CMPE230

Homework 3

Abdullah Yıldız – Yusuf Yüksel

2018400291 - 2014400051

In this project, we implemented a Find the Pairs game!

Game starts with all the cards closed. X represents that card is closed.

Model:

Our game consists of

24 push buttons for cards,

1 push button for resetting the game,

2 QLCD Number for the number of tries and pairs found

Start:

We are using a vector of 24 named *'alphabet'* to store the 12 letter pairs. At the beginning of the game, we shuffle this vector *using std::random_shuffle(alphabet.begin(),alphabet.end());*

To obtain a really random shuffle, we use srand(time(0)); which uses the computer's current time as seed.

We use an array of 24 named Buttons to store the QPushButton's.

At line 37, we are using a for loop to add all the pushbuttons to our array and connect them using connect(…) function.

We are using a queue to store pressed QPushButtons.

Logic:

Timer():

Creates a QTimer and connects closeCard() function and QTimer() together. This enables that after 1.1 second closeCard() function will be called.

closeCard():

- Enables the *popped* and *button* buttons, sets their name to *X.*
- In a for loop, connects all the buttons that it was disabled before the closeCard() function.

slotButtonClicked():

*button= (QPushButton *)sender();* obtains the sender of signal.

Stores the name of the button in a variable *mystring*, takes the substring of it then converts it to first string then integer.

Since the button names are B01,B02,…. , substrings 01,02,03 gives us index for alphabet vector. Then we can give text to buttons using elements of shuffled vector.

Every time a button is pressed, it is disabled and pushed to queue.

Every time a button is pressed, we check for the number of elements in the queue. If 2, then we have to pop them and check for equality.

If-else statement handles this,

- First, we disconnect all the push buttons to prevent user to press 3rd button, it also gives time to user to see the letter.
- *tries* is incremented.
- Queue is cleared, buttons are *button* and *popped*
    - If their texts are the same, then it is a match! pairs is incremented and displayed on screen. Then, we connect every button again and finish the function.
    - If not, then we call *Timer*() function. see "*timer()"*

SlotRestartClicked():

tries and pairs are set to zero and displayed on screen.

Until queue is cleared, we pop elements

All the buttons are enabled and set their text to X.

Mainwindow.h

Header file for Mainwindow.cpp

It has slots :

slotRestartClicked,slotButtonClicked,Timer,closeCard

main.cpp

It creates a Mainwindow object named w, and shows on screen.