CMPE230

SYSTEMS PROGRAMMING

PROJECT 3

PROJECT NAME

CARD MATCHING GAME

PROJECT TYPE

PROGRAMMING PROJECT

STUDENTS

FATİH DEMİR

EYMEN ESAD ÇELİKTÜRK

SUBMISSION DATE

23.05.2023, 23.59

1 Introduction

We implemented a simple memory matching game in Qt using C++. The cards in the game are laid out in a grid, and each card has a color name. The goal is to click on matching card pairs to find them. The number of tries decreases with each attempt, and the score increases when a match is found. When all possible matches are discovered or the permitted number of attempts is used, the game is over. The code employs game logic to look for matching pairs and Qt's signal-slot mechanism to handle button clicks. The development of a memory matching game in Qt using C++ is explained in detail in this documentation.

2 Program Structure

MainWindow Class: The program defines a class called MainWindow, which inherits from QMainWindow. This class represents the main window of the game application. It contains member variables and functions for managing the game logic and user interface.

We implemented several functions whose explanations are given below.

createGameGrid() Function: This function is responsible for creating the game grid consisting of cards with hidden colors. It uses a nested loop to create QPushButton objects representing each card and connects them to a slot for handling card clicks. The cards are added to the grid layout, and their text and properties are set randomly using the rand() function.

resetGame() Function: The resetGame() function resets the game state to its initial values. It is called when the user clicks the "New Game" button. This function resets the score, remaining tries, and updates the respective labels. It also enables all cards and sets their text to "?".

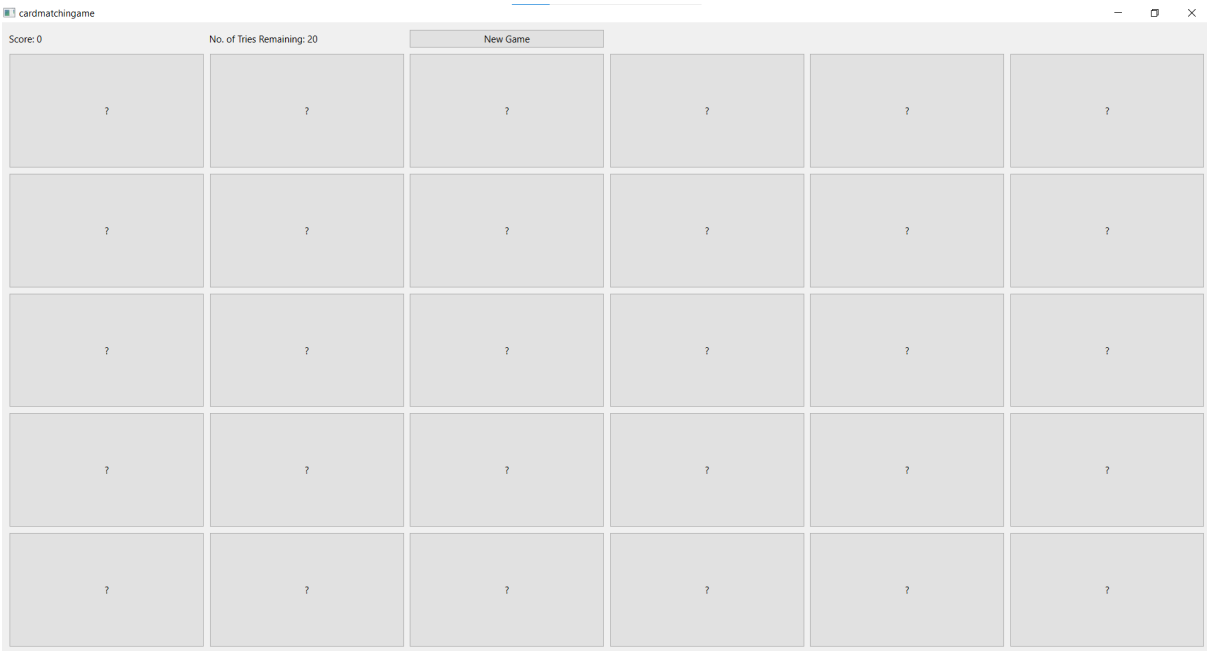
cardClicked() Function: This function is the slot that is called when a card button is clicked. It first checks if the clicked card is enabled. If not, it returns. Otherwise, it disables the clicked card, reveals its color by setting the text, and updates the game state based on whether it's the first or second card being turned. When the second card is turned, checkCards() function is called.

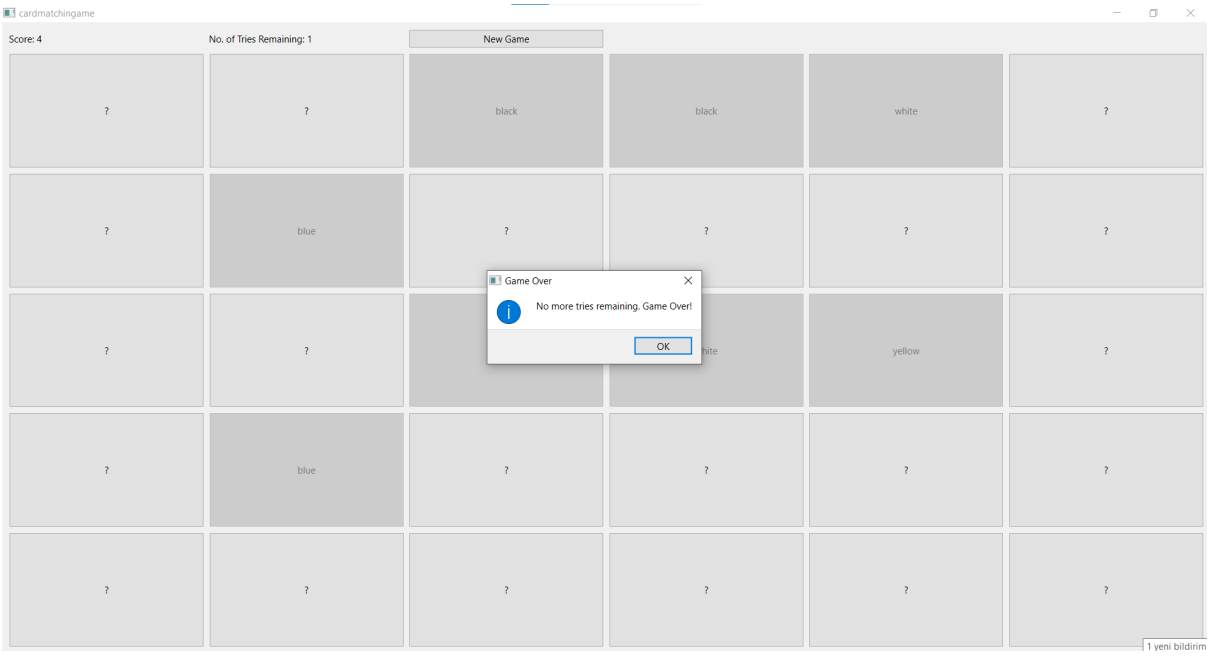
checkCards() Function: The checkCards() function is responsible for comparing the colors of the first and second turned cards. If they match, the score is incremented, and the cards remain disabled. If they don't match, a timeout of 500 milliseconds is set using QTimer::singleShot(). After the timeout, the cards are re-enabled, hidden again, and the first and second card pointers are reset. If the maximum number of tries is reached, the endGame() function is called.

endGame() Function: The endGame() function displays a message box indicating that the game is over because there are no more remaining tries. It also calls the resetGame() function to restart the game.

4 Input-Output and Examples

Examples are provided below.





7 Conclusion

By using Qt's GUI elements, signal-slot mechanism, and game logic implementation, in conclusion, the provided code demonstrates the development of a simple memory matching game using the Qt framework in C++.