Game Programming: Exercise 2: C++ Foundation

<table>
<tr><td>Learning objectives</td><td>

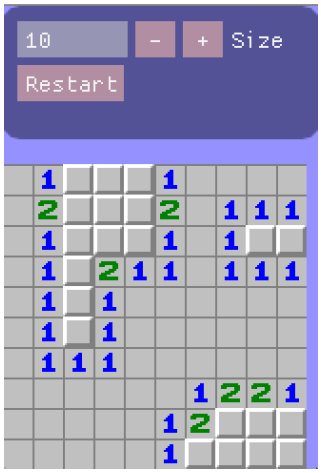
Use basic C++ language features (functions, classes, loops, conditionals, variables) to solve simple programming challenges
Use classes of the STL library to store data
Use C++ string objects
Handle input and output of command line programs
Use preprocessor macros

</td></tr>
<tr><td>Exercise 2.1</td><td>Write a program that prints the date and time of which the program was compiled and the current time and date. Hint: Use C++ preprocessor macros and <ctime> header.</td></tr>
<tr><td>Exercise 2.2</td><td>One classic method for composing secret messages is called a square code. The spaces are removed from the English text and the characters are written into a square. For example, the sentence "If man was meant to stay on the ground god would have given us roots" is 54 characters long, so it is written into a rectangle with 8 rows (the last row is empty) and 8 columns (in total 64 fields). The size of the square is the minimum that can contain the sentence.
<table>
<tr><td>i</td><td>f</td><td>m</td><td>a</td><td>n</td><td>w</td><td>a</td><td>s</td></tr>
<tr><td>m</td><td>e</td><td>a</td><td>n</td><td>t</td><td>t</td><td>o</td><td>s</td></tr>
<tr><td>t</td><td>a</td><td>y</td><td>o</td><td>n</td><td>t</td><td>h</td><td>e</td></tr>
<tr><td>g</td><td>r</td><td>o</td><td>u</td><td>n</td><td>d</td><td>g</td><td>o</td></tr>
<tr><td>d</td><td>w</td><td>o</td><td>u</td><td>l</td><td>d</td><td>h</td><td>a</td></tr>
<tr><td>v</td><td>e</td><td>g</td><td>i</td><td>v</td><td>e</td><td>n</td><td>u</td></tr>
<tr><td>s</td><td>r</td><td>o</td><td>o</td><td>t</td><td>s</td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>
The coded message is obtained by reading down the columns going left to right and skipping empty fields. For example, the message above is coded as:

<code>imtgdvs fearwer mayoogo anouuio ntnnlvt wttddes aohghn sseoau</code>

In your program, have the user enter a message in English with no spaces between the words. Display the encoded message. Find more examples words and their encryption in the source code.</td></tr>
</table>

(Continues on next page)

Exercise 2.3 3 1 X X 2 1 X X 1 1 1 1 0 0 1 2 3 Flipped 0 0 Enter row: *3* Enter column: *3* **Console version**  **GUI version**	Extend the code in **exercise2-3-MineSweeper/** to contain a full minesweeper game. The game must include the following features: • Custom game size (up to 10x10) • Choose an appropriate number of randomly positioned bombs. • If the selected field is empty when surrounding empty fields should be uncovered • If the user selects a bomb, then the game ends and user loose. • If remaining hidden fields all contain bombs, then the user wins. Hint: You can use nested vectors to store a dynamic two-dimensional array. You only need to modify **MineSweeper.cpp** and MineSweeper.hpp. Existing member functions in the header must not be changed, but you should add additional member functions and fields. See how the **CMakeLists.txt** script creates two different configurations of the game: • **Console version** (uses the **main.cpp** and **MineSweeper.cpp**) • **GUI version** (uses **main_gui.cpp** and **MineSweeper.cpp**). This version links to dependent libraries in SimpleRenderEngineProject and copy the necessary assets (and dlls on windows) to the project build path (see the **file(COPY …)** commands) The result should be similar to this: https://www.itu.dk/~mnob/minesweeper/MineSweeperGUI.html
Exercise 2.4 (Optional)	Extend the MineSweeper gui game in the following ways: • When the user hit a mine, all mine should be displayed • Right-click flags elements as potential mines. This should prevent the user from selecting the elements until the flag is removed by another right-click. This requires modifications to main_gui.cpp