A
Summer Internship Report
On
DSA (C++) Programming

(CE346 – Summer Internship - I)

Prepared by
Veer Rana (22CE109)

Under the Supervision of
Prof. Ronak N Patel

Submitted to

Charotar University of Science & Technology (CHARUSAT)
for the Partial Fulfillment of the Requirements for the
Degree of Bachelor of Technology (B.Tech.)
for Semester 5

Submitted at

U & P U. PATEL DEPARTMENT OF COMPUTER ENGINEERING
Chandubhai S. Patel Institute of Technology (CSPIT)
Faculty of Technology & Engineering (FTE), CHARUSAT
At: Changa, Dist: Anand, Pin: 388421.
July 2024

CHARUSAT
CHAROTAR UNIVERSITY OF SCIENCE AND TECHNOLOGY

CERTIFICATE

This is to certify that the report entitled "**DSA (C++) Programming**" is a bonafied work carried out by **Veer Rakeshkumar Rana (22CE109)** under the guidance and supervision of **Prof. Ronak N Patel** for the subject **Summer Internship – I (CE346)** of 5th Semester of Bachelor of Technology in **Computer Engineering** at Chandubhai S. Patel Institute of Technology (CSPIT), Faculty of Technology & Engineering (FTE) – CHARUSAT, Gujarat.

To the best of my knowledge and belief, this work embodies the work of candidate himself, has duly been completed, and fulfills the requirement of the ordinance relating to the B.Tech. Degree of the University and is up to the standard in respect of content, presentation and language for being referred by the examiner(s).

Under the supervision of,

Prof. Ronak N Patel
Assistant Professor
U & P U. Patel Dept. of Computer Engineering
CSPIT, FTE, CHARUSAT, Changa, Gujarat

Mr. Latesh Bansal
InternPe

Dr. Nikita Bhatt
Head - U & P U. Patel Department of Computer Engineering,
CSPIT, FTE, CHARUSAT, Changa, Gujarat.

Chandubhai S. Patel Institute of Technology (CSPIT)

Faculty of Technology & Engineering (FTE), CHARUSAT

At: Changa, Ta. Petlad, Dist. Anand, Pin: 388421. Gujarat.

InTernPe
LET'S START

@internPE

INTERNSHIP COMPLETION CERTIFICATE

CID : IPI #18747

To whomever it may concern

This is to certify that **VEER RAKESHKUMAR RANA** worked as an Intern in our company from **13-May-2024 to 09-June-2024**

Please find the internship details below:

Company Name: InternPe
Location: Remote
Domain :DSA(C++)Programming
Designation: Intern

During their working period, we found him/her to be a sincere and dedicated intern with a professional attitude and very good knowledge of the job.

We thank him/her for their efforts and contribution and wish him/her the best in future endeavors.

Yours Sincerely

(Co-Founder)
InternPe

InternPe

CERTIFIED
ISO
9001:2015
COMPANY

MSME
MICRO, SMALL & MEDIUM ENTERPRISES
सूक्ष्म, लघु एवं मध्यम उद्यम

www.internpe.in

ACKNOWLEDGEMENT

I would like to express my deepest gratitude to everyone who supported me throughout my internship and contributed to the completion of this report on Data Structures and Algorithms (C++) programming.

Firstly, I am sincerely thankful to my internship supervisor at InternPe, for providing me with the opportunity to work on projects. Their constant guidance, insightful feedback, and unwavering support were invaluable in honing my skills and deepening my understanding of DSA in C++.

I am also grateful to my academic mentor, Prof. Ronak N Patel, at Chandubhai S Patel Institute of Technology(CSPIT)-Computer Engineering for their continuous encouragement and for imparting fundamental knowledge that laid the groundwork for this internship.

I would also like to acknowledge the valuable resources and support provided by the InternPe's technical team, which facilitated my research and implementation of various data structures and algorithms in C++.

Finally, I extend my heartfelt appreciation to my family and friends for their unwavering support, understanding, and encouragement throughout this period.Thank you all for your significant contributions to the successful completion of this internship and the preparation of this report.

Sincerely,

VEER RANA

Abstract

This report summarizes my internship at InternPe, focusing on Data Structures and Algorithms (DSA) using C++. The internship aimed to enhance my skills in implementing various data structures and algorithms crucial for efficient programming. During the internship, I worked on different game projects and all implemented in C++. Practical projects involved developing solutions to complex problems, optimizing code to design robust software components. The internship provided a deep understanding of selecting appropriate data structures and algorithms for efficient problem-solving. It significantly improved my technical and problem-solving skills, preparing me for future challenges in software development. Overall, the experience at InternPe was highly valuable, contributing to my growth as a software developer.

Description of company-InternPe

InternPe is a forward-thinking company dedicated to bridging the gap between academic learning and practical industry experience. Specializing in providing internship opportunities across various domains, InternPe connects students and recent graduates with top-tier companies to facilitate real-world learning and professional growth.

InternPe offers a wide array of internship programs tailored to different fields such as software development, data science, marketing, finance, and more. By partnering with leading industry players, InternPe ensures that interns gain hands-on experience, work on real projects, and develop the skills required to excel in their chosen careers.

The company's mission is to empower the next generation of professionals by providing them with the necessary tools, resources, and opportunities to succeed in the competitive job market. InternPe's innovative approach and commitment to excellence make it a preferred choice for both interns seeking valuable experience and companies looking to nurture fresh talent.

Table of Contents

List of Figures

List of Tables

CHAPTER-1 INTRODUCTION

1.1 INTERNSHIP OBJECTIVES

Enhance Technical Proficiency:

- Gain a deep understanding of fundamental and advanced data structures and algorithms in C++.

Improve Problem-Solving Skills:

- Practice designing efficient algorithms for sorting, searching, and optimization.
- Apply theoretical knowledge to solve complex computational problems.

Practical Application:

- Write clean, efficient, and well-documented code following industry standards.
- Work on real-world projects that require the application of DSA principles.

Documentation and Reporting:

- Develop the ability to present technical information clearly and concisely to different audiences.
- Learn to document code effectively and write comprehensive reports on project progress and outcomes.

As an intern, I gained a solid foundation in data structures and algorithms using C++, enhance my problem-solving skills, and acquire practical experience that will prepare me for future roles in software development. This internship is a crucial step in my professional growth, providing me with the skills and knowledge necessary in the industry.

1.2 OVERVIEW OF INTERNSHIP ACTIVITIES

Table 1.1 Internship activities

	Date	Day	Name of Topic/Module
Week 1	13/5/2024	Monday	Notes about Introduction to C++
	14/5/2024	Tuesday	Introduction to Task-1 (Guess a Number Game)
	15/5/2024	Wednesday	Quiz-1 based on Basic of C++
	16/5/2024	Thursday	Watching reference video and creating Game
	17/5/2024	Friday	Submission of Task-1
Week 2	20/5/2024	Monday	Notes about Basic of OOPS
	21/5/2024	Tuesday	Introduction to Task-2 (Rock Paper Scissor Game)
	22/5/2024	Wednesday	Quiz-2 based on Basic of OOPS
	23/5/24	Thursday	Watching reference video and creating Game
	24/5/24	Friday	Submission of Task-2
Week 3	27/5/24	Monday	Notes about content of C++
	28/5/24	Tuesday	Introduction to Task-3 (TIC TAC TOE Game)
	29/5/24	Wednesday	Take Reference about GAME
	30/5/24	Thursday	Make Complete Game
	31/5/24	Friday	Submission of Task-3
Week 4	3/6/24	Monday	Introduction to Task-4(CONNECT FOUR Game)
	4/6/24	Tuesday	Reference from given document
	5/6/24	Wednesday	Make Complete Game
	6/6/24	Thursday	Submission of Game

CHAPTER 2 TOOLS AND TECHNOLOGIES

2.1 INTRODUCTION TO C++

Features and Advantages

C++ combines both high-level and low-level programming features, making it suitable for a wide range of applications. Some of its notable features include:

- **Object-Oriented Programming (OOP):** C++ supports classes and objects, allowing you to encapsulate data and behavior into reusable modules. This promotes code organization and reusability.
- **Inheritance and Polymorphism:** C++ supports class inheritance, enabling the creation of hierarchies of related classes. This facilitates code reuse and the implementation of polymorphic behavior through virtual functions.
- **Operator Overloading:** C++ allows you to define custom behaviors for operators when applied to user-defined types. For example, you can define how the `+` operator works for your custom class objects.
- **Templates:** C++ supports templates, which allow you to Write generic code that works with different data types. Function templates and class templates enable code flexibility and reusability.
- **Standard Template Library (STL):** The STL provides a collection of useful data structures (like vectors, maps, and queues) and algorithms (sorting, searching, etc.), simplifying common programming tasks.
- **Rich Ecosystem:** C++ has a vast ecosystem of libraries and frameworks that cover various domains, from game development to scientific computing.
- **Standardization:** C++ is standardized by the ISO C++ standards committee, ensuring that the language evolves in a controlled and well-defined manner.

C++ Application Areas

C++ is widely used in various domains due to its versatility and performance characteristics:

- **System Software:** Operating systems, device drivers, and embedded systems often use C++ due to its low-level capabilities.
- **Game Development:** Many video games are developed using C++ for its efficiency and ability to interact with hardware directly.
- **Graphics and Multimedia:** Graphics rendering engines and multimedia applications benefit from C++'s performance and low-level control.
- **Scientific Computing:** C++ is used in scientific simulations, data analysis, and mathematical modeling.

- **Large-Scale Software:** C++ is employed in building complex software systems, where the combination of OOP and low-level control is valuable.

Conclusion

C++ is a powerful and versatile programming language that has evolved from its C roots to become a cornerstone of modern software development. Its combination of low-level control and high-level abstractions makes it suitable for a wide range of applications, from system programming to high-performance applications. As you proceed with this guide, you will delve deeper into the various aspects of C++, mastering its syntax, concepts, and best practices.

2.1.1 INTRODUCTION TO OOPS

- **Classes and Objects**

  A class is a blueprint for creating objects, which are instances of that class. Classes define the attributes (data members) and behaviours (member functions) of objects.

- **Inheritance**

  Inheritance allows you to create a new class (derived class) based on an existing class (base class). The derived class inherits the attributes and behaviors of the base class and can add its own.

- **Polymorphism**

  Polymorphism allows objects of different classes to be treated as objects of a common base class. It enables dynamic behavior based on the actual object type.

- **Encapsulation**

  Encapsulation is the principle of bundling data and related behaviors into a single unit (class). It hides the internal implementation details and exposes a controlled interface.

- **Abstraction**

  Abstraction focuses on providing a simplified view of an object or system by exposing only the necessary details to the user. It helps manage complexity and make interactions more intuitive.

- **Conclusion**

  Understanding advanced Object-Oriented Programming concepts like inheritance, polymorphism, encapsulation, and abstraction empowers you to design more sophisticated and flexible software. By modeling real-world entities as classes and utilizing the principles of OOP, you can create well-structured, maintainable, and extensible applications in C++.

2.1.2 INTRODUCTION TO INHERITANCE AND POLYMORPHISM

Inheritance and polymorphism are key concepts in Object-Oriented Programming (OOP) that facilitate code reuse, extensibility, and dynamic behavior. This chapter explores these concepts in depth, illustrating how they can be effectively applied in C++.

- **Inheritance**

  Inheritance is a mechanism that allows a class to inherit properties and behaviors from another class. The class that is inherited from is called the base class or parent class, and the class that inherits is called the derived class or child class.

- **Inheritance Types**

1. Public Inheritance: Public and protected members of the base class become public and protected members of the derived class, respectively.
2. Protected Inheritance: Public and protected members of the base class become protected members of the derived class.
3. Private Inheritance: Public and protected members of the base class become private members of the derived class.

- **Polymorphism**

  Polymorphism allows objects of different classes to be treated as objects of a common base class. It enables dynamic behavior based on the actual object type.

- **Pure Virtual Functions and Abstract Classes**

  A pure virtual function is a function declared in a base class but not defined. It's meant to be overridden by derived classes. A class containing at least one pure virtual function is an abstract class, which cannot be instantiated.

- **Conclusion**

  Inheritance and polymorphism are powerful tools in Object-Oriented Programming that enable code reuse, extensibility, and dynamic behavior. By effectively utilizing inheritance to model relationships between classes and employing polymorphism to achieve flexibility and maintainability, you can create sophisticated and adaptable software systems in C++.

2.1.2 INTRODUCTION TO EXCEPTION HANDLING

Exception handling is a crucial feature in C++ that allows you to handle unexpected or exceptional situations in a controlled and organized manner. In this chapter, we'll explore how exception handling works, the benefits it offers, and how to effectively use it in your programs.

Exception handling provides a mechanism to deal with errors, anomalies, or exceptional conditions that might occur during program execution. It allows you to separate error handling code from the main logic, enhancing code readability and maintainability.

- try Block: This block contains the code that may raise an exception.
- catch Block: This block catches and handles exceptions raised in the try block.
- throw Statement: This statement is used to raise an exception explicitly.

2.2 HARDWARE REQUIMENTS

- Processor: Intel dual core or above your head
- Processor Speed: 1.0GHZ or above
- RAM: 4 GB RAM or above
- Hard Disk: 20 GB hard disk or above

2.2 SOFTWARE REQUIMENTS

- Tools: VS Code, ScreenRecoder
- Technology: C++

CHAPTER 3 TASK DESCRIPTION

Table 3.1 Task-1

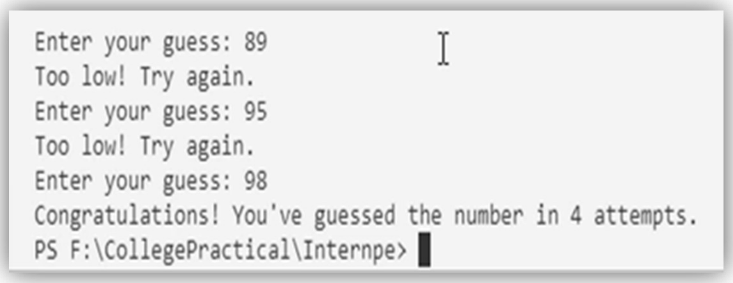
Task 1	
Task Description	Make a Guess a Number Game With the help of C++ After that make a separate Video on it.
Trigger	Compile code and run on terminal.
Basic Path	1 Enter Guessed number. 2 If Show too low! Then enter bigger number from previously entered number. 3 If Show too high! Then enter smaller number from previously entered number.
Post Condition	Print **‘Congratulations! You guessed number in (number of attempt)’**
Output Screenshots	 Enter your guess: 89 Too low! Try again. Enter your guess: 95 Too low! Try again. Enter your guess: 98 Congratulations! You've guessed the number in 4 attempts. PS F:\CollegePractical\Internpe> Fig. 3.1 Task-1 Output

Table 3.2 Task-2

Task 2	
Task Description	Make a ROCK PAPER SCISSOR GAME With the help of C++ After that make a separate Video on it.
Trigger	Compile code and run on terminal.
Basic Path	1. Enter R for Rock, P for Paper, S for Scissors (Choose any one out of three) 2. Program (Computer) will generate any random form given three
Post Condition	If User and Computer choice matched Shows **'It's a tie!'** Else Shows **'You Win! Or Computer Win!'**

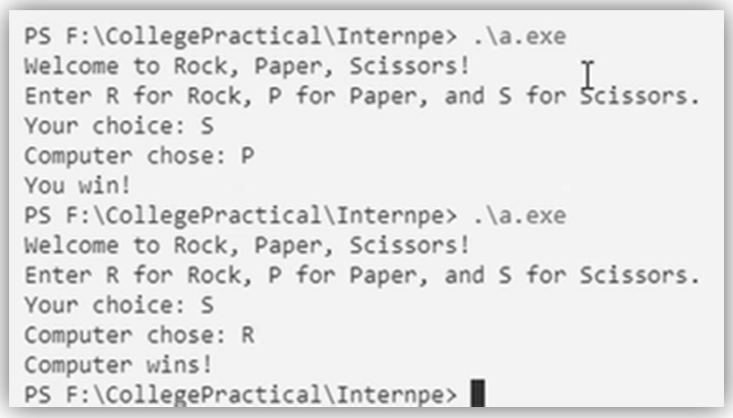
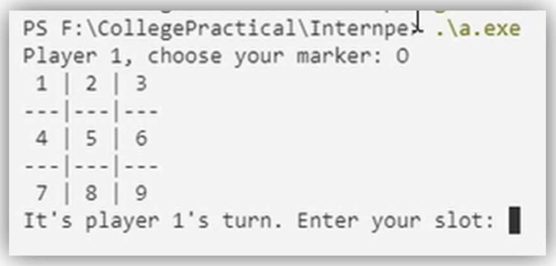
Output Screenshots	 `PS F:\CollegePractical\Internpe> .\a.exe`
	`Welcome to Rock, Paper, Scissors!`
	`Enter R for Rock, P for Paper, and S for Scissors.`
	`Your choice: S`
	`Computer chose: P`
	`You win!`
	`PS F:\CollegePractical\Internpe> .\a.exe`
	`Welcome to Rock, Paper, Scissors!`
	`Enter R for Rock, P for Paper, and S for Scissors.`
	`Your choice: S`
	`Computer chose: R`
	`Computer wins!`
	`PS F:\CollegePractical\Internpe>`
	Fig. 3.2 Task-2 Output

Table 3.3 Task-3

<table>
<tr><th colspan="2">Task 3</th></tr>
<tr><td><b>Task Description</b></td><td>Make a TIC TAC TOE GAME With the help of C++
After that make a separate Video on it.</td></tr>
<tr><td><b>Trigger</b></td><td>Compile code and run on terminal.</td></tr>
<tr><td><b>Basic Path</b></td><td>1. Choose player-1 mark form (0/X) and another one apply for player-2.

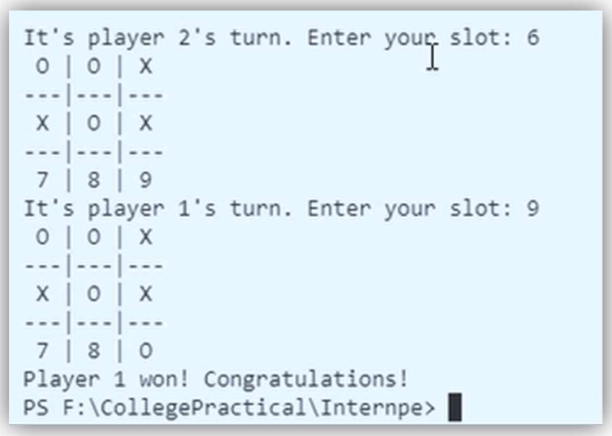


```
PS F:\CollegePractical\Internpe> .\a.exe
Player 1, choose your marker: O
 1 | 2 | 3
---|---|---
 4 | 5 | 6
---|---|---
 7 | 8 | 9
It's player 1's turn. Enter your slot:
```

Fig. 3.3 Task-3 Starting of game

2. Player-1 has to choose it's slot form 3X3 matrix.

3. After Player-2 has to choose it's slot form 3X3 matrix except already marked slot.</td></tr>
<tr><td><b>Post Condition</b></td><td>If any player first make horizontal, vertical or diagonal pattern with same mark then It will be winner.

It will print <b>'Player-(1/2) won! Congratulations!'</b></td></tr>
<tr><td><b>Output Screenshots</b></td><td>

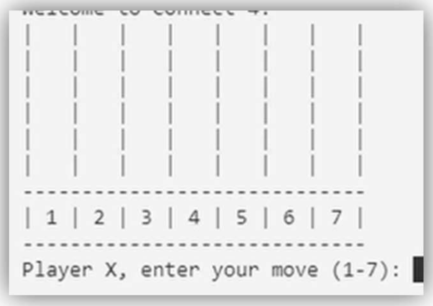
```
It's player 2's turn. Enter your slot: 6
 O | O | X
---|---|---
 X | O | X
---|---|---
 7 | 8 | 9
It's player 1's turn. Enter your slot: 9
 O | O | X
---|---|---
 X | O | X
---|---|---
 7 | 8 | O
Player 1 won! Congratulations!
PS F:\CollegePractical\Internpe>
```

Fig. 3.4 Task-3 Output</td></tr>
</table>

Table 3.4 Task-4

<table>
<tr><th colspan="2">Task 4</th></tr>
<tr><td><b>Task Description</b></td><td>Make a CONNECT FOUR GAME With the help of C++
After that make a separate Video on it.</td></tr>
<tr><td><b>Trigger</b></td><td>Compile code and run on terminal.</td></tr>
<tr><td><b>Basic Path</b></td><td>

There is 6 rows and 7 columns in Game board.



```
Welcome to Connect 4!
|   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |
-----------------------------
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
-----------------------------
Player X, enter your move (1-7):
```

Fig. 3.5 Task-4 Starting of Game

1. Player-X has to first select it's move (column (1-7)) form game board.

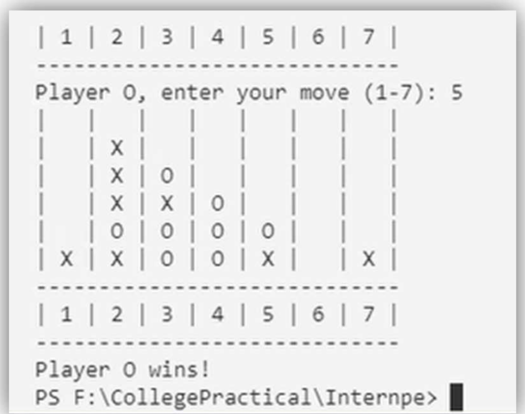
2. Player-0 has second select it's move as Player-X.

</td></tr>
<tr><td><b>Post Condition</b></td><td>

If any player first make horizontal, vertical or diagonal pattern with same mark then It will be winner.

It will print **'Player-(X/0) wins!'**

</td></tr>
<tr><td><b>Output Screenshots</b></td><td>



```
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
-----------------------------
Player O, enter your move (1-7): 5
|   |   |   |   |   |   |   |
|   | X |   |   |   |   |   |
|   | X | O |   |   |   |   |
|   | X | X | O |   |   |   |
|   | O | O | O | O |   |   |
| X | X | O | O | X |   | X |
-----------------------------
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
-----------------------------
Player O wins!
PS F:\CollegePractical\Internpe>
```

Fig. 3.6 Task-4 Output

</td></tr>
</table>

CHAPTER 4 LEARNING EXPERIENCES

4.1 KNOWLEDGE ACQUIRED/SKILLS LEARNT

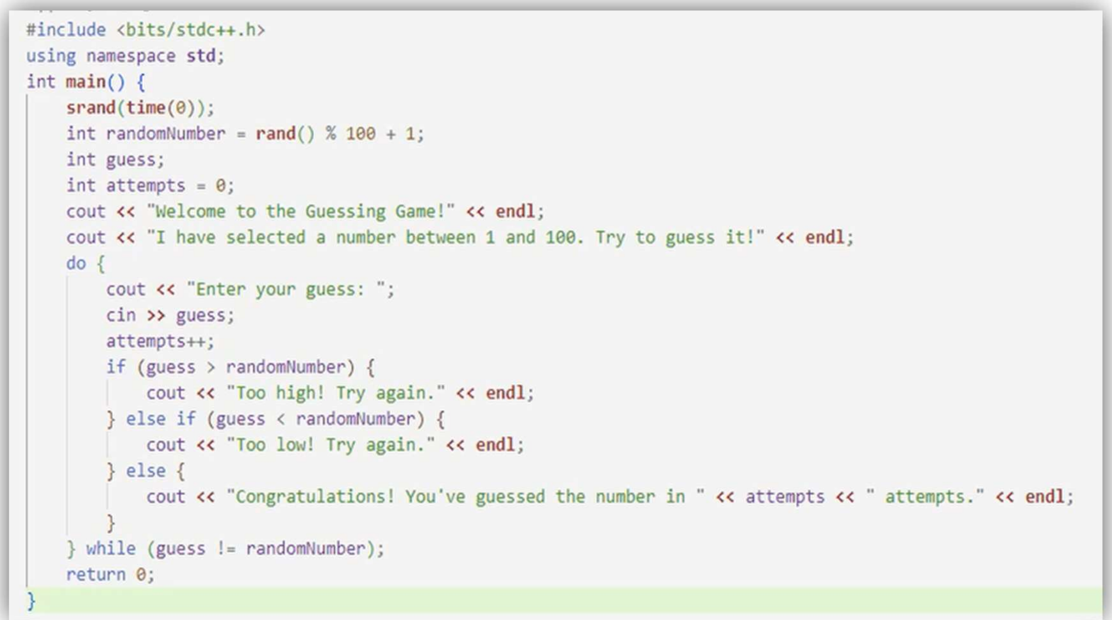
During my internship at InternPe, I had the opportunity to develop games, including a guessing game, tic-tac-toe, rock-paper-scissors, and connect 4, which helped me acquire and enhance the following skills:

1. **Algorithm Design**:
   - Designed and implemented algorithms for game logic and functionality.
   - Practiced sorting, searching, and optimization techniques to improve game performance.
2. **Problem-Solving**:
   - Applied critical thinking to develop efficient solutions for game-related problems.
   - Enhanced my ability to break down complex tasks into manageable steps.
3. **C++ Programming**:
   - Strengthened my proficiency in C++ by writing clean, efficient, and well-documented code.
   - Gained experience with C++ features such as classes, inheritance, polymorphism, and templates.
4. **Game Development**:
   - Developed interactive games (guessing game, tic-tac-toe, rock-paper-scissors, and connect 4).
   - Learned to implement game mechanics, handle user input, and manage game states.
5. **Debugging and Testing**:
   - Improved my debugging skills by identifying and fixing issues in game code.
   - Conducted thorough testing to ensure the games functioned correctly and were free of bugs.
6. **Project Management**:
   - Managed time effectively to meet project deadlines and deliverables.
   - Learned to document code and write comprehensive reports on project progress.

This internship provided me with a solid foundation in data structures and algorithms using C++, honed my problem-solving skills, and gave me practical experience in game development. These skills will be invaluable as I pursue a career in software development.

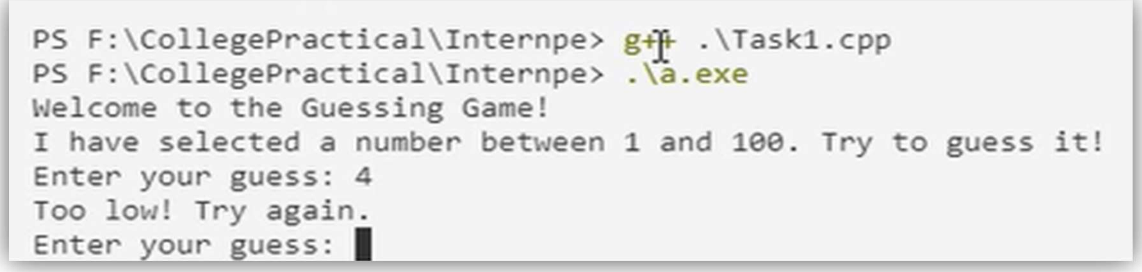
4.2 INDUSTRY PRACTICES ADAPTED

Task-1



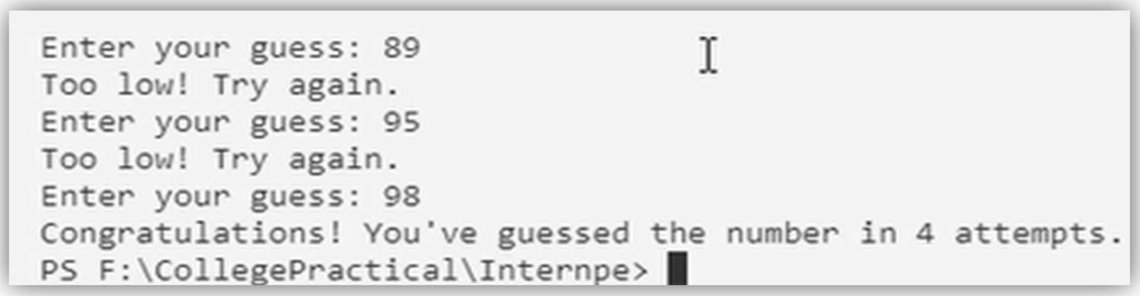
```
#include <bits/stdc++.h>
using namespace std;
int main() {
    srand(time(0));
    int randomNumber = rand() % 100 + 1;
    int guess;
    int attempts = 0;
    cout << "Welcome to the Guessing Game!" << endl;
    cout << "I have selected a number between 1 and 100. Try to guess it!" << endl;
    do {
        cout << "Enter your guess: ";
        cin >> guess;
        attempts++;
        if (guess > randomNumber) {
            cout << "Too high! Try again." << endl;
        } else if (guess < randomNumber) {
            cout << "Too low! Try again." << endl;
        } else {
            cout << "Congratulations! You've guessed the number in " << attempts << " attempts." << endl;
        }
    } while (guess != randomNumber);
    return 0;
}
```

Fig. 4.1 Task-1 Code



```
PS F:\CollegePractical\Internpe> g++ .\Task1.cpp
PS F:\CollegePractical\Internpe> .\a.exe
Welcome to the Guessing Game!
I have selected a number between 1 and 100. Try to guess it!
Enter your guess: 4
Too low! Try again.
Enter your guess:
```

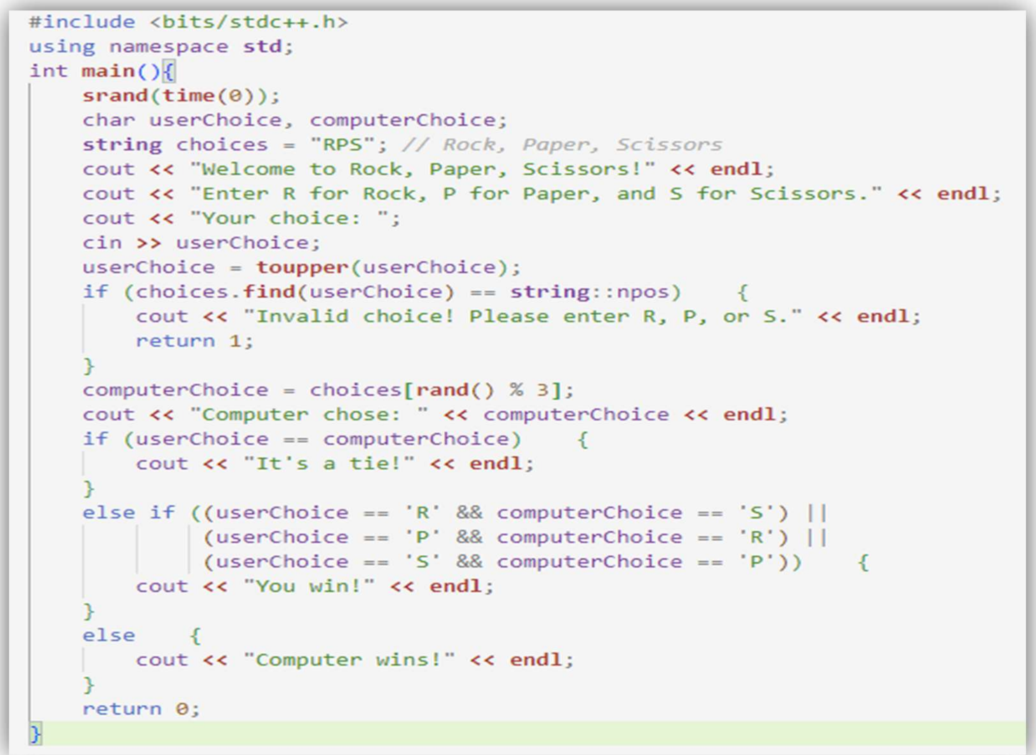
Fig. 4.2 Task-1 Output-1



```
Enter your guess: 89
Too low! Try again.
Enter your guess: 95
Too low! Try again.
Enter your guess: 98
Congratulations! You've guessed the number in 4 attempts.
PS F:\CollegePractical\Internpe>
```

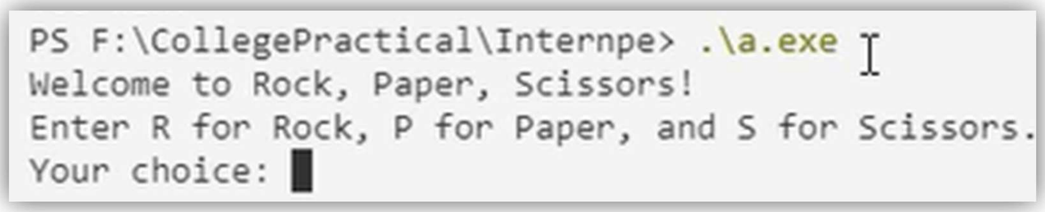
Fig. 4.3 Task-1 Output-2

Task-2



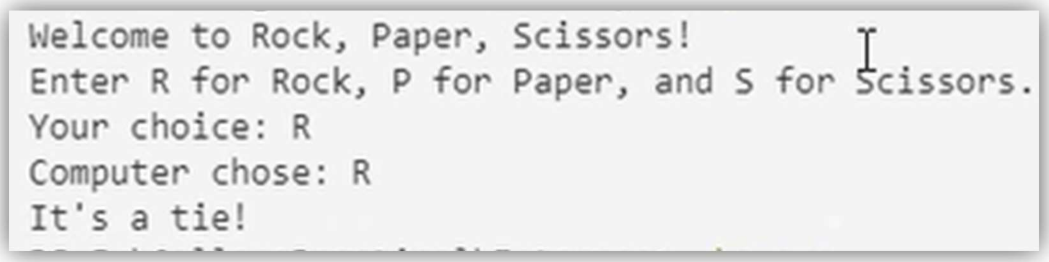
```
#include <bits/stdc++.h>
using namespace std;
int main(){
    srand(time(0));
    char userChoice, computerChoice;
    string choices = "RPS"; // Rock, Paper, Scissors
    cout << "Welcome to Rock, Paper, Scissors!" << endl;
    cout << "Enter R for Rock, P for Paper, and S for Scissors." << endl;
    cout << "Your choice: ";
    cin >> userChoice;
    userChoice = toupper(userChoice);
    if (choices.find(userChoice) == string::npos)    {
        cout << "Invalid choice! Please enter R, P, or S." << endl;
        return 1;
    }
    computerChoice = choices[rand() % 3];
    cout << "Computer chose: " << computerChoice << endl;
    if (userChoice == computerChoice)    {
        cout << "It's a tie!" << endl;
    }
    else if ((userChoice == 'R' && computerChoice == 'S') ||
             (userChoice == 'P' && computerChoice == 'R') ||
             (userChoice == 'S' && computerChoice == 'P'))    {
        cout << "You win!" << endl;
    }
    else    {
        cout << "Computer wins!" << endl;
    }
    return 0;
}
```

Fig. 4.4 Task-2 Code



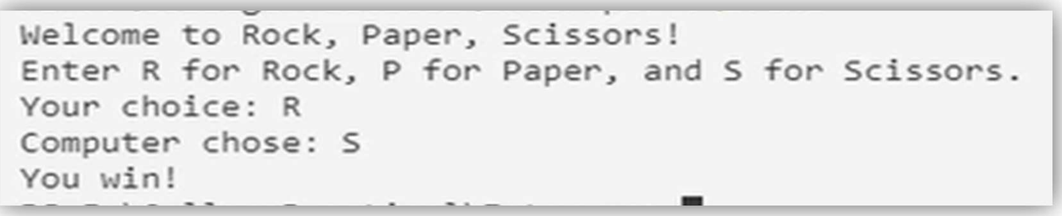
```
PS F:\CollegePractical\Internpe> .\a.exe
Welcome to Rock, Paper, Scissors!
Enter R for Rock, P for Paper, and S for Scissors.
Your choice: 
```

Fig. 4.5 Task-2 Output-1



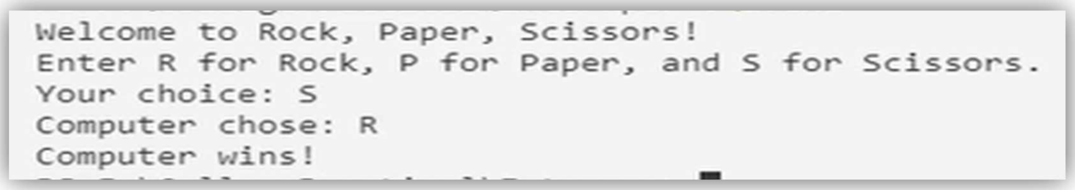
```
Welcome to Rock, Paper, Scissors!
Enter R for Rock, P for Paper, and S for Scissors.
Your choice: R
Computer chose: R
It's a tie!
```

Fig. 4.6 Task-2 Output-2



```
Welcome to Rock, Paper, Scissors!
Enter R for Rock, P for Paper, and S for Scissors.
Your choice: R
Computer chose: S
You win!
```

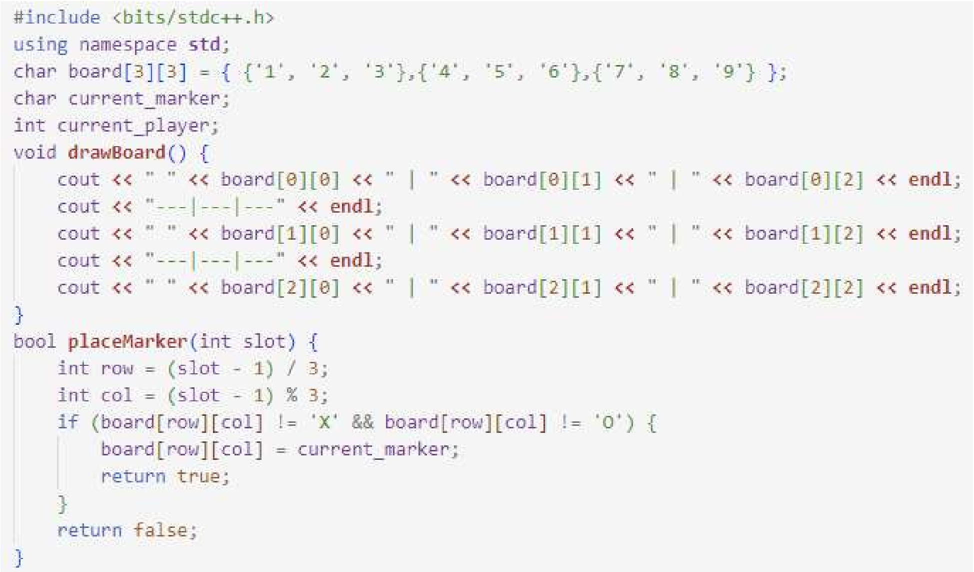
Fig. 4.7 Task-2 Output-3



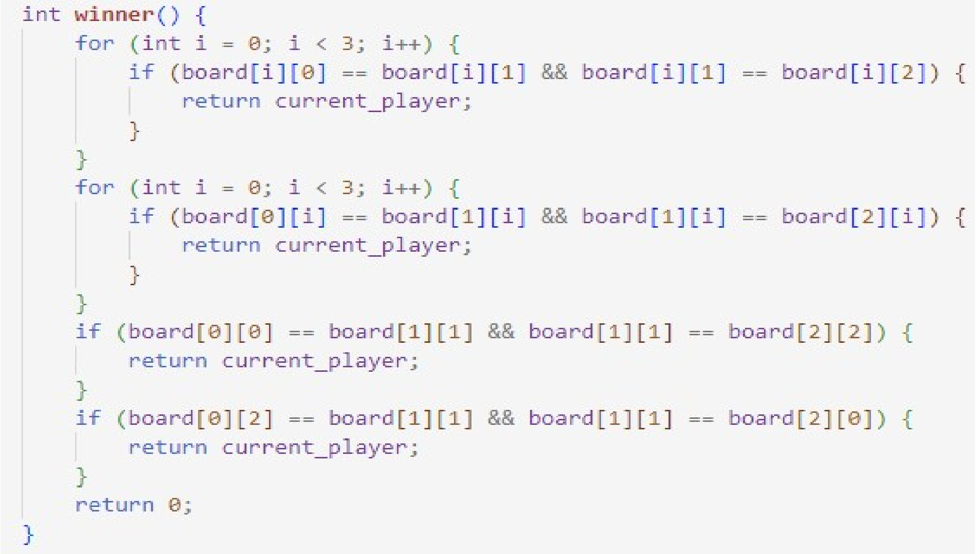
```
Welcome to Rock, Paper, Scissors!
Enter R for Rock, P for Paper, and S for Scissors.
Your choice: S
Computer chose: R
Computer wins!
```

Fig. 4.8 Task-2 Output-4

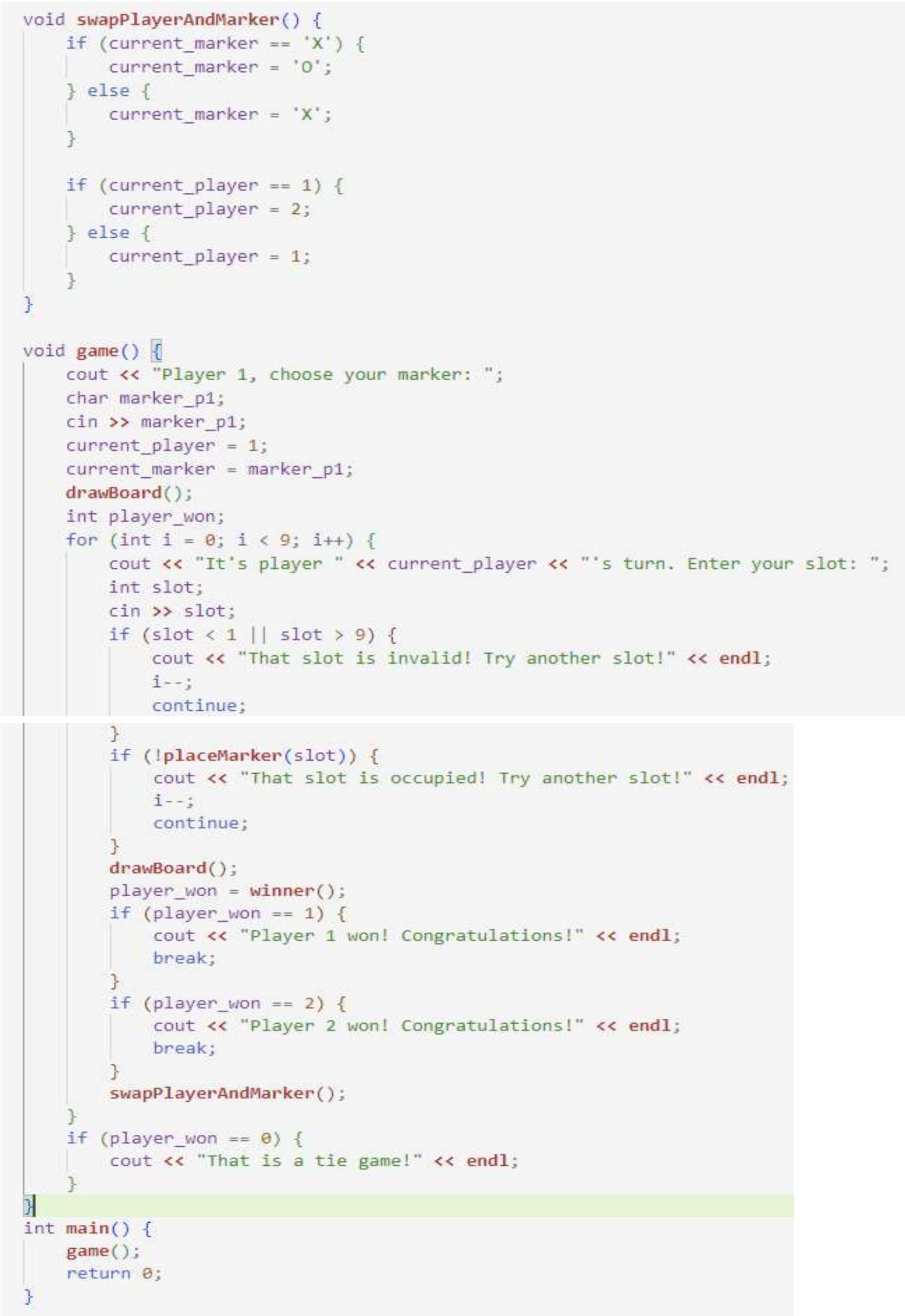
Task-3



```cpp
#include <bits/stdc++.h>
using namespace std;
char board[3][3] = { {'1', '2', '3'},{'4', '5', '6'},{'7', '8', '9'} };
char current_marker;
int current_player;
void drawBoard() {
    cout << " " << board[0][0] << " | " << board[0][1] << " | " << board[0][2] << endl;
    cout << "---|---|---" << endl;
    cout << " " << board[1][0] << " | " << board[1][1] << " | " << board[1][2] << endl;
    cout << "---|---|---" << endl;
    cout << " " << board[2][0] << " | " << board[2][1] << " | " << board[2][2] << endl;
}
bool placeMarker(int slot) {
    int row = (slot - 1) / 3;
    int col = (slot - 1) % 3;
    if (board[row][col] != 'X' && board[row][col] != 'O') {
        board[row][col] = current_marker;
        return true;
    }
    return false;
}
```

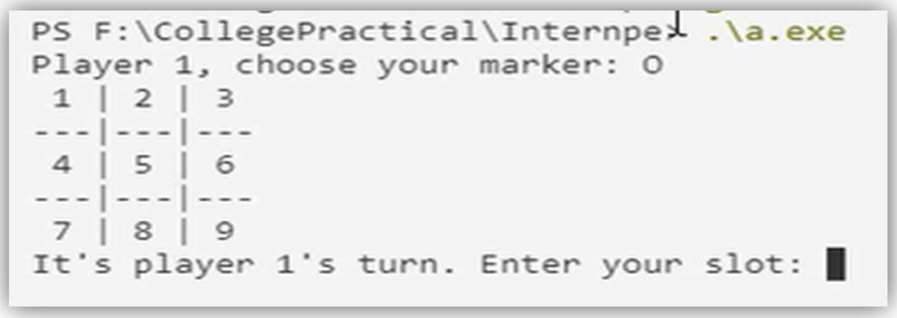


```cpp
int winner() {
    for (int i = 0; i < 3; i++) {
        if (board[i][0] == board[i][1] && board[i][1] == board[i][2]) {
            return current_player;
        }
    }
    for (int i = 0; i < 3; i++) {
        if (board[0][i] == board[1][i] && board[1][i] == board[2][i]) {
            return current_player;
        }
    }
    if (board[0][0] == board[1][1] && board[1][1] == board[2][2]) {
        return current_player;
    }
    if (board[0][2] == board[1][1] && board[1][1] == board[2][0]) {
        return current_player;
    }
    return 0;
}
```



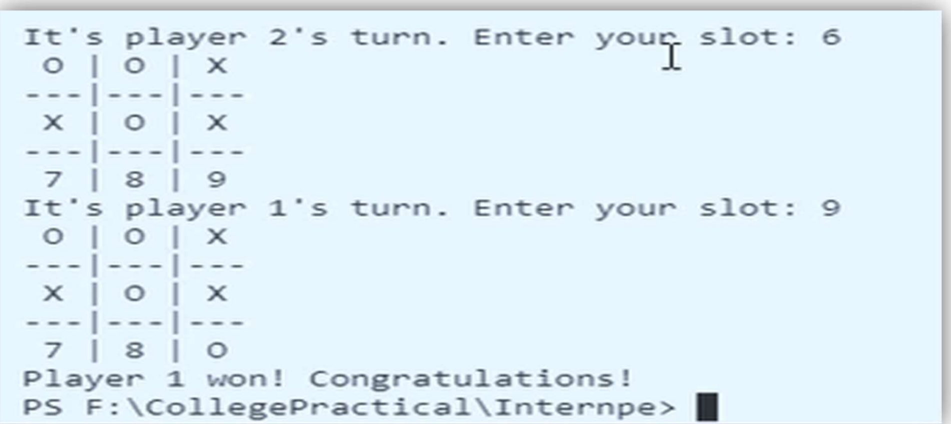
```
void swapPlayerAndMarker() {
    if (current_marker == 'X') {
        current_marker = 'O';
    } else {
        current_marker = 'X';
    }

    if (current_player == 1) {
        current_player = 2;
    } else {
        current_player = 1;
    }
}

void game() {
    cout << "Player 1, choose your marker: ";
    char marker_p1;
    cin >> marker_p1;
    current_player = 1;
    current_marker = marker_p1;
    drawBoard();
    int player_won;
    for (int i = 0; i < 9; i++) {
        cout << "It's player " << current_player << "'s turn. Enter your slot: ";
        int slot;
        cin >> slot;
        if (slot < 1 || slot > 9) {
            cout << "That slot is invalid! Try another slot!" << endl;
            i--;
            continue;
        }
        if (!placeMarker(slot)) {
            cout << "That slot is occupied! Try another slot!" << endl;
            i--;
            continue;
        }
        drawBoard();
        player_won = winner();
        if (player_won == 1) {
            cout << "Player 1 won! Congratulations!" << endl;
            break;
        }
        if (player_won == 2) {
            cout << "Player 2 won! Congratulations!" << endl;
            break;
        }
        swapPlayerAndMarker();
    }
    if (player_won == 0) {
        cout << "That is a tie game!" << endl;
    }
}
int main() {
    game();
    return 0;
}
```

Fig. 4.9 Task-3 Code



```
PS F:\CollegePractical\Internpe> .\a.exe
Player 1, choose your marker: O
 1 | 2 | 3
---|---|---
 4 | 5 | 6
---|---|---
 7 | 8 | 9
It's player 1's turn. Enter your slot:
```

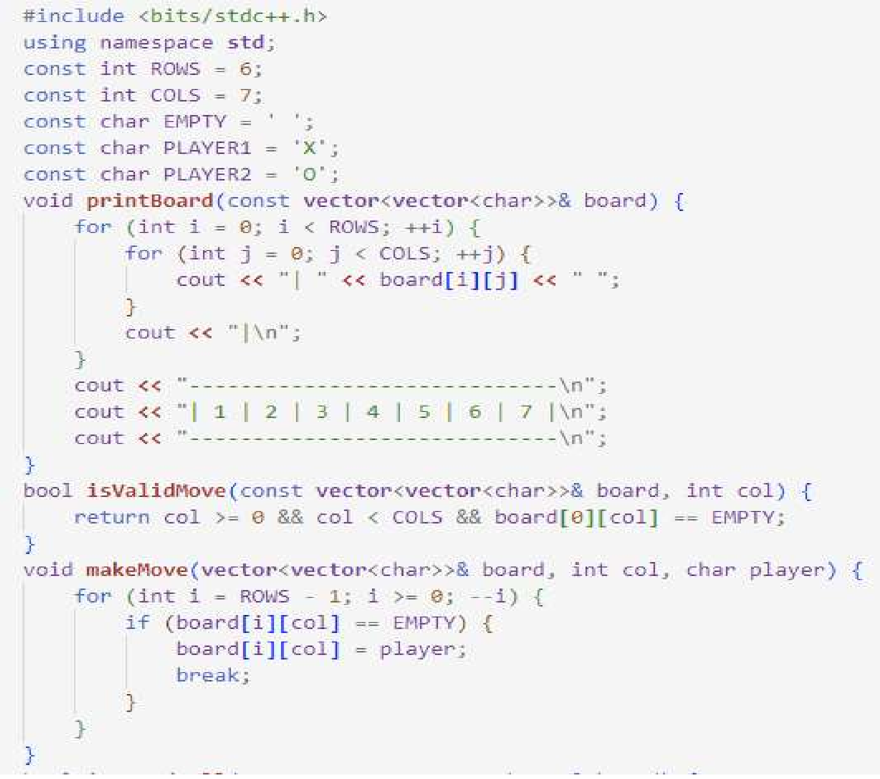
Fig. 4.10 Task-3 Output-1



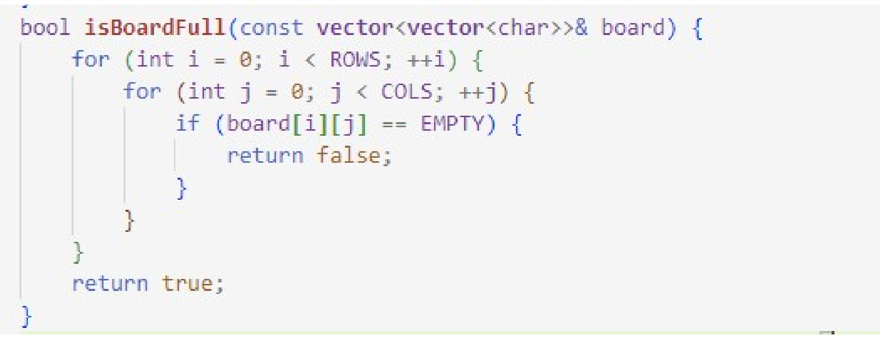
```
It's player 2's turn. Enter your slot: 6
 O | O | X
---|---|---
 X | O | X
---|---|---
 7 | 8 | 9
It's player 1's turn. Enter your slot: 9
 O | O | X
---|---|---
 X | O | X
---|---|---
 7 | 8 | O
Player 1 won! Congratulations!
PS F:\CollegePractical\Internpe>
```

Fig. 4.11 Task-3 Output-2

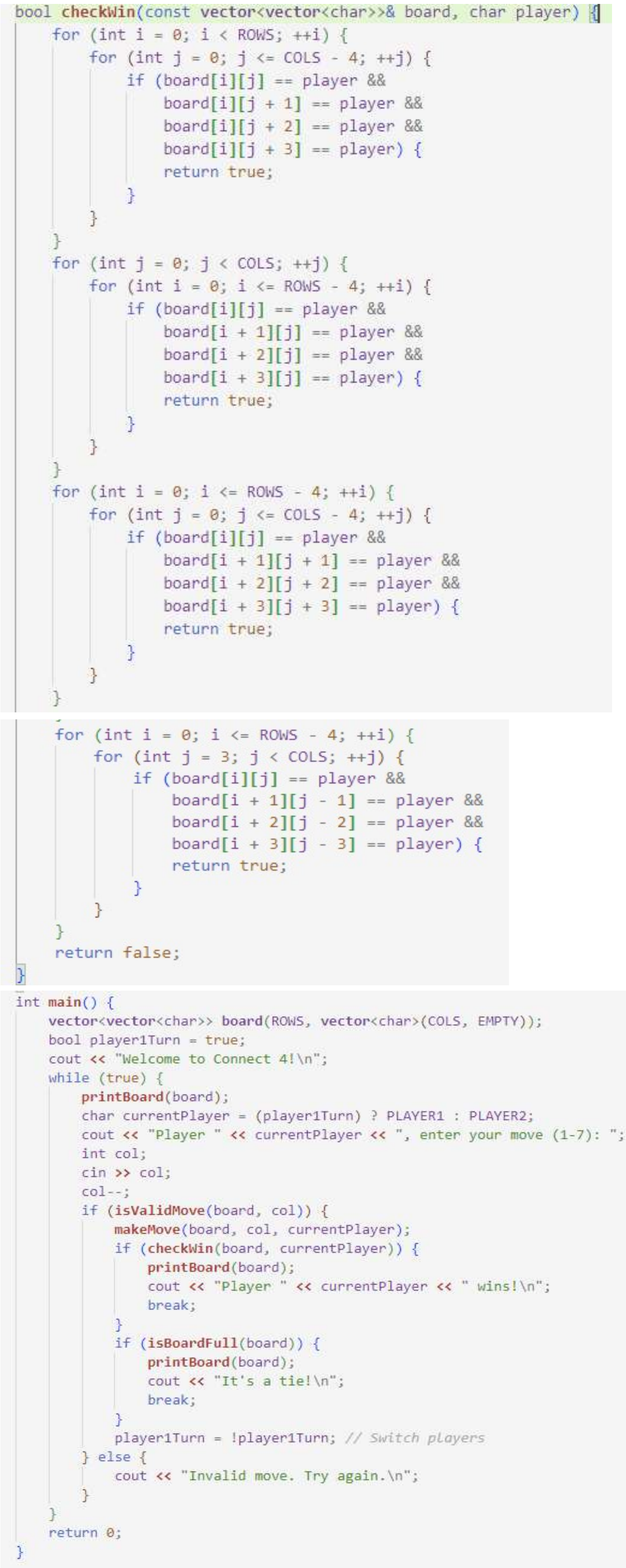
Task-4



```cpp
#include <bits/stdc++.h>
using namespace std;
const int ROWS = 6;
const int COLS = 7;
const char EMPTY = ' ';
const char PLAYER1 = 'X';
const char PLAYER2 = 'O';
void printBoard(const vector<vector<char>>& board) {
    for (int i = 0; i < ROWS; ++i) {
        for (int j = 0; j < COLS; ++j) {
            cout << "| " << board[i][j] << " ";
        }
        cout << "|\n";
    }
    cout << "-----------------------------\n";
    cout << "| 1 | 2 | 3 | 4 | 5 | 6 | 7 |\n";
    cout << "-----------------------------\n";
}
bool isValidMove(const vector<vector<char>>& board, int col) {
    return col >= 0 && col < COLS && board[0][col] == EMPTY;
}
void makeMove(vector<vector<char>>& board, int col, char player) {
    for (int i = ROWS - 1; i >= 0; --i) {
        if (board[i][col] == EMPTY) {
            board[i][col] = player;
            break;
        }
    }
}
```



```cpp
bool isBoardFull(const vector<vector<char>>& board) {
    for (int i = 0; i < ROWS; ++i) {
        for (int j = 0; j < COLS; ++j) {
            if (board[i][j] == EMPTY) {
                return false;
            }
        }
    }
    return true;
}
```

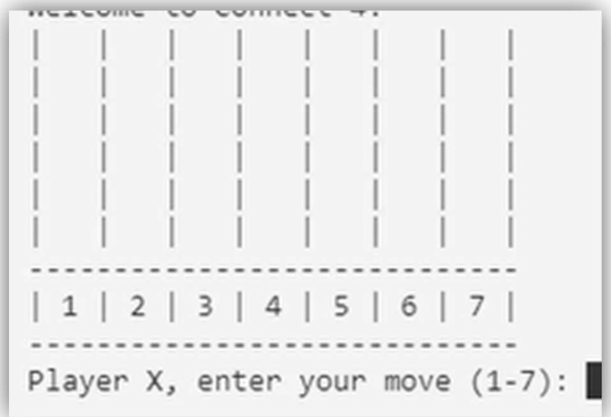


```
bool checkWin(const vector<vector<char>>& board, char player) {
    for (int i = 0; i < ROWS; ++i) {
        for (int j = 0; j <= COLS - 4; ++j) {
            if (board[i][j] == player &&
                board[i][j + 1] == player &&
                board[i][j + 2] == player &&
                board[i][j + 3] == player) {
                return true;
            }
        }
    }
    for (int j = 0; j < COLS; ++j) {
        for (int i = 0; i <= ROWS - 4; ++i) {
            if (board[i][j] == player &&
                board[i + 1][j] == player &&
                board[i + 2][j] == player &&
                board[i + 3][j] == player) {
                return true;
            }
        }
    }
    for (int i = 0; i <= ROWS - 4; ++i) {
        for (int j = 0; j <= COLS - 4; ++j) {
            if (board[i][j] == player &&
                board[i + 1][j + 1] == player &&
                board[i + 2][j + 2] == player &&
                board[i + 3][j + 3] == player) {
                return true;
            }
        }
    }
```

```
    for (int i = 0; i <= ROWS - 4; ++i) {
        for (int j = 3; j < COLS; ++j) {
            if (board[i][j] == player &&
                board[i + 1][j - 1] == player &&
                board[i + 2][j - 2] == player &&
                board[i + 3][j - 3] == player) {
                return true;
            }
        }
    }
    return false;
}
```

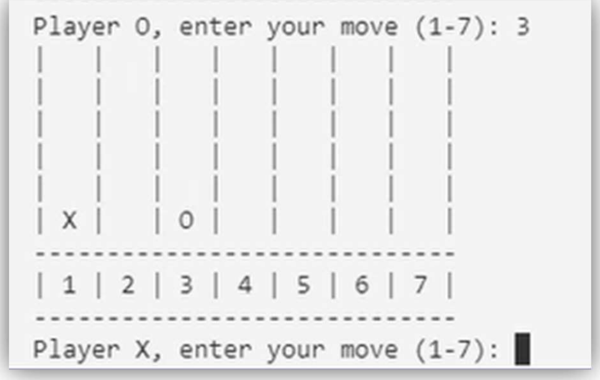
```
int main() {
    vector<vector<char>> board(ROWS, vector<char>(COLS, EMPTY));
    bool player1Turn = true;
    cout << "Welcome to Connect 4!\n";
    while (true) {
        printBoard(board);
        char currentPlayer = (player1Turn) ? PLAYER1 : PLAYER2;
        cout << "Player " << currentPlayer << ", enter your move (1-7): ";
        int col;
        cin >> col;
        col--;
        if (isValidMove(board, col)) {
            makeMove(board, col, currentPlayer);
            if (checkWin(board, currentPlayer)) {
                printBoard(board);
                cout << "Player " << currentPlayer << " wins!\n";
                break;
            }
            if (isBoardFull(board)) {
                printBoard(board);
                cout << "It's a tie!\n";
                break;
            }
            player1Turn = !player1Turn; // Switch players
        } else {
            cout << "Invalid move. Try again.\n";
        }
    }
    return 0;
}
```

Fig. 4.12 Task-4 Code



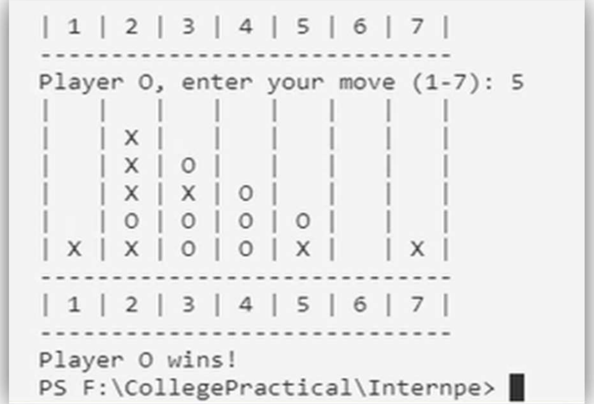
```
Welcome to Connect 4!
|   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |
-----------------------------
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
-----------------------------
Player X, enter your move (1-7):
```

Fig. 4.13 Task-4 Output-1



```
Player O, enter your move (1-7): 3
|   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |
| X |   | O |   |   |   |   |
-----------------------------
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
-----------------------------
Player X, enter your move (1-7):
```

Fig. 4.14 Task-4 Output-2



```
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
-----------------------------
Player O, enter your move (1-7): 5
|   |   |   |   |   |   |   |
|   | X |   |   |   |   |   |
|   | X | O |   |   |   |   |
|   | X | X | O |   |   |   |
|   | O | O | O | O |   |   |
| X | X | O | O | X |   | X |
-----------------------------
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
-----------------------------
Player O wins!
PS F:\CollegePractical\Internpe>
```

Fig. 4.15 Task-4 Output-3

4.3 REALTIME APPLICABILITY OF TECHNOLOGIES LEARNT

During my internship at InternPe, I developed a solid foundation in Data Structures and Algorithms (DSA) using C++ through game development projects, including a guessing game, tic-tac-toe, rock-paper-scissors, and connect 4. The skills and technologies I learned have broad real-time applicability in the software development industry:

1. **Algorithm Optimization**:
   - **Applications**: Optimized algorithms are crucial for enhancing the performance of applications in fields such as finance, healthcare, and e-commerce.
   - **Example**: Implementing efficient sorting and searching algorithms in e-commerce platforms to handle large volumes of product data and provide quick search results.
2. **Game Development**:
   - **Applications**: The game development experience translates into the ability to create interactive and engaging applications, which is valuable in the entertainment, education, and simulation industries.
   - **Example**: Developing educational games and simulations for training purposes in various fields, including healthcare and aviation.
3. **Problem-Solving Skills**:
   - **Applications**: Strong problem-solving skills are vital in software development, allowing developers to identify and solve complex issues efficiently.
   - **Example**: Troubleshooting and debugging software in real-time systems, ensuring reliability and performance in critical applications such as medical devices and aerospace systems.
4. **Code Efficiency and Performance**:
   - **Applications**: Writing clean and efficient code is crucial for developing high-performance applications, particularly in embedded systems, real-time systems, and mobile applications.
   - **Example**: Optimizing code for mobile applications to ensure they run smoothly on devices with limited resources.

The technologies and skills acquired during the DSA (C++) internship at InternPe have provided me with a comprehensive toolkit that is highly applicable across various domains in the software industry, ensuring I am well-prepared to tackle real-world challenges in my future career.

CHAPTER 5 CONCLUSION

The DSA (C++) internship at InternPe has been an invaluable experience, providing me with a strong foundation in data structures and algorithms, enhanced problem-solving abilities, and practical skills through game development projects. The knowledge and skills gained, including efficient data management, algorithm optimization, and collaboration, are highly applicable across various domains in the software industry. This internship has significantly prepared me for real-world challenges and future roles in software development.

REFERENCES

Task-1 : https://www.youtube.com/watch?v=HTv-TgYaXI8&authuser=0

Task-2 : https://www.youtube.com/watch?v=HTv-TgYaXI8&authuser=0

Task-4 :

https://drive.google.com/file/d/1MJNl2PVAofxQXlyZU73OJersHQpVtvFH/view?usp=classroom_web&authuser=0