11-02-2020,Gliwice

Semester - 3rd

Group - 2nd

Section - 2nd

Computer Programming 3

Author- Tirth Donda

Email- ttirth3@gmail.com

Tutor- dr inż. Michał Staniszewski

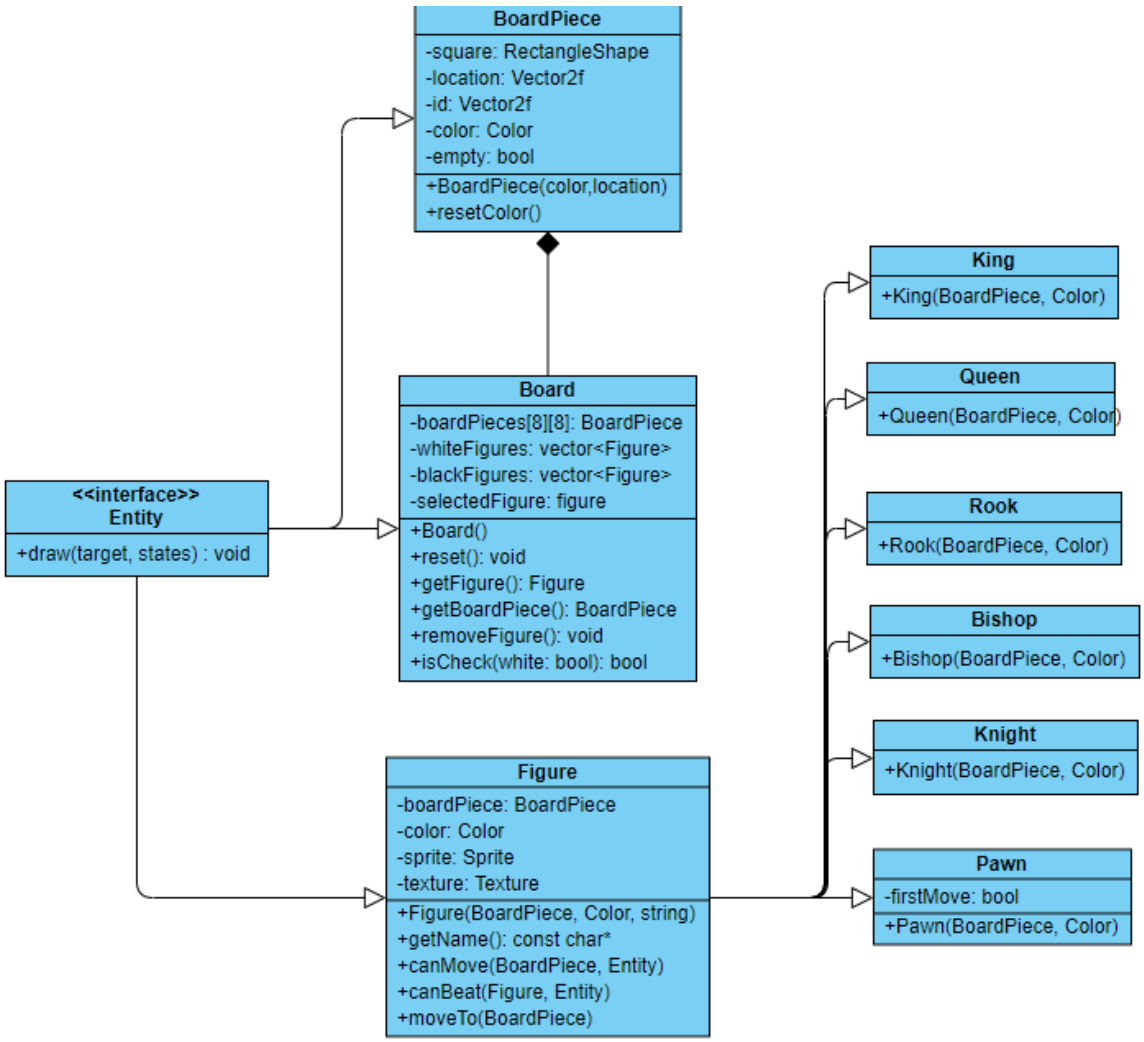
1.Introduction and functionality

Chess is 2-player strategy game played on a chequered board with 64 squared arranged in an 8x8 grid. I wrote this game using C++ language. I did graphical user interface to this game, using SFML. GUI have a bitmap for all the board pieces.

2. Internal specifications

2.1 function: validating a move – after the user has selected a piece and location by mouse, several things must be checked to verify if it is valid move or not. Game checks the types of piece and check the type of move it should be allowed. Likewise, is there another piece on destination or if this move puts the king in check.

2.2 Class diagram: picture below shows the classes I have used.



Inheritance: this allows one class to inherit from another

```
#include "Entity.h"

class BoardPiece : public Entity
{
private:
    sf::Vector2f location;
    sf::Vector2i id;
    sf::RectangleShape square;
    sf::Color color;
    bool empty = true;
public:
    static const int SQUARE_SIZE = 50;
    BoardPiece(sf::Color color, sf::Vector2f loc) : co
        id = (sf::Vector2i)loc;
```

Templates: Template is needed to pass data type as a parameter so that we don't need to write the same code for different data types.

```
    BoardPiece* boardPieces[8][8];
    //template stl class
    std::vector<Figure*> whiteFigures;
    std::vector<Figure*> blackFigures;
    Figure* selectedFig = NULL;
    bool selected = false;
    bool checkBlack = false;
    bool checkWhite = false;
```

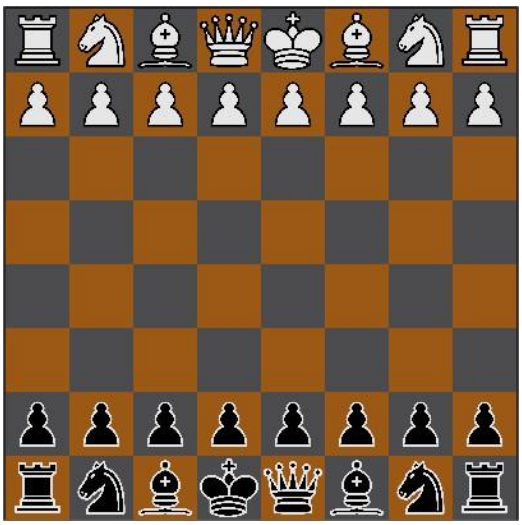
RTTI: the **RTTI** is a mechanism, that exposes information about an object's datatype during runtime.

```
sf::Vector2i destination = bp->getId();
//RTTI
Board *board = dynamic_cast<Board*>(b);
//legal rook move
if (current.x == destination.x ) {
    //next check for obstacles on the way
    int i = current.y > destination.y ? destination.y : current.y;
    int max = current.y > destination.y ? current.y : destination.y;
    i++;
    for (i; i < max; i++) {
        if (!board->getBoardPiece(current.x,i)->isEmpty()) {
            return false;
```

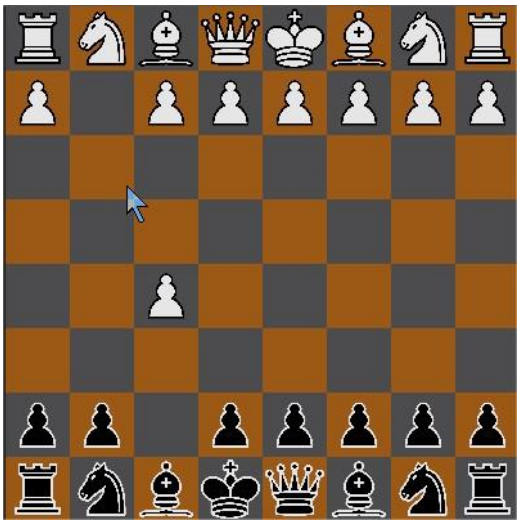
Polymorphism: The word polymorphism means having many forms. Typically, polymorphism occurs when there is a hierarchy of classes and they are related by inheritance. In my game it is used almost all over the places.

3. external specification

Below are some screenshots from my game



below white piece is selected which will move diagonally to beat black piece.



Below black queen checks for white king

4. Testing: the tests were time consuming because I had to play many games. I discovered a problem with black pawn beating same color pieces, then I fixed it.

5. Conclusion: During creating of the project there were some test done to assure that it works as I wanted it to work. It represent the experience which I gained during the course and new topics that I learnt . This project gave me real insight in the world of programming.