TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING
PULCHOWK CAMPUS

A PROJECT REPORT ON

APPLICATION OF OBJECT ORIENTED PROGRAMMING IN BAGHCHAL USING C++

By:
Subham Shrestha (077/BCT/082)
Sudip Tiwari (077/BCT/085)
Utsav Manandhar (077/BCT/093)

A PROJECT REPORT TO THE DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING ON OBJECT ORIENTED PROGRAMMING APPLICATION USING C++

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
LALITPUR, NEPAL

August, 2022

ACKNOWLEDGEMENT

First and foremost, we would like to express our sincere gratitude towards our lecturer, Daya Sagar Baral for his insightful lectures, constant guidance and helpful encouragement.

We would like to thank the Department of Electronics and Computer Engineering, Pulchowk Campus, Institute of Engineering for providing us opportunity of collaborative undertaking which has helped us to implement the knowledge as project for second year, and developing project of our own which will greatly enhance our knowledge and provide us a new experience of teamwork.

We are also grateful towards our respected seniors who have helped us with their knowledge, experience and suggestions. We would also like to thank all of our friends who have directly and indirectly helped us in doing this project. Last but not the least, we place a deep sense of appreciation to our family members who have been constant source of inspiration for us.

Any kind of suggestion or criticism will be highly appreciated and acknowledged.

Authors:
Subham Shrestha
Sudip Tiwari
Utsav Manandhar

ABSTRACT

The main aim of this project was to develop game program using an Object Oriented Programming language, C++. For this project, we made a classic two player Baghchal game using Simple and Fast Multimedia Library(SFML) for graphical interface.The goal of creating this game is also to learn about game development.

Baghchal is a strategic, two-player board game that originated in Nepal. The game is asymmetric in that one player controls four tigers and the other player controls up to twenty goats. The tigers 'hunt' the goats while the goats attempt to block the tigers' movements. This game has variants with a different board, but the rules are the same.

Keywords: Baghchal, SFML, OOP, C++

TABLE OF CONTENTS

1. OBJECTIVES

This project does not aim to advance our Game development skills within a matter of days or weeks. Instead, the major objective of this project is to encourage us to implement object oriented approach to think, model and develop basic algorithms into program code. The main objectives of this project can be summed us as:

- To understand Object Oriented Programming paradigm and build a project using it. C++ is preferred over other languages like Python because C++ offers better efficiency and speed and is suitable for almost every platform including embedded systems whereas Python can be used only on certain platforms that support high-level languages.
- To explore the basic attributes of C++ programming language.
- To learn about resource re-usability by building user defined header files.
- To be familiar with software development library designed to provide a simple Application Programming Interface (API) to various multimedia components in computers using SFML library.
- To develop effective and efficient program by optimizing time and space constraints.
- To learn the fundamental concepts about game development.
- To acquire teamwork and communication skills as a result of working as a team.

2. INTRODUCTION

Baghchal is a recreational turn based game played by two players on a 5x5 point grid with one player controlling tiger pieces and the other player controlling goat pieces.

The game Baghchal is asymmetric in that the tiger player controls four tigers and the goat player controls up to twenty goats. In the first phase of the game, four tigers are placed on each corner of the board and the goat player places their goats on any empty point on the grid on their turn. The tiger player can move one of their pieces to a valid point on their turn. A valid point is any point that is empty and either adjacent to the tiger or one step in the direction of an adjacent goat. In the second phase, after all the goats are placed, the goats move the same as tigers,

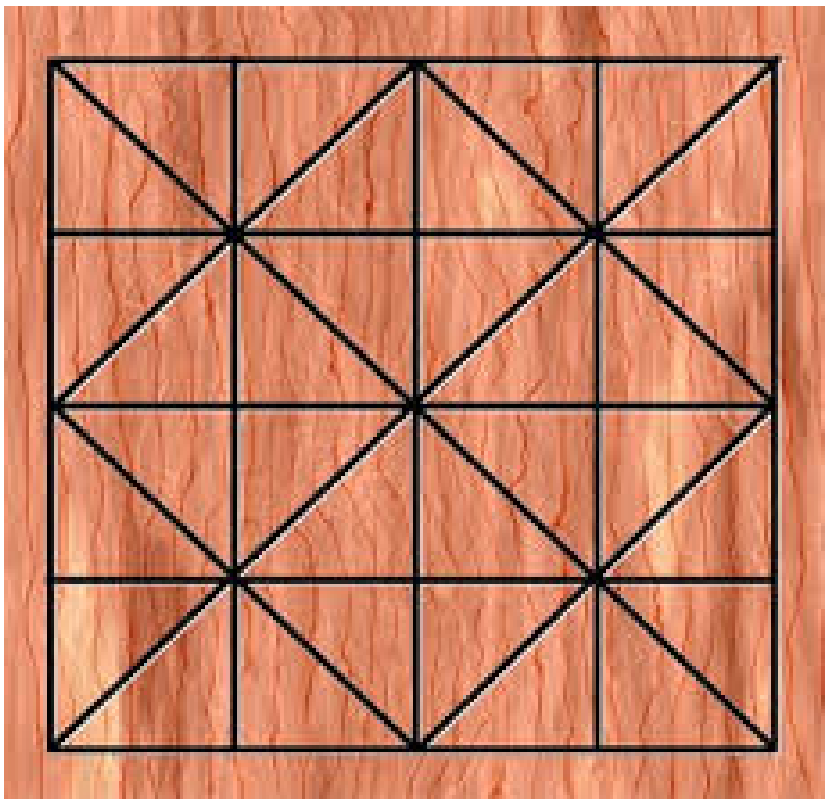


Figure 2.1: Baghchal Game Board

A tiger can ‘eat’ a goat if it jumps over the goat to an empty tile. The objective of the tiger player is to ’eat’ all the goats and that of the goat player is to block all possible moves of the tiger player.

2.1. Theoretical Background On Object Oriented Programming, C++ and SFML

Object-oriented programming (OOP) is a computer programming model that organizes software design around data, or objects, rather than functions and logic. An object can be defined as a data field that has unique attributes and behavior.OOP focuses on the objects that developers want to manipulate rather than the logic required to manipulate them. C++ is a general-purpose programming language that was developed as an enhancement of the C language to include object-oriented paradigm. It is an imperative and a compiled language.

C++ is a middle-level language rendering it the advantage of programming low-level (drivers, kernels) and even higher-level applications (games, GUI, desktop apps etc.). The basic syntax and code structure of both C and C++ are the same.

Applications of C++:
Operating Systems Systems Programming. e.g. Linux-based OS (Ubuntu etc.)
Browsers (Chrome Firefox)
Graphics Game engines(Photoshop, Blender, Unreal-Engine)
Database Engines (MySQL, MongoDB, Redis etc.)
Cloud/Distributed Systems

Simple and Fast Multimedia Library (SFML) is a cross-platform software development library designed to provide a simple application programming interface (API) to various multimedia components in computers. SFML provides a simple interface to the various components of the PC, to ease the development of games and multimedia applications. It is composed of five modules: system, window, graphics, audio and network. With SFML, the application can be compiled and run out of the box on the most common operating systems: Windows, Linux, MacOS and soon Android iOS. SFML has official bindings for the C and .Net languages. And thanks to its active community, it is also available in many other languages such as Java, Ruby, Python, Go, and more.

2.2. OBJECT ORIENTED FEATURES

The primary objective of the project was to learn object oriented programming concepts by practically implementing them. So, we have tried our best to implement the features of object oriented programming in our project.

2.3. Objects and Classes

As a basic building block of OOP, it is common to include these concepts in our program. Since game elements can be treated as objects, the classes Bagh and Bakhraa, which in turn are derived from the base class Animal are instantiated into various objects array that act as pieces in the board. Each piece is itself an object of their respective class such that each piece all 24 of have their on respective state data ehich can be easily accessed. There are other minor class included like Path class and Gameover class.

2.4. Abstraction

Abstraction is a feature of object oriented programming that hides the internal details of how object does its work and only provides the interface to use the service. We can manage complexity through abstraction. In OOP, classes are used for creating user-defined data for abstraction. When data and its operation are presented together, it is called ADT (Abstract Data Type). Hence, a class is an implementation of abstract data type. So, in OOP, classes are used in creating Abstract Data Type. For instance, we have created a class GameOver and made it available in the program. Now we can implement the class in creating objects and its manipulation without knowing its implementation. We have also used the class implementation of SFML available to us to create desired graphics images. While we know how to create and manipulate these objects, the exact details are hidden from us.

2.5. Encapsulation And Data Hiding

Combining data and function together into a single unit is called encapsulation. We can say encapsulation is a protective box that prevents the data from being accessed by other code that is defined outside the box. We can easily achieve abstraction by making use of encapsulation. We can achieve encapsulation through classes. In classes, each data or function is kept under an access specifier (private, protected or public).

1. **Public:** It contains data and functions that the external users of the class may know about.
2. **Private:** This section can only be accessed by code that is a member of a class.
3. **Protected:** This section is visible to class members, derived class members, friend functions of class, friend classes and friend classes of derived classes.

The insulation of data from direct access by the program is called data hiding. Data hiding by making them private or protected makes it safe from accidental alteration. Understanding this, we have tried to make our data private as much as it was possible.

2.6. Inheritance

Inheritance is a feature of OOP that helps in the reuse of code and eliminates the use of redundant code. It consequently creates a hierarchical system of classes between base classes and derived classes.

In our program, both the Bakhraa and Baagh pieces are very similar to one another. In fact, during the second phase of the game, their functionality is nearly identical. Hence we have created a base class 'Animal' from which two classes are derived 'Baagh' and 'Bakhraa'. Since the Animal class serves only to provide a template for the two classes, we have added pure virtual functions to it in order to make it an abstract class.

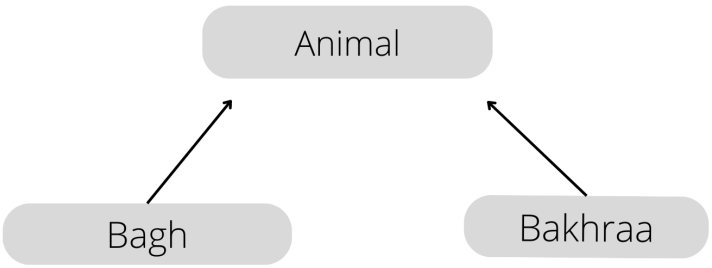
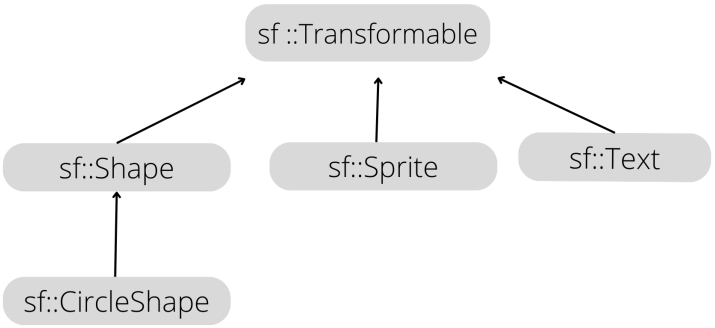


Fig: Hierarechical Inheritance

```
class Animal : public CircleShape
{
protected:
    int pos_x, pos_y;
public:
    void set_Position(int x, int y);
    int get_X();
    int get_Y();
    virtual void setstate(goatstate s) = 0;
```

Not only this, The Animal class is inherited from the SFML class 'CircleShape' which has its own class hierarchy.



2.7. Polymorphism

Polymorphism is another important feature of OOP. It allows different objects to respond to same operation in different ways. The different ways of using same function or operator depending on what they are operating on is called polymorphism. In c++, polymorphism is mainly divided into two types:

1. Compile time polymorphism
2. Run time polymorphism

Pure Virtual Functions: Pure virtual fuctions is a virtual function that only has a declaration but doesn't have a definition. Since they have no definition, these functions cannot be called and object consisting of pure virtual function cannot be created. It's usefulness comes from the fact that any class that derives from a base class consisting of a pure virtual function must implement the function for the derived class.

Abstract Base Class: A class that has a pure virtual function is an Abstract Base Class. These classes cannot be used to instantiate an object but serve the following function.

1. Deter from creation of the base class.
2. Act as a base class from any derived classs and allow for easy polymorphism

3. APPLICATION

Baghchal is one of the popular games of South Asian, especially Nepalese people. Most of us have already played this game at least once in our life using stones and lines drawn by the same stones in our childhood, preferably while taking our oxen to graze on pasture lands.

Initially, this project was not designed to be applied and compete in global market. However, with some efforts and modifications, following application area can be covered:

1. The Baghchal game can be played between users in real time when they are playing on the same computer screen.
2. The game serves are stress buster and also contributes in logical thinking and reasoning.

4. LITERATURE REVIEW

4.1. C++

C++ is a general purpose programming language that was developed by Danish computer scientist Bjarne Stroustrup as an extension of the C programming language.It is an intermediate level language, as it comprises a confirmation of both high level and low level language features.

It was originally created as a successor to C with more features, hence the name ”C++”. It is an Object Oriented Programming language but is not purely Object Oriented. It is also sometimes referred to as ”C with classes” due to its Object Oriented features and machine level control.

4.1.1. Structure of program

A general c++ program includes the following parts:

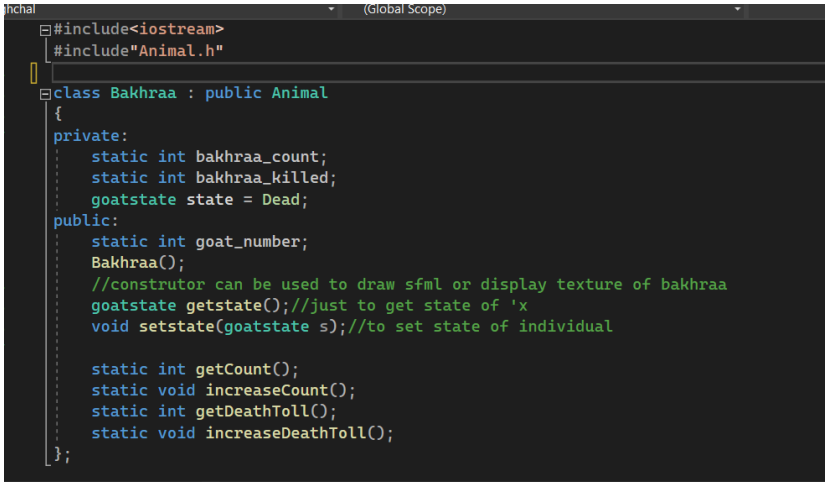
1. Standard Libraries Section
2. Class Definition Section
3. Functions Definition Section
4. Main Function Section

```cpp
#include <iostream>
#include<cmath>
using namespace std;
int sum(int x , int y);


int main()
{

    cout<<"The sum is "<<sum(5,3)<<endl;
    return 0;
}
int sum(int x, int y)
{
    return x+y;
}
```

Standard Libraries Section This section is written and the top of the program and consists of the first lines of code read by the compiler. This is also known as pre processor section. This section is where all the header files and namespaces used in the program such as ¡iostream¿ and std. This includes standard libraries as well as user-defined header files and programs.

Class Definition section The classes are used to map real world entities into programming. The classes are key building blocks of any object oriented program. A C++ program may include several class definitions.In our case, the classes are defined in separate header files that have been included in the Standard Libraries Section.

```cpp
#include "Animal.h"
#include "Bakhraa.h"
#include "Baagh.h"
```



Functions Definition Section Functions are a feature of Procedure Oriented Programming but are very useful in OOP as well. In the above program sum() is a function declared and used in the main function.

Main Function Section The main function is the main body of the program which the compiler executes first after all preprocessing. The main function is where all other functions of the program are called from. In C++, we write the main function such as an int return type such that it returns 0 if the program is successfully executed and 1 if there was an error in the execution.

4.2. SFML

Simple and Fast Multimedia Library (SFML) is a cross-platform software development library designed to provide a simple application programming interface (API) to various multimedia components in computers. It is written in C++ with bindings available for C++, Crystal, Java, Julia, .NET, Python, Ruby, and Rust.

SFML handles creating and input to window and also provides a graphics module for simple hardware acceleration of 2D computer graphics which includes text rendering and audio rendering. SFML consists of various modules such as:

1. System – vector and string classes, portable threading and timer facilities
2. Window – window and input device management

3. Graphics – hardware acceleration of 2D graphics including sprites, polygons and text rendering

4. Audio – hardware-accelerated spatialised audio playback and recording

5. Network – TCP and UDP network sockets, data encapsulation facilities, HTTP and FTP classes

5. EXISTING SYSTEM

The game Baghchal is hardly a novel concept. Baghchal is a game that predates modern computers and technology. Therefore it is no surprise that the game has been adapted to the computer game format numerous times even on commercial levels.

There are many similar applications already made on multiple platforms such as Windows, Android and IOS. As our sole purpose was learning, we have made a simpler versions of these established applications while adding some modifications of our own.

6. METHODOLOGY

To complete our destined project, we aimed to follow the given methods:

6.1. Initiating and planning

We initialized by planning and dividing the work among 3 of our members. We familiarized ourselves with the required library (SFML). Besides, we refreshed our knowledge and game logic of the game itself for the rules needed for Baghchal.

6.2. Algorithm Design

After getting the required rules and information, we designed the alogrithm and flowchart of the game project. A basic working model algorithm was designed for further tests, validation and continuity.

6.3. Software Design

With the basic set of algorithm in our hand, our focus then shifted to writing the code in Object Oriented paradigm. The project was carried out on C++ as its main skeletal and SFML as its graphics implementation since it was easy to learn and use. As for IDE and compiler, we opted for Visual Studio and Visual Studio Code as our IDE and Ming-W g++ as compiler in Windows operating system and GCC(GNU Compiler Collection) C++ Compiler(g++) in Linux.

6.4. Testing and debugging

We first developed Minimum Viable Product(MVP) sample of project for testing and debugging. Further testing and debugging were be done to add features and edit the project code as per our need and capability.

7. IMPLEMENTATION

Baghchal is one of the popular games of South Asian, especially Nepalese people. Most of us have already played this game at least once in our life using stones and lines drawn by the same stones in our childhood, preferably while taking our oxen to graze on pasture lands. Initially, this project was commenced with the idea of application of the Object Oriented approach and paradigm to develop a simple game of Baghchal. Since our objective was not targeted to build a full-fledged game to compete in global applications market. The full version of program is available in github: **https://github.com/Utsav-Manandhar/baghchal.git**

7.1. System Block Diagram

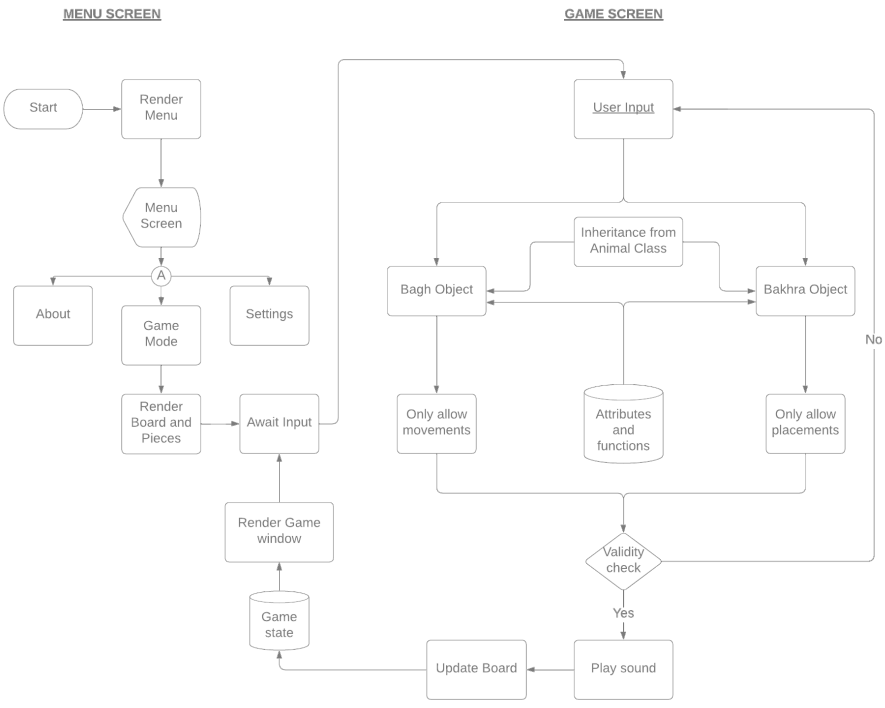


Figure : Block Diagram

7.2. Main Menu

The game starts with the main menu that consists of game title, Play Singleplayer button and Play Multiplayer button. **Multiplayer**: User can play in two player mode. One player plays as bakhraa and another plays as baagh. **Singleplayer**: User can play in single player mode. One player plays as **Bakhraa** and another plays as **Baagh**.

7.3. Game Board

The Board is a traditional Baghchal board. It comprises of a 5x5 grid where the players place and move their pieces. The pieces can move at most two tiles either laterally or diagonally along the lines of the board. The board is represented by a 5x5 two dimensional array named grid. Each element of this grid represents either an empty slot, Bakhraa piece or a tiger piece depending on its value.

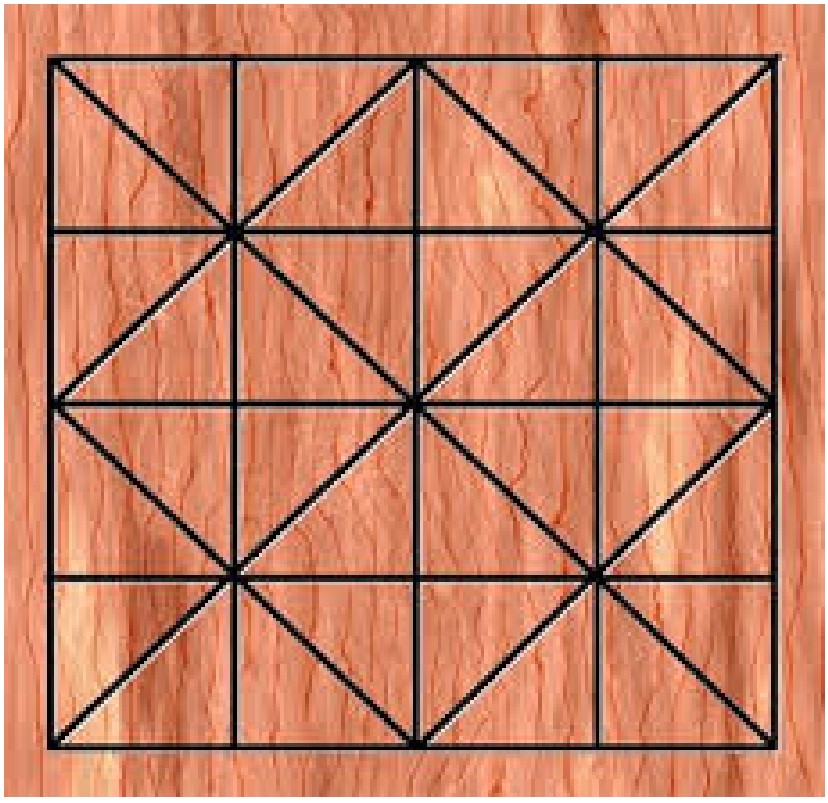


fig: Baghchal Game Board

7.4. Pieces

We have mainly two kind of pieces available namely Bagh and bakhraa. We have 4 fixed Baghs and 20 variable Bakhraas. Bagh number is intially fixed in the board but in case of Bakhraa, they should be placed by player as the game continues.

For both the pieces we have created an abstract class namely **Animal** and derived classes : **Bagh** and **Bakhraa**.Hierarchical inheritance is being used to derive the two piece classes.

7.5. Moves and Interactivity

Phase One In this phase of the game the Bakhraa player places their pieces at any empty location and the Baagh player moves their pieces to empty locations in response. Placing is done by simply clicking the space on which you wish to drop the piece and moving is done by dragging and dropping the desired piece. **Phase Two** This phase occurs after all 20 Bakhraas have been placed. At this point the Bakhraa player can no longer place Bakhraaas and must move the already placed Bakhraas. **Legal Moves**

1. Bakhraas can be placed in phase one in any empty space.
2. Both Baaghs and Bakhraas can move to an empty tile if there is a line on the board joining the two spaces,
3. A Baagh can capture or kill a Bakhraa if it ”jumps” over a tile containing a Bakhraa and if there is a straight line on the board joining the three spaces.

7.6. Win condition

Bagh win : Bagh wins if it can kill at least 5 Bakhraas. The killed number of Bakhraas is displayed in the board itself. If Bagh is able to kill at least 5 goats then the gameover function is invoked. Through the function the gameover display is rendered onto the screen.

Bakhraa win : Bakhraa wins if it is able to block all 4 Baghs. The Bakhraas cant move untill all 20 Bakhraas are placed onto the box. Bakhraa should block the Bagh moves before Bagh kills at least 5 Bakhraas. When all the Baghs are blocked and are not able to move then gameover function is invoked through which gameover display is rendered out in the screen.

8. RESULTS

With the completion of the project, the major objective of the project, learning Object Oriented Programming with C++ was achieved. The development and objective of the game is full-filled however still some additional features can be added to it to make it more interactive and user friendly.

From this project, we can take positives as we achieved the cooperation between the team members, learnt the basics of game progarmming and SFML modules. We got more familiariazed with version control with git and Github, documentation through latex and many more.

The following screenshots from different states throughout the game illustrate the final result of the project:

8.1. Main Menu

fig: Main Menu Screen of the game

8.2. Game Board

fig: Game Board along with simple UI/UX elements

8.3. Placing pieces

fig: Random board situation in the middle of the game

8.4. Baagh killing Bakhra

fig: Random board situation when a Baagh is in a favourable position to kill a Bakhraa

8.5. Baagh win

fig: Game over screen after Baagh wins the game

8.6. Bakhraa win

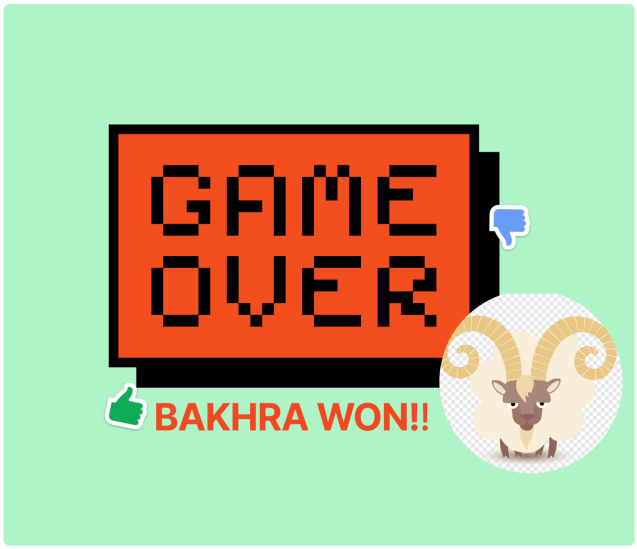


fig: Game over screen after Bakhraa wins the game

9. PROBLEMS FACED AND SOLUTIONS

We faced a number of problems over the course of creating a Baghchal game.

9.1. Code partition

While the use of Object Oriented programming helped us in dividing the code into manageable chunks, navigating between certain parts of the code and finding specific lines were slightly tedious due to the sheer volume of the code.

One thing we did to mitigate this problem is utilize the 'commenting' feature of programming languages. Subsections of the code are labeled and clear descriptions are added in order to properly identify a chunk of code or a function. Also, clear and descriptive variable names and function names are used to further improve readability of code.

9.2. Ghost pieces

Due to many simultaneously moving parts in our application, certain features weren't being executed as intended. A particularly annoying instance of this was the updating of the board when the goats were on their moving phase. Certain positions were being prematurely or unnecessarily updated. This was causing the board to be cluttered with invisible "ghost goats" that were interfering with normal gameplay. This issue was particularly hard to solve as the "ghost goats" didn't appear on the board but were present only internally, thus making them difficult to detect.

The problem was solved by adding additional checks to update the positions of pieces. This prevented the problematic code from being unintentionally executed and hence stopped the production of "ghost goats" and hence stopped undesired effects.

9.3. Unintentional movement

Baghchal consists of a simple but specific moveset for its pieces. However, during testing certain unintentional moves were enabled for the tiger piece that allowed it to move in an 'L' shape across the board.(Similar to the 'Knight' piece in chess)

This issue was solved rather easily by adding additional checks to the movement of the tiger piece.

10. LIMITATIONS AND FUTURE ENHANCEMENTS

Due to time limitation, and various examination schedule,we were unable to add some additional features which might enhance our program. There are many more features that can still be added to make the program more attractive, result oriented and useful.Some of the limitation include:

1. No undo move feature.

2. No online multiplayer via networking

3. Only one single style board

4. Lack of proper graphical interface

5. Dynamic screen size to make the program work in multiple screen sizes.

The possible future enhancements are as follows.

1. A multiplayer mode, in which two people from any place can play with each other via networking.

2. An overhaul in the way we store the state of the game with the help of stack to track moves. This would allow for move take backs and will make it so that we don't need to copy the state of the board again and again to make changes.

3. A better AI for single player, implementing Alpha-Beta pruning alongside Minimax algorithm.

4. Use of bitboard to represent the board

5. Board variation or newer board design

11. CONCLUSION AND RECOMMENDATIONS

The overall project is based on Object-Oriented programming in C++.This project has been developed using most of the features of OOP.After the completion of program, we have developed some basic knowledge about object oriented programming and Simple Fast Multimedia Library(SFML).

We also learned a great deal about collaboration and team work from the project.We learned to used various industry tools such as Github for version control of our project, LATEX for typesetting documents, Canva for making presentation and Lucidcharts for making flowcharts.

This project was unquestionably a good way of learning and implementing the way for programming practice. This project leads us to the winding up on the programming practice that for developing software a good judgment and proper analysis of the topic is required at first rather than the coding. The coding is not the initial step for emergent of any program, rather a good planning on the basic framework and making decision on the way of implementing the program is the most. After the coding of the program the system may not be as per our requirement but debugging, if any error, and testing and execution of the program is furthermore required. After the completion of the system, its management takes, is another most required obsession that is to be handled with great care. Thus, for developing either a larger of smaller program, proper decision should be made according to the occurrence of the situation.

Thus, after the completion of our project, it taught us a great deal about the development cycle, including planning, analysis, development, testing and debugging. We learned the power of Object Orientated Programming paradigm in making efficient software.

12. REFERENCES

12.1. Book References:

"The Secrets of Object oriented Programming", Daya Sagar Baral, Diwakar Baral

"The C++ Programming Language", Bjarne Stroustrup

"C++ How To Program" , Paul Deitel, Harvey Deitel

"Object Oriented Programming with C++" , Robert Lafore

"SFML Game Development By Example", Raimondas Pupius

12.2. Web References:

https://www.sfml-dev.org/

https://devdocs.io/cpp/

https://docs.microsoft.com/en-us/cpp/

SFML Tutorials by Suraj Sharma