1. Introduction

Project Overview

The Minesweeper game is a classic puzzle game where the objective is to clear a rectangular board containing hidden "mines" without detonating any of them. This implementation is built using the Qt framework, providing a graphical user interface (GUI) for a better user experience.

Development Tools

- Framework: Qt
- Language: C++
- IDE: Qt Creator (or any IDE supporting Qt)

2. Project Structure

Classes and Files

- **Minesweeper**: Main class handling the game logic and UI.
  - **minesweeper.h**: Header file declaring the Minesweeper class.
  - **minesweeper.cpp**: Implementation file defining the Minesweeper class methods.
- **main.cpp**: Entry point of the application.
- **minesweeper.ui**: UI file defining the layout.
- **resources.qrc**: Resource file for managing image assets.

3. Classes and Their Functions

Public Methods

- **Minesweeper(QWidget *parent = nullptr):** Constructor to initialize the game.
- **~Minesweeper():** Destructor to clean up resources.

Private Slots

- **void handleCellClick():** Handles left-click events on cells.
- **void handleRightClick():** Handles right-click events on cells.
- **void restartGame():** Restarts the game.

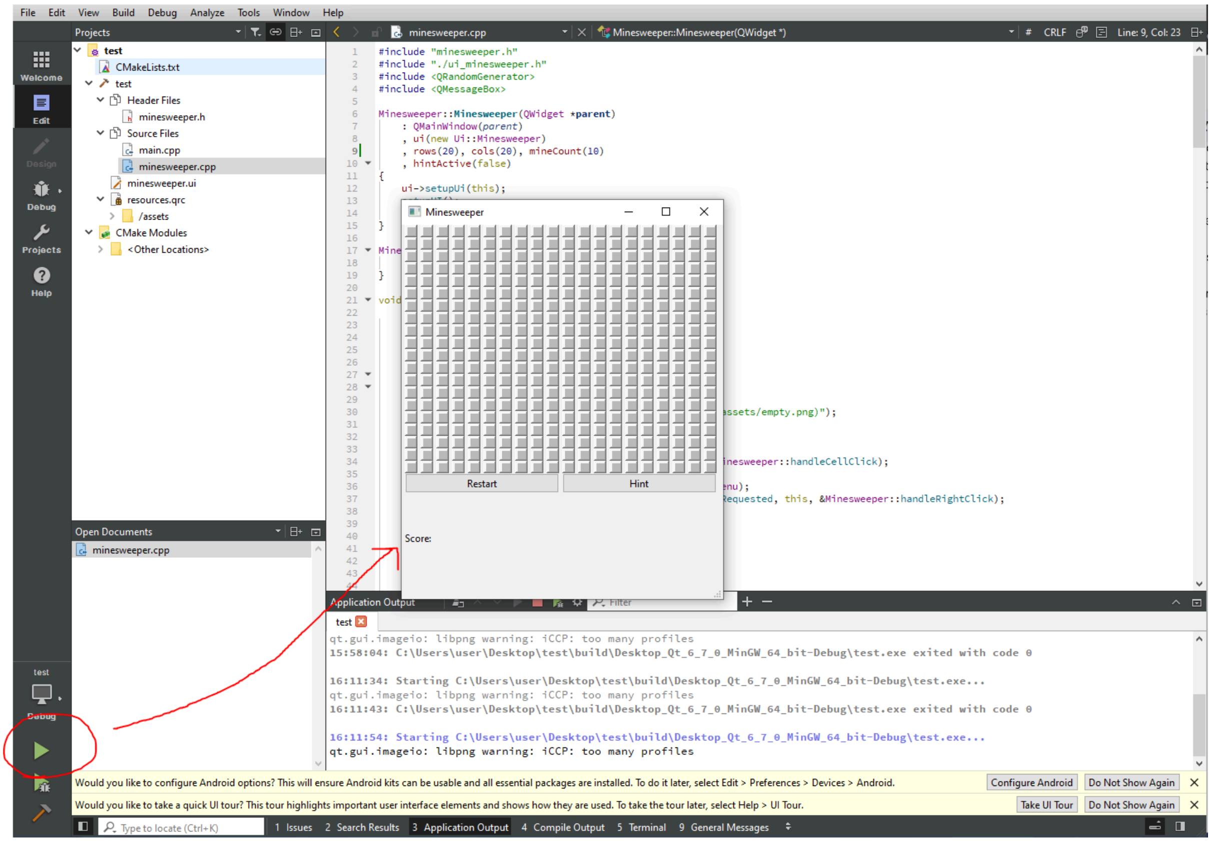
- **void giveHint():** Provides a hint by revealing a safe cell.

Private Methods

- **void setupUI():** Sets up the user interface components.
- **void initializeGame():** Initializes the game board and places mines.
- **void revealCell(int row, int col):** Reveals the contents of a cell.
- **void revealAdjacentCells(int row, int col):** Reveals all adjacent cells recursively.
- **int countAdjacentMines(int row, int col):** Counts the number of mines adjacent to a cell.
- **bool isMine(int row, int col):** Checks if a cell contains a mine.
- **void disableHint():** Disables the hint functionality.
- **void revealAllMines():** Reveals all mines on the board.
- **void checkWinCondition():** Checks if the player has won the game.

4. Gameplay and Functionality

Game is run when we run our project in QT Creator:



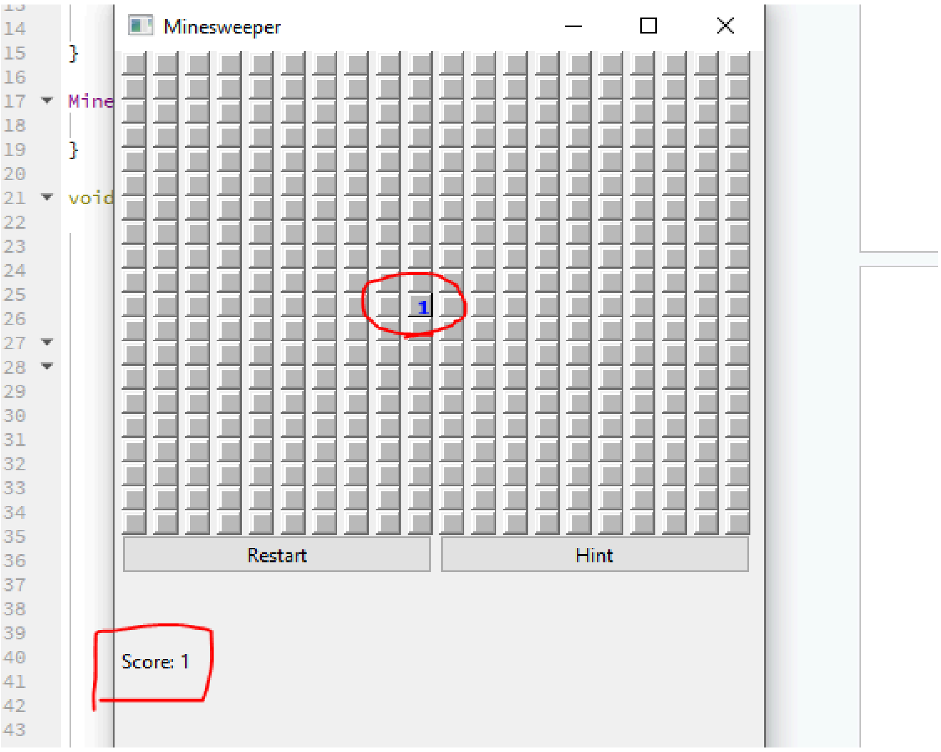
```
#include "minesweeper.h"
#include "./ui_minesweeper.h"
#include <QRandomGenerator>
#include <QMessageBox>

Minesweeper::Minesweeper(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::Minesweeper)
    , rows(20), cols(20), mineCount(10)
    , hintActive(false)
{
    ui->setupUi(this);

}

Mine

}

void

assets/empty.png)");

inesweeper::handleCellClick);

enu);
Requested, this, &Minesweeper::handleRightClick);
```

```
qt.gui.imageio: libpng warning: iCCP: too many profiles
15:58:04: C:\Users\user\Desktop\test\build\Desktop_Qt_6_7_0_MinGW_64_bit-Debug\test.exe exited with code 0

16:11:34: Starting C:\Users\user\Desktop\test\build\Desktop_Qt_6_7_0_MinGW_64_bit-Debug\test.exe...
qt.gui.imageio: libpng warning: iCCP: too many profiles
16:11:43: C:\Users\user\Desktop\test\build\Desktop_Qt_6_7_0_MinGW_64_bit-Debug\test.exe exited with code 0

16:11:54: Starting C:\Users\user\Desktop\test\build\Desktop_Qt_6_7_0_MinGW_64_bit-Debug\test.exe...
qt.gui.imageio: libpng warning: iCCP: too many profiles
```

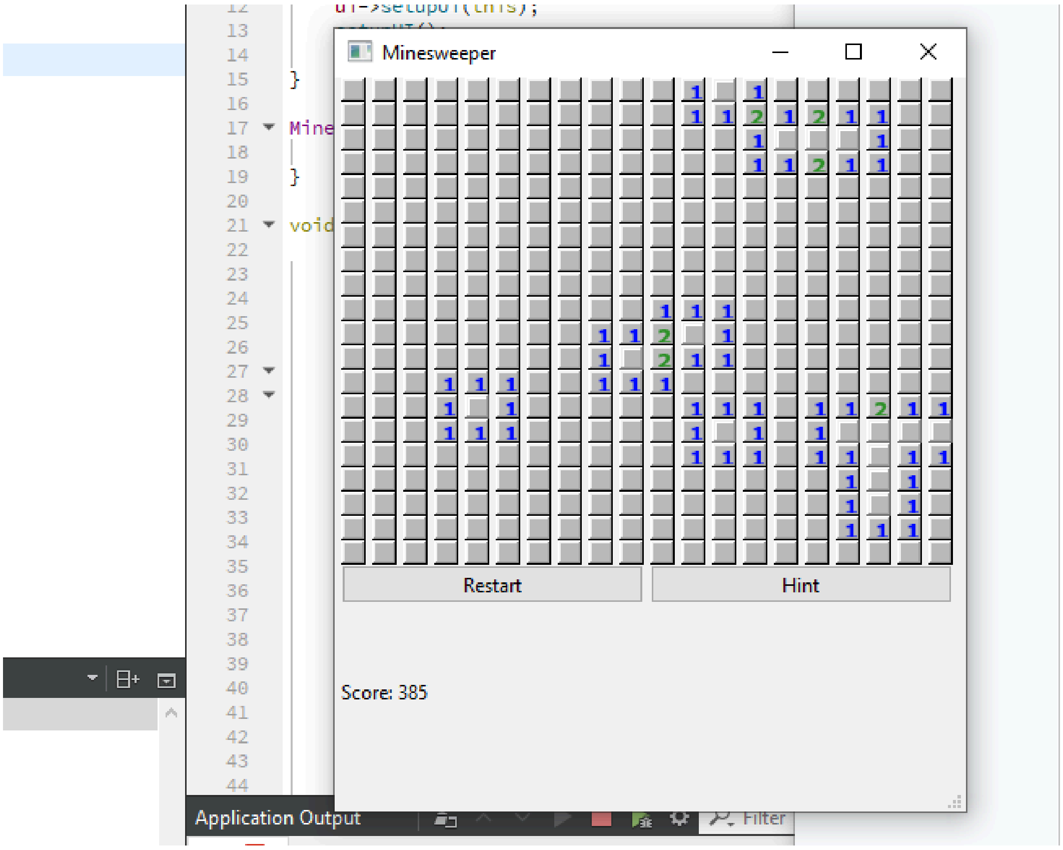
Would you like to configure Android options? This will ensure Android kits can be usable and all essential packages are installed. To do it later, select Edit > Preferences > Devices > Android.

Would you like to take a quick UI tour? This tour highlights important user interface elements and shows how they are used. To take the tour later, select Help > UI Tour.

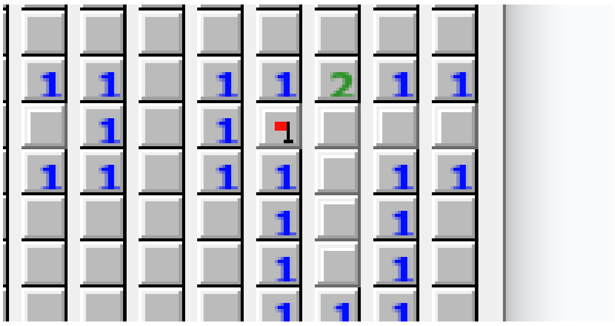
Left click reveals a tile and also increments the score:



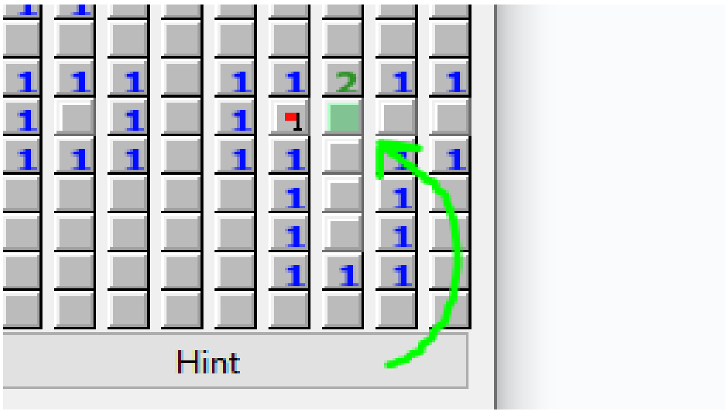
When we click a safe-around tile, **adjacent tiles are revealed recursively**:



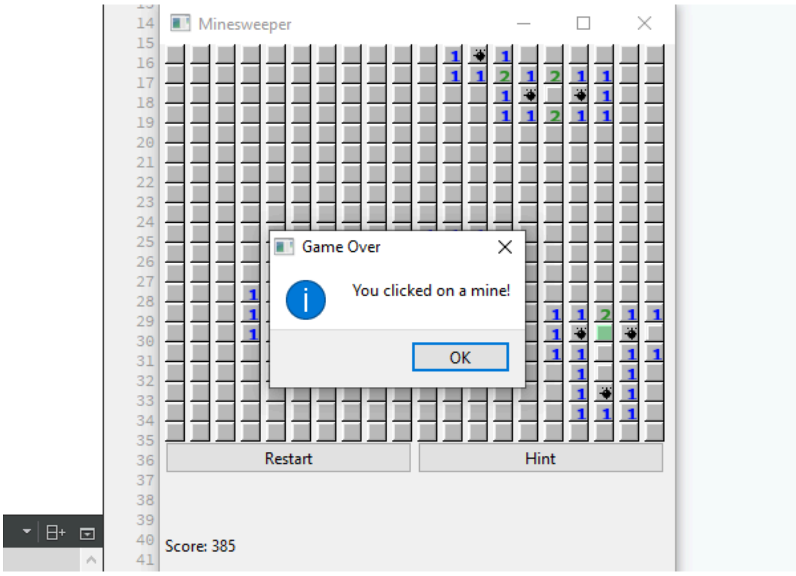
If the player suspects that a specific tile contains a mine,
they can **flag the tile by right clicking** it:



If the player needs some help, they can click the **Hint** button to get a **guaranteed click:**



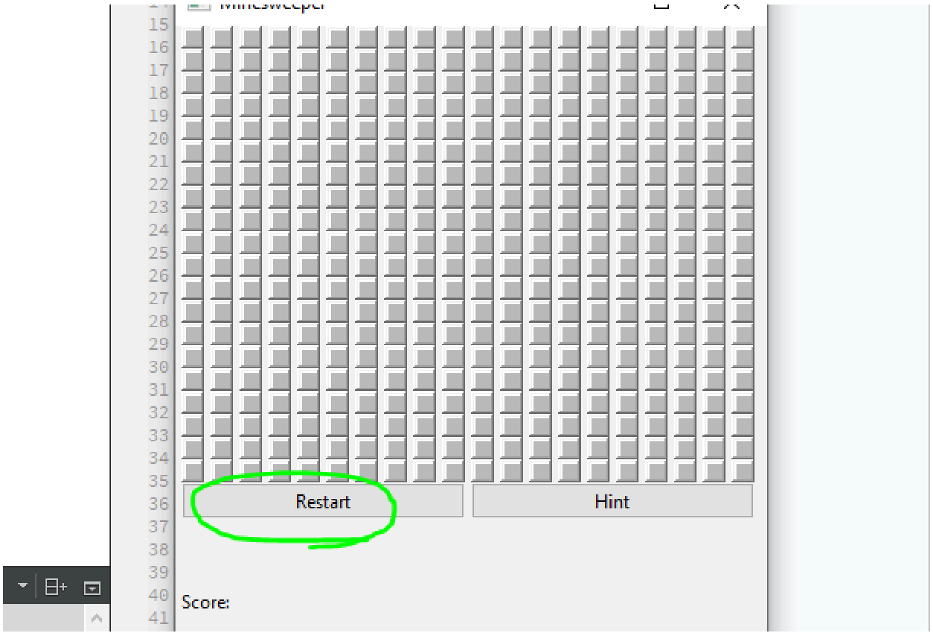
After clicking on a mine, game is lost and a pop-up screen shows. All mines' places are revealed.



You clicked on a mine!

Score: 385

Clicking the Restart button will restart the game with a new mine field



5. Some Challenges

Spamming the Hint button shouldn't give new hints:

-> Solved by using a hintActive flag.

Buttons should be unclickable after losing but before restart:

-> Setting the grid's setEnabled value to false (or simply disable the whole grid) after revealing all mines (game ended).

Hint shouldn't be prophetic but should be near to our knowledge:

-> Solved by using a nested loop inside the giveHint() method to look at revealed tiles' adjacent tiles for a safe tile and then give a hint.