

“Flappy Bird”

Department of Computer Science and
Engineering

A Project on Console Based Games using C Programming Language



Date: 18th December, 2023

Submitted to:

Meher Afroz

Lecturer, Department of CSE

Submitted By:

Rakib Hossan (20235203073)

Abdul Halim (20235203046)

Kaniz Fatema (20235203058)

Santo Mitro (20235203073)

Monisa Biswas (20235203041)

Fatema Akter Jhorna (20235203059)

Abstract

The “Flappy Bird” Game is designed to showcase and highlight the extensive efforts invested by gaming engineers or companies in crafting a single game. Our game provides a glimpse into the era when people engaged in gaming on computers or consoles before the advent of advanced graphics. The program is deliberately kept simple and user-friendly, ensuring ease of operation for anyone. It features a Main menu with six options for Console-Based Games developed using basic C Programming language principles. While this game may seem basic compared to contemporary titles, it represents the foundational roots of today's gaming milestones.

Acknowledgments

We express our gratitude to the Almighty for endowing us with the intellect to comprehend, analyze, and patiently develop processes. Special thanks go to our project supervisor, Meher Afroj, Lecturer in the Computer Science and Engineering Department at Bangladesh University of Business and Technology, for her professional guidance and unwavering motivation throughout the project. Her invaluable support has been pivotal in elevating the project to its current level of development.

Appreciation extends to all the esteemed faculty members of the Department of Computer Science and Engineering at Bangladesh University of Business and Technology. Their dedicated time spent on requirements analysis and project evaluation has been instrumental.

We extend heartfelt thanks to everyone who provided direct encouragement, offered constructive criticism, and indirectly contributed to shaping and refining this project at various stages of development.

Declaration

We hereby assert that the project on "Flappy Bird" submitted in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and Engineering at Bangladesh University of Business and Technology (BUBT), is our original work.

We affirm that it does not contain any material accepted for the award of any other degree or diploma, except where proper reference is made in the project text. To the best of our knowledge, it does not include any materials previously published or authored by any other individual, except where appropriate acknowledgment is issued in the project.

Contents

1	Introduction	1
1.1	Introduction	1
1.2	Objective.	2
1.3	Project Scope	2
1.4	Our Contributions	3
1.5	Conclusions	3
2	Existing Literature	5
2.1	Introduction	5
2.2	History of gaming	7
2.3	Growth in gaming industry	8
2.4	Conclusions	9
3	Proposed Model	11
3.1	Introduction	11
3.2	Software Requirement	11
3.2.1	SS of Encountered Interfaces	12
4	Implementation of Our System	13
4.1	Introduction	13
4.2	Overview	14
5	Experimental Results and Evaluation	15
5.1	Introduction	15
5.2	Result Analysis	16
6	Future Scope and Limitation	34
6.1	Future Scope of Our Project	34
6.2	Limitation	34
6.3	Conclusions	34
	Bibliography	35

1.1 Introduction

"Computers work like a melody, with hardware playing the role of vocal cords and software acting as lyrics—two crucial parts. In this section, we dive into the wonder of computer hardware, a versatile technology seamlessly woven into cameras, phones, thermostats, and more.

Here, hardware refers to the physical components, and software is made up of coded instructions that make the computer function. Our project, 'Multiple Console Based Games,' invites you on a nostalgic journey into the early days of console gaming. While not as fancy as today's games, it honors the lasting passion of gaming fans who tirelessly contribute to this lively world. Additionally, it advocates for genuine software, discouraging the use of pirated games. Come join us as we explore the beginnings of gaming and applaud the unwavering commitment of the gaming community."

1.2. Objective

The goal of this project is to offer users an immersive experience of playing console games in the contemporary graphical landscape. We have incorporated numerous features to enhance user satisfaction and engagement.

Menu Interface

- i. Start Game
- ii. View Highest Score
- iii. Reset Highest Score
- iv. Instructions
- v. About Game
- vi. Quit

1.3. Project Scope

The primary goal of this project is to develop a sophisticated software solution that incorporates advanced C structures, file operations, functions, switch conditions, loops, and other programming elements. The specific focus is on creating a framework that enables intricate and high-quality gaming experiences. The software will be designed to handle emphasizing precision, efficiency, and a seamless user experience. Through the implementation of robust coding practices, the objective is to deliver a software solution that meets the demanding requirements of modern graphical gaming environments.

This program proves exceptionally beneficial for computers with minimal processing power and limited Random Access Memory (RAM), such as those equipped with

**Atom or Celeron (Dual Core) processors and
2 GB of RAM.**

1.4. Our Contributions

- Playing console-based games on low-end machines is a secure and feasible option.
- There is no need for individuals to make a purchase to access these games.
- Moreover, the possibility exists to include these games as complimentary offerings bundled with the operating system.

1.5 Conclusions

Perfection remains elusive in our imperfect world, and we are no exception to this reality. Despite our earnest endeavors, it's crucial to acknowledge the inherent imperfections.

While we've invested our utmost efforts to convey information effectively, it's vital to recognize that achieving absolute perfection is challenging. Even though we've been meticulous in developing the project, there's still room for improvement.

Similar to any undertaking, this project has its limitations. Despite addressing crucial aspects, achieving 100% accuracy is constrained by inherent drawbacks. This highlights the opportunity for further enhancement, welcoming future contributions to refine and elevate the system.

2

Existing Literature

2.1 Introduction

A video game console is an electronic device that displays video games on a screen, allowing one or more players to use a game controller for interaction. These can be home consoles, stationary devices connected to a TV and controlled with a separate game controller, or handheld consoles with a built-in display and controller. Hybrid consoles combine features of both.

Video game consoles are specialized for gaming, prioritizing affordability and accessibility over raw computing power. They often use simplified distribution methods like game cartridges. While this simplifies game launches, it results in proprietary formats, creating competition for market share. Modern consoles also function as media players, supporting films and music from optical media or streaming services.

Consoles follow a 5–7-year cycle called a generation, with similar technical capabilities. The industry uses a razorblade model, selling consoles at a low profit or loss and making revenue from licensing fees for games. Planned obsolescence drives consumers to the next console generation. Sony (PlayStation), Microsoft (Xbox), and Nintendo (Switch) are the current market leaders.



Figure 3.1: Benefits of Gaming

2.2 History of gaming

The first video game consoles surfaced in the early 1970s when Ralph H. Baer conceived the idea of playing basic spot-based games on a television screen in 1966. This concept evolved into the Magnavox Odyssey in 1972. Inspired by the success of the table tennis game on the Odyssey, Nolan Bushnell, Ted Dabney, and Allan Alcorn at Atari, Inc. developed the first successful arcade game, Pong. They aimed to create a home version, which was released in 1975. Initially, consoles were dedicated to a specific set of games embedded in the hardware. The introduction of programmable consoles with swappable ROM cartridges began with the Fairchild Channel F in 1976 and gained popularity with the Atari 2600 in 1977.

Handheld consoles evolved as handheld electronic games transitioned from mechanical to electronic/digital logic. This shift moved away from LED indicators to liquid-crystal displays (LCD), closely resembling video screens. Early examples include the Micro vision in 1979 and Game Watch in 1980, with the concept fully realized by the Game Boy in 1989.

Since the 1970s, advancements in technology, including improved electronic and computer chip manufacturing, have led to increased computational power at lower costs and smaller sizes. The introduction of 3D graphics and hardware-based graphic processors for real-time rendering, as well as digital communications like the Internet, wireless networking, and Bluetooth, have further propelled the evolution of both home and handheld consoles. These consoles have been categorized into generations, each lasting about five years, with similar technology specifications and features, such as processor word size. While there is no universally accepted breakdown of home consoles by generation, the Wikipedia definition, including representative consoles is commonly used.

2.3 Growth in gaming industry

The gaming sector has witnessed a surge of 500 million new players in the last three years, reaching a global total of 2.7 billion individuals. Projections indicate an anticipated influx of over 400 million additional gamers by the end of 2023. The demographics of these newcomers are evolving, with 60% being female, 30% under 25 years old, and one-third identifying as non-white. In contrast, long-term gamers, constituting 61% male, 79% over 25 years old, and 76% identifying as white, present a different profile.

The evolution in gaming platforms and changing demographics is guiding gaming companies from a product-focused strategy to becoming experience-driven platforms. Emma Turner, director at Accenture's Software Solutions division, highlights the industry's struggle to harmonize the preferences of new users, who value online interactions, with the expectations of devoted gamers, the most profitable customer segment.

With the continuous expansion of the gaming community, the social element is increasingly crucial to overall gamer experiences. A noteworthy 84% of respondents express that video games facilitate connections with others who share similar interests. Additionally, three-quarters acknowledge that a significant portion of their social interactions now occurs on gaming platforms.

2.4 Conclusions

The Console Games has brought revolutionary changes to today's world of gaming. Even in today's world many people play console games on mobile consoles. Nintendo is one of the most popular consoles I have played once, which is now known as Nintendo Switch OLED.

3.1 Introduction

1. The current system relies on manual processes.
2. The revamped system provides an interactive menu for the Flappy Bird game, featuring six exciting options for users to explore.
3. This system prioritizes user-friendliness.
4. Scores (with player name) are securely stored in the score.txt file.

3.2 Software Requirement

Platform: Windows/Linux

IDE: Code blocks with GNU GCC Compiler

Hardware Requirements:

RAM: 2 GB

Processor: Atom or higher (Minimum 1 GHz speed)

3.2.1 SS of Encountered Interfaces

Below are screenshots showcasing the interfaces within our program. Explore the user interfaces you'll encounter in the program.

```
===== Flappy Bird Menu =====  
1. Start Game  
2. View Highest Score  
3. Reset Highest Score  
4. Instructions  
5. About Game  
6. Quit  
=====  
Enter your choice:
```

Figure 3.1: Menu Interface on console screen

Implementation of Our System

4.1 Introduction

Explanation of Key functions:

1. Game Menu:

In this section, players can choose from the following options to either start playing or exit the console.

2. Start Game:

Allows players to initiate gameplay.

3. View Highest Score:

Enables players to check the current high score.

4. Reset Highest Score:

Allows players to reset the high score.

5. Instructions:

Provides players with guidance on how to play the game.

6. About Game:

Furnishes players with information about the game itself.

7. Quit:

Allows players to exit the game effortlessly.

4.2 Overview

Presented herewith is the visual representation of the undertaken project.



Experimental Results and Evaluation

5.1 Introduction

In this segment, we have meticulously examined and presented an in-depth analysis of the project's outcomes. Additionally, we have made concerted efforts to showcase the output interfaces of our program, providing a comprehensive overview of the results achieved.

5.2 Result Analysis

The presented section showcases several screenshots depicting the output of our program. The initial screenshot provides an overview of the program's main menu.

```
// Function to display the main menu
void ShowMenu()
{
    printf("\n" YELLOW "=====Flappy Bird Menu =====\n" NC);
    printf("1. " CYAN "Start Game\n" NC);
    printf("2. " CYAN "View Highest Score\n" NC);
    printf("3. " MAGENTA "Reset Highest Score\n" NC);
    printf("4. " CYAN "Instructions\n" NC);
    printf("5. " CYAN "About Game\n" NC);
    printf("6. " RED "Quit\n" NC);
    printf(YELLOW "=====\n" NC);
    printf("Enter your choice: ");
}
```

Figure 5.1: SS of Main Menu code

```
===== Flappy Bird Menu =====
1. Start Game
2. View Highest Score
3. Reset Highest Score
4. Instructions
5. About Game
6. Quit
=====
Enter your choice:
```

Figure 5.1.1: Main Menu
on console screen

```

do
{
    ShowMenu();
    scanf("%d", &choice);

    switch (choice)
    {
        case 1:
            bird.x = 10;
            bird.y = 10;

            for (int i = 0; i < pipeCount; i++)
            {
                pipes[i].x = 25 + 15 * i;
                pipes[i].y = (rand() % 7) + 5;
            }

            LoadHighestScore(); // Load the highest score from the file

            int frame = 0;
            int gameState = 0;

            printf("\nPress " YELLOW "UP" GREEN " to jump, " YELLOW "P" GREEN " to pause, and " RED "Q" GREEN " to quit.\n");

            for (int i = 0; i <= ySize; i++)
            {
                printf("\n");
            }

            Draw();

            system("pause\n");

            while (1)
            {
                if (GetAsyncKeyState(VK_UP))
                {
                    bird.y -= 2;
                }

                if (GetAsyncKeyState(0x50)) // 0x50 is the virtual key code for 'P'
                {
                    PauseGame();
                }

                if (GetAsyncKeyState(qKey))
                {
                    choice = 6; // Quit the game
                    break;
                }
            }
        }
    }
}

```

Figure 5.2.1: SS of Start game code

```

    if (frame == 2)
    {
        bird.y++;

        for (int i = 0; i < 3; i++)
        {
            pipes[i].x--;
        }

        frame = 0;
    }

    gameState = HitTest();
    Draw();
    Pipes();

    if (gameState != 0)
    {
        GameOver();
        break;
    }

    frame++;
    Sleep(100);
    score++;
}
break;

```

Figure 5.2.2: SS of Start game code

```

// Function to draw the game screen
void Draw()
{
    char buff[5000];
    char scoreStr[50];
    sprintf(scoreStr, "Score: %d", score);
    sprintf(buff, "\e[17A");

    strcat(buff, "\e[s");
    strcat(buff, "\e[1;32;40m");
    strcat(buff, "\e[1;H");
    strcat(buff, scoreStr);
    strcat(buff, "\e[0m");
    strcat(buff, "\e[u");
}

```

Figure 5.2.3: SS of Game Screen Draw with score code

```
// Function to handle pipe movement and generation
void Pipes()
{
    for (int i = 0; i < pipeCount; i++)
    {
        if (pipes[i].x == -1)
        {
            (i == 0) ? (pipes[i].x = pipes[2].x + 15) : (pipes[i].x = pipes[i - 1].x + 15);
            pipes[i].y = (rand() % 7) + 5;
        }
    }
}

// Function to check for collisions and determine the game state
int HitTest()
{
    if (bird.y == 15 || bird.y == 2)
    {
        return 1; // Bird hit the floor or ceiling
    }

    for (int i = 0; i < pipeCount; i++)
    {
        if ((bird.x - 2 < pipes[i].x + 2) && (bird.x > pipes[i].x - 2) && ((bird.y < pipes[i].y - 2) || (bird.y > pipes[i].y + 1)))
        {
            return 2; // Bird hit a pipe
        }
    }

    return 0; // Bird is still alive
}
```

Figure 5.2.3: SS of Number of pipes and hit-test code



Figure 5.2.6: SS of Game paused on console screen


```
// Function to load the highest score from the file
void LoadHighestScore()
{
    FILE *file = fopen("scores.txt", "r+");
    if (file == NULL)
    {
        perror(RED"Error opening file"NC);
        return;
    }

    char playerName[50];
    int playerScore;
    while (fscanf(file, "Player: %s\nScore: %d\n", playerName, &playerScore) != EOF)
    {
        if (playerScore > highestScore)
        {
            highestScore = playerScore;
        }
    }

    fclose(file);
}

// Function to update the highest score
void UpdateHighestScore()
{
    if (score > highestScore)
    {
        highestScore = score;
    }
}
```

Figure 5.3.1: SS of Load and update highest score code from score.txt file

```
// Function to display the highest score
void ShowHighestScore()
{
    FILE *file = fopen("scores.txt", "r");
    if (file == NULL)
    {
        perror(RED"Error opening file"NC);
        return;
    }

    printf(CYAN "\n\n\n\n===== Highest Score =====\n\n" NC);

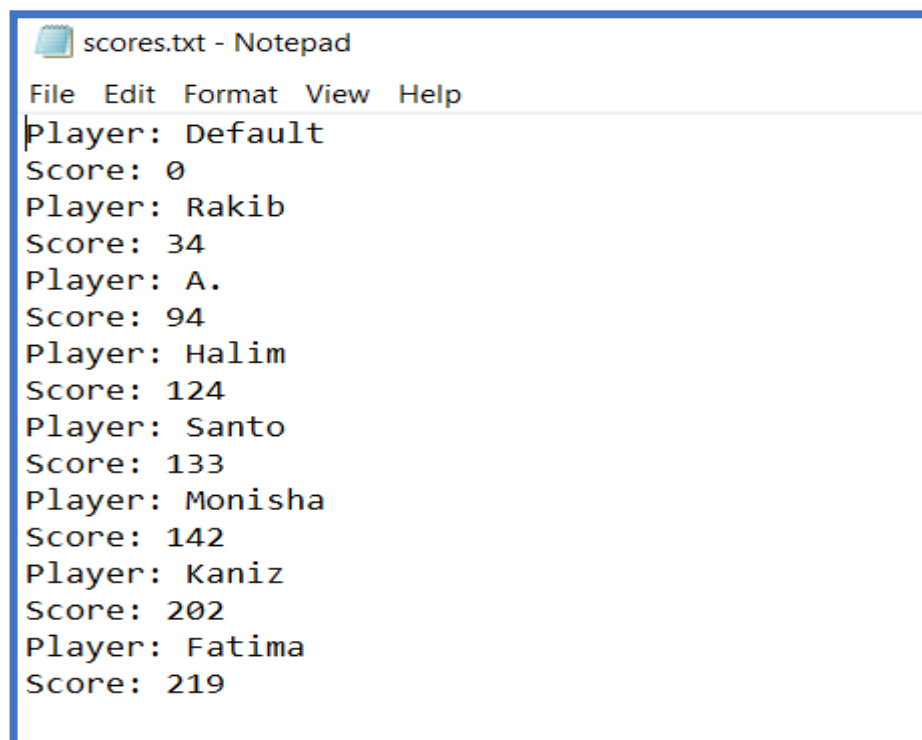
    char playerName[50];
    int playerScore;

    while (fscanf(file, "Player: %s\nScore: %d\n", playerName, &playerScore) != EOF)
    {
        if (playerScore == highestScore)
        {
            printf(RED "%s: %d\n" NC, playerName, playerScore);
        }
        else
        {
            printf("%s: %d\n", playerName, playerScore);
        }
    }
    printf(CYAN "\n\n\n\n===== \n\n\n\n" NC);
    fclose(file);
}
```

Figure 5.3.2: SS of Show highest score code from score.txt file

```
===== Highest Score =====  
Default: 0  
Rakib: 34  
A.: 94  
Halim: 124  
Santo: 133  
Monisha: 142  
Kaniz: 202  
Fatima: 219  
=====
```

Figure 5.3.3: SS of Show highest score on console screen



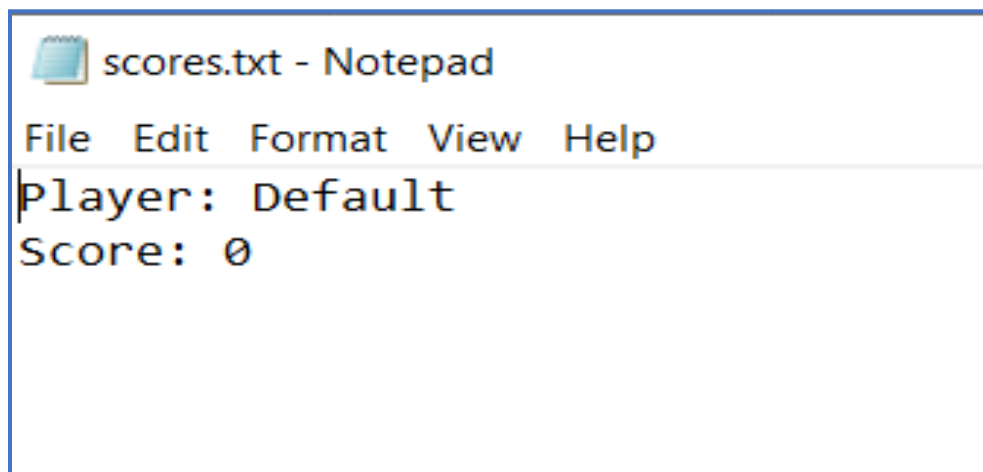
```
scores.txt - Notepad  
File Edit Format View Help  
Player: Default  
Score: 0  
Player: Rakib  
Score: 34  
Player: A.  
Score: 94  
Player: Halim  
Score: 124  
Player: Santo  
Score: 133  
Player: Monisha  
Score: 142  
Player: Kaniz  
Score: 202  
Player: Fatima  
Score: 219
```

Figure 5.3.3: SS of Load scores from scores.txt file

```
===== Flappy Bird Menu =====
1. Start Game
2. View Highest Score
3. Reset Highest Score
4. Instructions
5. About Game
6. Quit
=====
Enter your choice: 3

Score has been reset to 0.
```

Figure 5.4.1: SS of Reset highest score on console screen



```
scores.txt - Notepad
File Edit Format View Help
Player: Default
Score: 0
```

Figure 5.4.2: SS of Reset highest score
In score.txt file

```
// Function to display game instructions
void ShowInstructions()
{
    printf(GREEN "\n\n\n\n===== Flappy Bird Instructions =====\n\n" NC);
    printf(GREEN "1. Press " YELLOW "UP" GREEN " to make the bird jump.\n" NC);
    printf(GREEN "2. Avoid the pipes and survive as long as possible.\n" NC);
    printf(GREEN "3. Press " YELLOW "P" GREEN " to pause the game.\n" NC);
    printf(GREEN "4. Press " RED "Q" GREEN " to quit the game.\n" NC);
    printf(GREEN "5. Have fun and aim for the highest score!\n" NC);
    printf(GREEN "\n===== \n\n\n\n" NC);
}
```

Figure 5.5.1: SS of Instructions code

```
===== Flappy Bird Menu =====
1. Start Game
2. View Highest Score
3. Reset Highest Score
4. Instructions
5. About Game
6. Quit
=====
Enter your choice: 4

===== Flappy Bird Instructions =====

1. Press UP to make the bird jump.
2. Avoid the pipes and survive as long as possible.
3. Press P to pause the game.
4. Press Q to quit the game.
5. Have fun and aim for the highest score!

=====
```

Figure 5.5.2: SS of Instructions on console screen

```
// Function to display information about the game
void ShowAboutGame ()
{
    printf(CYAN "\n\n\n\n          ===== About Flappy Bird =====\n\n" NC);
    printf(CYAN "Flappy Bird is a simple and addictive game where you control a bird\n");
    printf(CYAN "and navigate it through a series of pipes. Your objective is to survive\n");
    printf(CYAN "as long as possible without hitting the pipes or the ground. The longer\n");
    printf(CYAN "you survive, the higher your score becomes. Challenge yourself to beat\n");
    printf(CYAN "your own highest score and enjoy the classic gameplay of Flappy Bird!\n");
    printf(CYAN "\n\n          ===== Developed By =====\n");
    printf(CYAN "\n      |-----|\n");
    printf(CYAN "          Rakib Hossan          \n");
    printf(CYAN "      |-----|\n");
    printf(CYAN "      |-----|\n");
    printf(CYAN "          Abdul Halim          \n");
    printf(CYAN "      |-----|\n");
    printf(CYAN "      |-----|\n");
    printf(CYAN "          Kaniz Fatema          \n");
    printf(CYAN "      |-----|\n");
    printf(CYAN "      |-----|\n");
    printf(CYAN "          Santo Mitro          \n");
    printf(CYAN "      |-----|\n");
    printf(CYAN "      |-----|\n");
    printf(CYAN "          Monisa Biswas          \n");
    printf(CYAN "      |-----|\n");

    printf(CYAN "      |-----|\n");
    printf(CYAN "          Fatema Akter Jhorna          \n");
    printf(CYAN "      |-----|\n");
    printf(CYAN "\n          =====\n\n\n\n" NC);
}
```

Figure 5.6.1: SS of About game code

```
===== About Flappy Bird =====

Flappy Bird is a simple and addictive game where you control a bird
and navigate it through a series of pipes. Your objective is to survive
as long as possible without hitting the pipes or the ground. The longer
you survive, the higher your score becomes. Challenge yourself to beat
your own highest score and enjoy the classic gameplay of Flappy Bird!

===== Developed By =====

|-----|
|      Rakib Hossan      |
|-----|
|-----|
|      Abdul Halim      |
|-----|
|-----|
|      Kaniz Fatema      |
|-----|
|-----|
|      Santo Mitro      |
|-----|
|-----|
|      Monisa Biswas     |
|-----|
|-----|
|      Fatema Akter Jhorna |
|-----|

=====
```

Figure 5.6.2: SS of About game on console screen

```
int main()
[ {
    srand(time(NULL)); // Seed the random number generator
    system("title \"Flappy Bird\""); // Set the console title

    int choice;

    do
    [ {
        ShowMenu();

        // Validate user input to ensure it is an integer
        if (scanf("%d", &choice) != 1)
        [ {
            printf(RED "\nInvalid input. Please enter a number.\n" NC);
            // Clear the input buffer in case of invalid input
            while (getchar() != '\n');
            continue;
        ]

        switch (choice)
        [ {
```

Figure 5.7.1: SS of Validate user input about integer(code)


```
case 6:
    printf(YELLOW "\n\n\n===== Goodbye! =====\n\n\n" NC);
    break;

default:
    printf(RED "\nInvalid choice. Please try again.\n" NC);
    break;
}

// Clear the input buffer
while (getchar() != '\n');

}
while (choice != 6);

getch();
}
```

Figure 5.7.2: SS of Good Bye and Invalid input code

```
===== Flappy Bird Menu =====
1. Start Game
2. View Highest Score
3. Reset Highest Score
4. Instructions
5. About Game
6. Quit
=====
Enter your choice: 8

Invalid choice. Please try again.

===== Flappy Bird Menu =====
1. Start Game
2. View Highest Score
3. Reset Highest Score
4. Instructions
5. About Game
6. Quit
=====
Enter your choice:
```

Figure 5.8.1: SS of Invalid input on console screen

```
===== Flappy Bird Menu =====  
1. Start Game  
2. View Highest Score  
3. Reset Highest Score  
4. Instructions  
5. About Game  
6. Quit  
=====
```

Enter your choice: 6

```
===== Goodbye! =====
```

Figure 5.7.2: SS of Good Bye on console screen

Future Scope and Limitation

6.1 Future Scope of Our Project

In the future, graphics can be incorporated into this project, allowing it to serve as a testing ground or a built-in game within software, similar to the Dino game in the Chrome browser.

6.2 Limitation

The project is intentionally designed with minimal flexibility. Once the user pause game, the screen gets down and there is no provision for returning to a previous state; the entire console needs to be reset to exit the game.

6.3 Conclusions

After developing this entire project using Object-Oriented Programming, we can assert that even more refined interfaces and outputs can be achieved with minimal and hassle-free modifications. However, our journey has just commenced, and there is a long way to go. We have a vast world of knowledge at our disposal, waiting to be explored and learned.

Bibliography

- Herbert Schildt. Teach Yourself C. 1997.
- Herbert Schildt. Teach Yourself C++. 1998.
 - Schildt (1997)
 - Schildt (1998)
- https://www.youtube.com/results?search_query=advanced+c+programming
- YouTube: <https://www.youtube.com/@freecodecamp>

