, such as microprocessors and video displays, revolutionized the industry. Game developers began creating electronic arcade games that offered enhanced graphics, sound effects, and gameplay mechanics, captivating a vast audience.

2. PROBLEM STATEMENT:

To utilize the concepts of object-oriented programming and implement a C++-based arcade gaming system that offers users the opportunity to indulge in a thrilling gaming experience while incorporating modern elements to elevate the overall gameplay. The aim is to develop a unified system that combines five unique games, offering users a variety of gameplay options for entertainment and excitement.

3. UTILISING MAIN PILLARS OF OOP:

- Abstraction: We have designed an abstract game class named as “ARCADE” which contains the data members and pure virtual methods common to all classes. The pure virtual methods are:
    - initialize () which is responsible for initializing the SDL components,
    - handleEvents() which handles different controls depending on the game,
    - run () which is responsible for the proper running of the game, and

- cleanup () which frees the memory after execution is completed.

- **Inheritance:** we have used two types of inheritance in our code :
  1. HIERARCHIAL INHERITANCE: The five game classes namely , "PingPong" , "AstroStrike" , "SpookyChase" , "Tetris" , "MindMaze" are derived from the abstract class.
  2. MULTIPLE INHERITANCE: We created a "MainMenu" class which has multiple parent classes which are the above mentioned game classes. Due to which we faced "DIAMOND PROBLEM" which was solved by using VIRTUAL keyword with the parent classes of the MainMenu.

- **Encapsulation:** For encapsulation, we converted the source files of the all five games to header files and included them in the MainMenu's source file. We also declared the methods and data members of our game classes as private and protected. This ensured that our implementation is hidden from the user.

4. FEATURES OF OUR PROGRAM:

Some features of our program that distinguishes ours from others are:

1. Our arcade system has visually stunning graphics that encompass a rich array of audio elements, intricate images, and various font styles.
2. Our Program Uses Filing to Display "How to Play" / Instructions for each Game.
3. Our system offers a multitude of games, each equipped with a unique set of features designed to captivate and engage users.
4. Our system features user-friendly instructions for each game, allowing users to easily understand the rules and mechanics. This intuitive support enhances their gaming experience, enabling them to excel and fully enjoy their gameplay.
5. Our program enables seamless transitions between games, providing users the freedom to explore various gaming options without exiting the main screen. This maximizes their convenience and enjoyment by eliminating disruptions and inconveniences.

5. PROJECT DESCRIPTION:

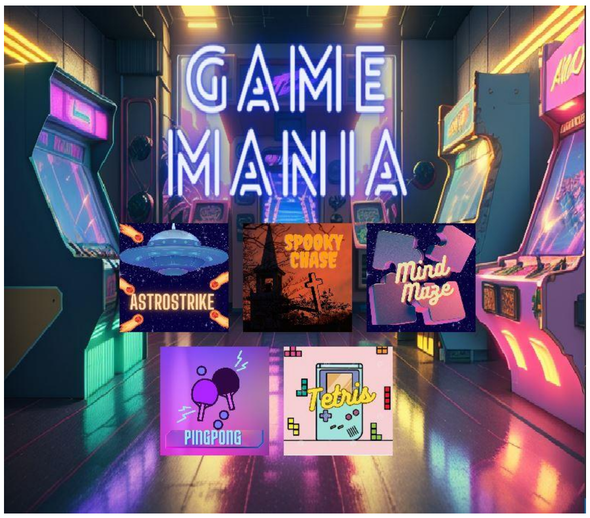
We have made an arcade gaming system as our lab project. When the program is executed a captivating main screen is displayed providing the user the choice to select from five games which are listed below:

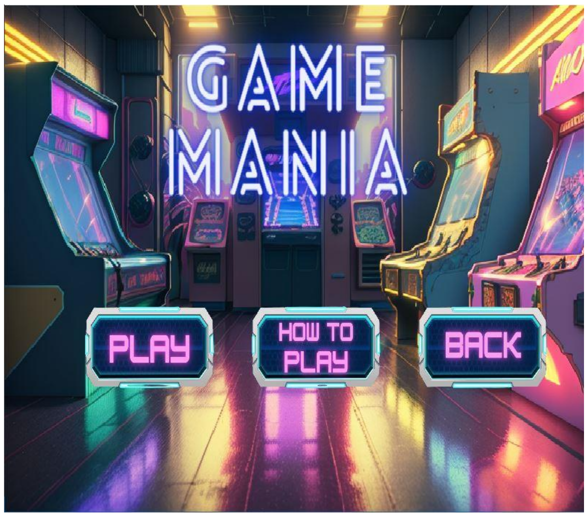
- Astro Strike
- Spooky Chase
- Mind Maze
- Ping Pong
- Tetris

Upon clicking on a game, a sub-menu is shown to the user, which includes the options for starting the game, displaying instructions for playing the game, and exiting the sub-menu.

6. HOW TO PLAY:

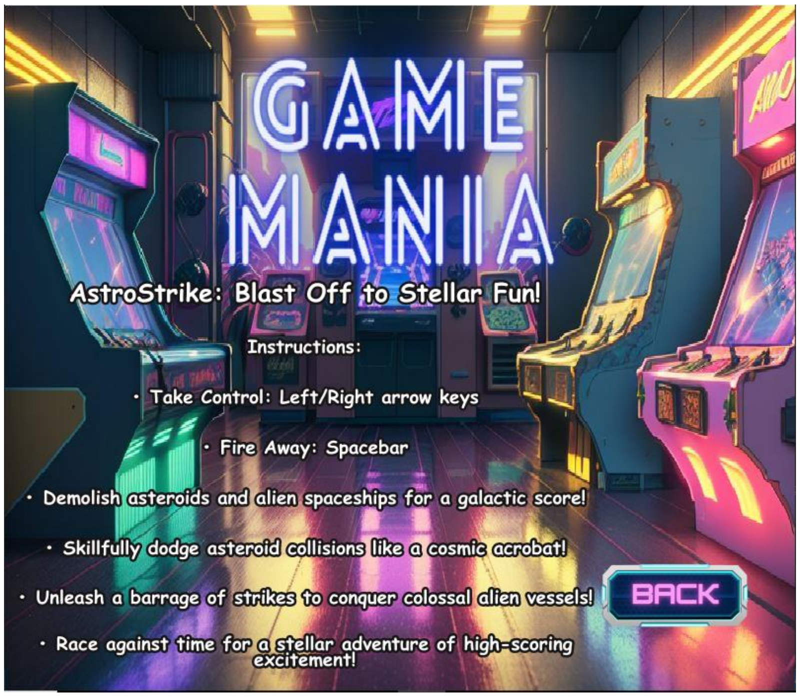
To start any game, the user has to click the game icon





Then select any desirable option.

By clicking on How To Play the users can view the instructions beforehand so that they can have a smooth experience.



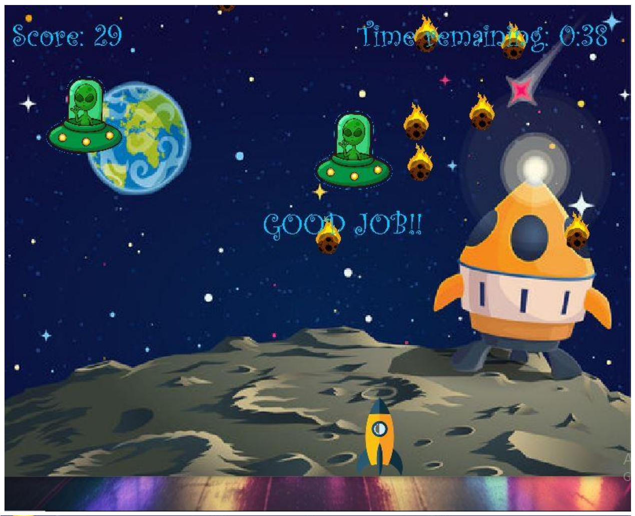
AstroStrike: Blast Off to Stellar Fun!

Instructions:

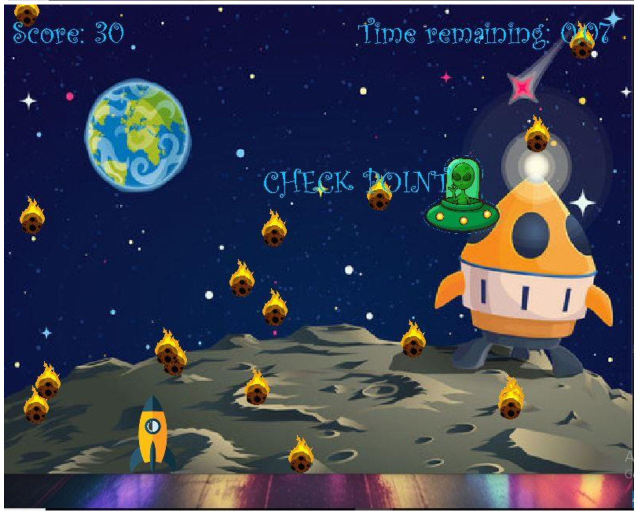
- Take Control: Left/Right arrow keys
- Fire Away: Spacebar
- Demolish asteroids and alien spaceships for a galactic score!
- Skillfully dodge asteroid collisions like a cosmic acrobat!
- Unleash a barrage of strikes to conquer colossal alien vessels!
- Race against time for a stellar adventure of high-scoring excitement!

Upon Clicking on Play Button the chosen Game Starts

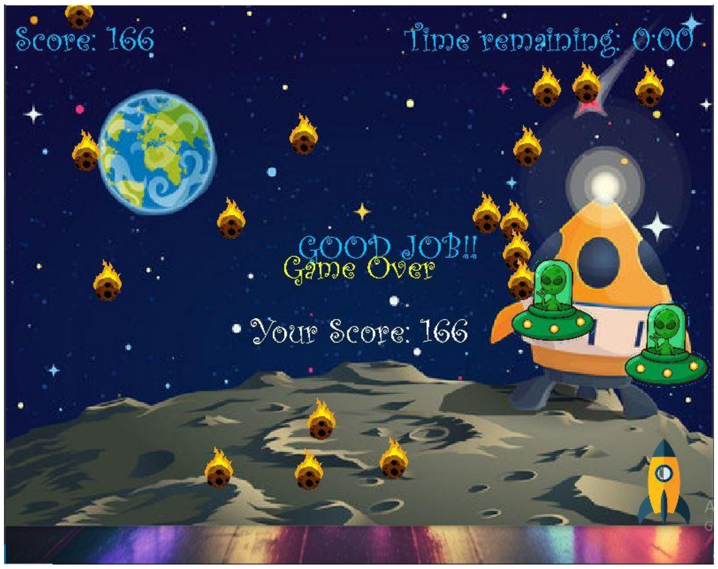
- **INSTRUCTIONS FOR ASTRO STRIKE:**



This is a Timer Based Shooting Game
Take Control of Spaceship: Left/Right arrow keys
Use Space Bar to shoot bullets and demolish Enemies

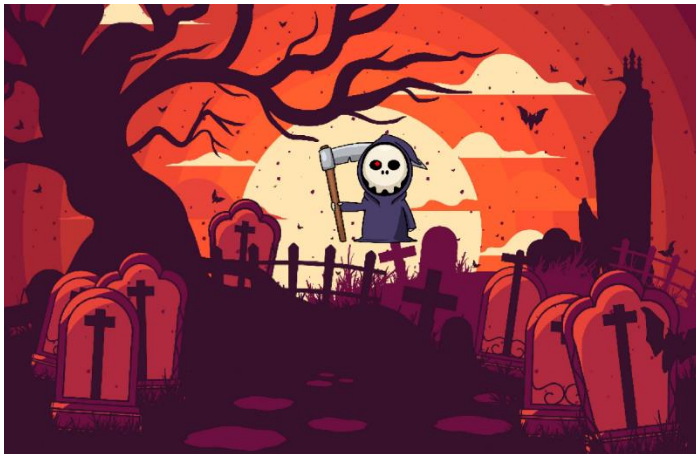


Aliens require 3 bullets to be destroyed and add 10 points to the Total Score while asteroids require only one bullet and add 1 point to the score.

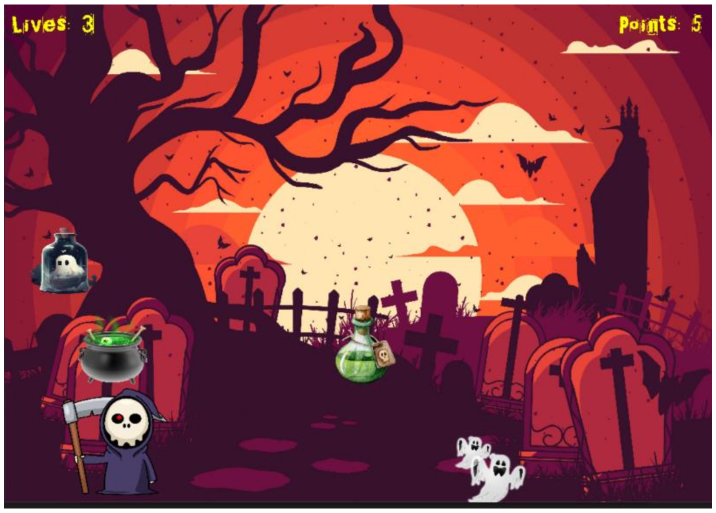


When the Game is Over Players Score is displayed along with Game Over message.

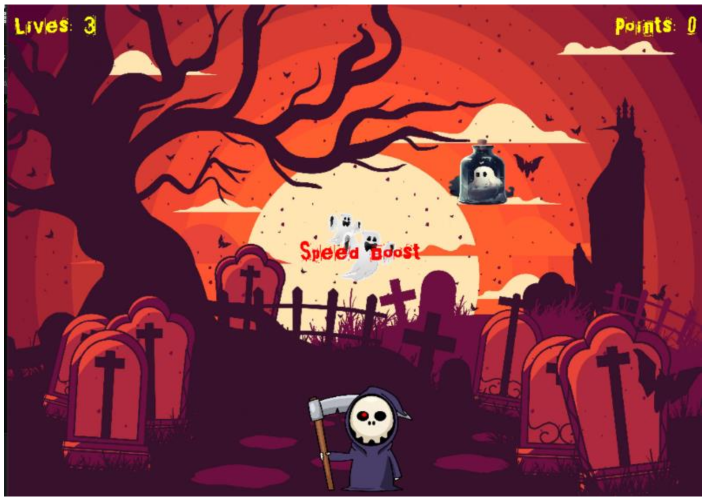
- **INSTRUCTIONS FOR SPOOK:**



To navigate the haunted realm use Up/Down/Left/Right Arrow Keys.

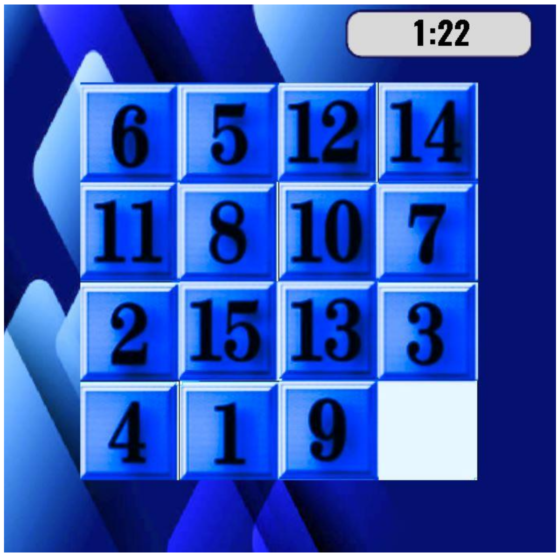


In total there are 3 lives which decrease on every collision with the ghostly spirits, your goal is to collect 100 points with at least one live remaining. The captured ghosts in the bottle award you 5 points on every collection.



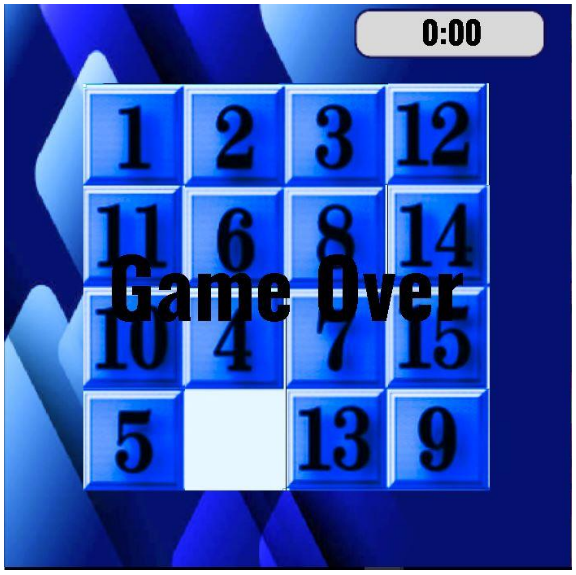
There are two types of powerups: Invincibility (Fear not of the decay as your lives are back), SpeedBoost (navigate the world, now with 2x speed)

- **INSTRUCTIONS FOR MIND MAZE:**



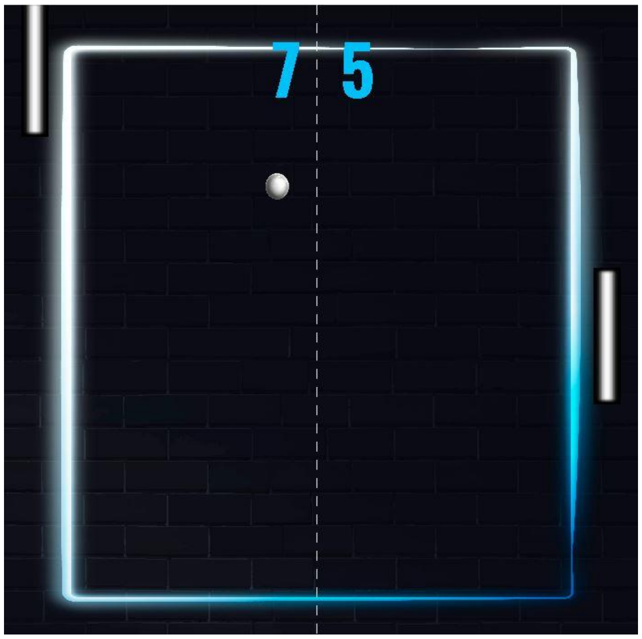
This is a number-sliding puzzle.

This is a single-player game, in which the player has to move the puzzle pieces marked by numbers. The movement is done using arrow keys.



The pieces have numbers 1 to 15 and the motive is to set the numbers in ascending order within two minutes or else the game is over.

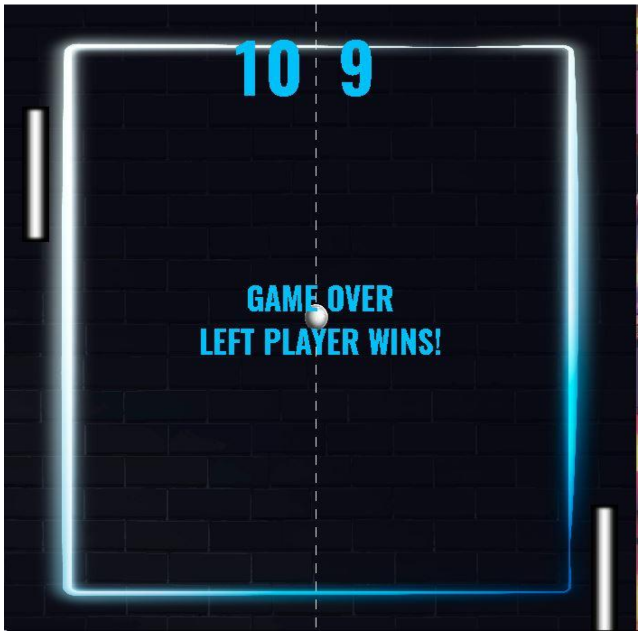
- **INSTRUCTIONS FOR PING PONG:**



This game is the replication of the classic "PONG" game.

This is a multiplayer game containing a ball and two paddles.

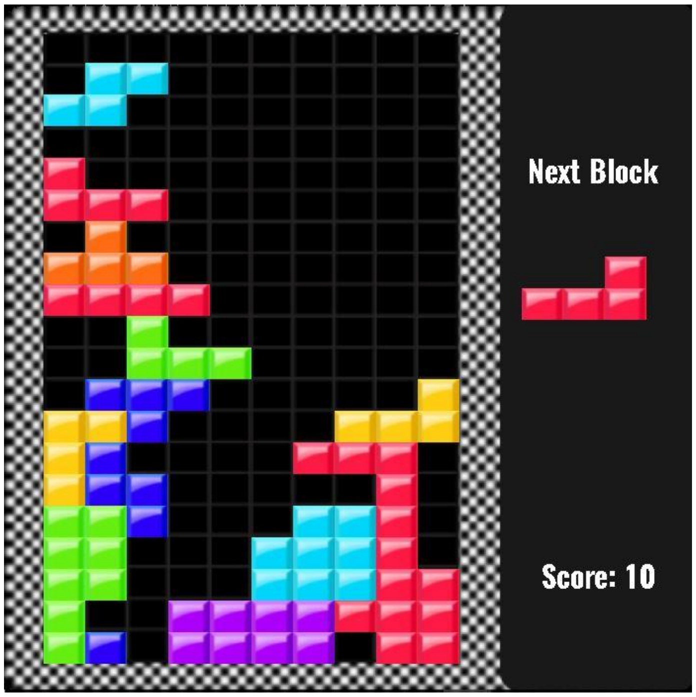
One paddle is controlled by the 'UP' and 'DOWN' arrow keys while the second one is controlled by using the 'W' key for moving up and the 'S' key for moving down.



Both players must try to hit the ball. If the ball is missed by one, a point is awarded to the other player.

The player who first reaches 10 points is declared the winner.

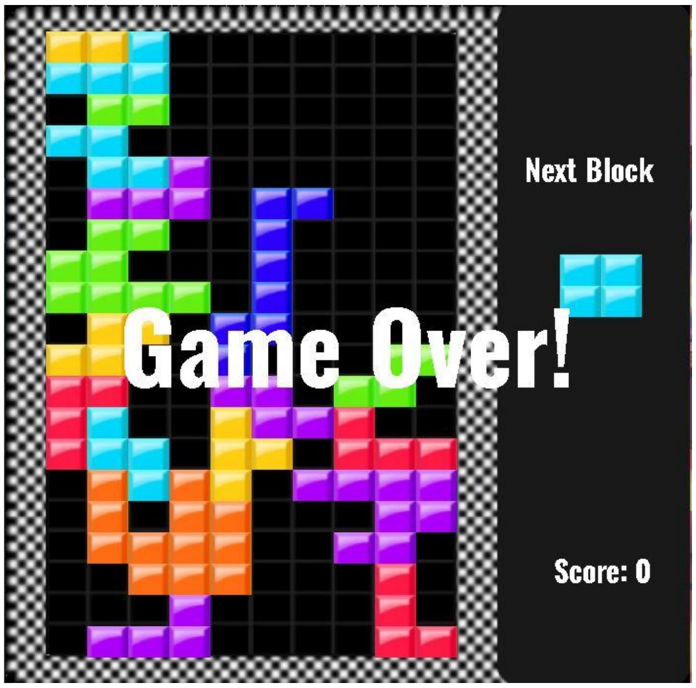
- **INSTRUCTIONS FOR TETRIS:**



This is a replication of the classic arcade game "TETRIS"

This is a single-player game, in which the player has to arrange coloured blocks, known as "TETROMINOES" in a straight line in the grid.

The tetrominoes are made up of 4 blocks, arranged in seven different patterns.



As soon as one line is completed it disappears, and 10 points are awarded.

Player can rotate the tetromino using the arrow keys to set it accordingly. A tetromino can be dropped quickly by pressing the 'DOWN' arrow key.

If the blocks are not arranged properly in a line, then eventually the grid will be filled to the top and the game will be over.

7. STEPS OF OUR SOLUTION:

Since the motive was to design a user-friendly and visually captivating arcade system, we decided to use the SDL Library for including graphics in the games. Along with SDL, we also used SDL_image for handling images in our program, SDL_ttf for dealing with fonts, and SDL_mixer, for incorporating audio in our games.

There are five games that were divided among the members and created separately. Afterward, we combined all the programs by making an abstract class “Game” which contains the data members and methods common in all the games.

All five games, namely “AstroStrike”, “Spooky Chase”, “Mind Maze”, “PingPong”, and, “Tetris”; are derived from the abstract class.

In each game, certain constants are set such as the window’s width and height and the dimensions of objects being used in that game. Each game has background music. Each game can be quitted mid-way by pressing ESC Key.

Each game’s source code was turned into a header file and included in the source code of the “Main window” to ensure encapsulation. The main window displays the game options to the users and also deals with the submenu of each game.
The source code of the main window is also turned into a header file, so the functionalities are hidden from the user. This as a header file is included in a source code that only starts the main window.

Upon clicking any game option a sub-menu is presented, which has three options, known as “Start”, “How To Play”, and ”Quit”.

DETAILS OF THE GAMES:

1. ASTROSTRIKE:

   "Astro Strike," is a game developed using the SDL library. The objective of the game is to control a spaceship and defeat enemies within a specified time limit. The code initializes the SDL library and loads necessary resources such as textures, fonts, and audio. It sets up the game window, renderer, and event handling for user input. Players can control the spaceship's movement using arrow keys and shoot bullets by pressing the spacebar. The game state is updated within a game loop, handling bullet movement, collision detection between bullets and enemies, and score calculation. The rendering process displays various game elements, including the player, bullets, enemies, score, remaining time, and special messages like game over and checkpoint notifications. The game loop continues until the specified game duration is reached, once the game is over, a game over message is displayed, providing players with their final score. Overall, the code offers a basic framework for an enjoyable 2D shooter game with a time-based challenge.

2. SPOOKY CHASE:

"Spooky Chase", this game utilizes the SDL library for graphics, audio, and input handling. The game's objective is for the player, represented by a character named Grim, to collect objects while avoiding obstacles. The player controls Grim's movement using the arrow keys. The game includes features such as collectibles that award points, obstacles that decrease lives upon collision, and power-ups that provide temporary advantages. The game continues until the player loses all lives or reaches a certain number of points. The score and remaining lives are displayed on the screen during gameplay. The code handles event inputs, updates object positions, detects collisions, renders game elements, and plays audio effects. The game window has a background image, and the game is accompanied by background music. Overall, the code provides a framework for a 2D game with simple mechanics and a spooky theme.

3. MIND MAZE:

This is a 15 pieces slider puzzle. There is a background image on which the timer is displayed and a puzzle grid is displayed.

The source code contains a puzzleGame class that deals with the whole game. It has handleEvents() functions that manage the movements of the pieces when the keys are pressed. It makes sure that the place where the player wants to move the place is empty. There is another function isPuzzleSolved () that checks if the values of the puzzle pieces are the same as the value which was expected at that place. The formatTime() function takes the milliseconds and converts them to seconds and minutes and then converts them to string to display on the screen. Then we have a render() function that renders text and images on the screen. The displayWonMessage() and displayGameOverMessage() display the text on the screen according to the situation. The update() function updates the screen according to the status of the puzzle being solved or not. There is a delay () function that closes the game window after two seconds.

4. PING PONG:

This is a classic game that everyone is familiar with. This is a two-player game in which we have a background on which two paddles and a ball are displayed. The screen has a partition between and the scores for both players are displayed on either side. When the game starts each player has to hit the ball or else a point is awarded to the opponent. Whoever first reaches the score of 10 is declared a winner.

The source code contains a Game class that manages the whole pong game. Struct has been utilized to represent the paddles in the game. The game class has a loadMedia () option that loads all the images and audio files. There is a handleEvents () function that controls the movement of the ball and the paddles based on the keys pressed. The update() method updates the game state, such as moving the ball and paddles, checking collisions, and updating scores. The method render(), renders the game on the screen, including paddles, ball, scores, and a game won / game over message.

5. TETRIS:

The provided code is an implementation of the classic game Tetris using the SDL library for graphics and audio rendering. It defines a `Tetris` class that encapsulates the game logic and rendering functionality. The game window, renderer, and necessary resources such as textures, fonts, and audio files are initialized in the `init()` function. The class handles keyboard input events for moving and rotating the falling tetriminos, updates the game state by checking for completed lines and updating the score, and renders the game field, falling tetrimino, score, and game over message using SDL functions. The game loop runs until the game is over or the user quits, providing an interactive Tetris gaming experience.

The code demonstrates good organization and encapsulation by utilizing a class to encapsulate the Tetris game functionality. It makes use of SDL functions to handle graphics, audio, and user input, allowing for a visually appealing and engaging game experience. The implementation of features such as block movement, rotation, line completion, and scoring ensures that the core mechanics of the Tetris game are present. With further expansion and refinement, the code could serve as a solid foundation for building a more feature-rich and polished Tetris game.

8. THE ADOPTED APPROACH:

The approach we adopted to make this Game Mania to the best of our abilities is that we utilized the SDL (Simple DirectMedia Layer) libraries as a foundation for implementing various games, including Spooky Chase, Astro Strike, Tetris, Ping Pong and Mind Maze. SDL offers a robust set of features and functions that facilitate game development, making it a versatile choice for creating different types of games.

For each game, the code implements specific gameplay mechanics, graphics rendering, collision detection, and scoring systems tailored to the unique requirements of the game. SDL's capabilities, such as rendering images and sprites, playing background music and sound effects, and capturing user input, greatly contribute to creating engaging and interactive gaming experiences.

9. FURTHER EXPANSION OF THE PROJECT:

To enhance and expand our arcade gaming system, we propose the inclusion of a coin system reminiscent of classic arcades. Additionally, we aim to offer a diverse collection of games with multiple levels, providing users with a wide range of gaming options to suit their preferences and ensure an enjoyable experience. Furthermore, we plan to introduce a leaderboard feature that showcases the top-performing players. By incorporating these functionalities, we aim to elevate the overall charm and appeal of our arcade gaming system.