Instruction for the project SOKOBAN

Basics of Programming 2022, Data Engineering, 3rd term.

Project Sokoban

1. The aim

Implement a board game Sokoban. The board consists of square fields. Field can be:

- empty,
- occupied by the player,
- occupied by the chest (crate),
- set as the chests' destination,
- wall.

The board is always surrounded by walls. A player (human) has the ability to move from one field to another, only if it is free. In addition, the player can push the chests (only one at a time), provided the field behind the crate is free. The player can not pull the crates and move them sideway. The game is to move the crates in the warehouse in such a way that all the crates are set on the destination. A more detailed description - https://en.wikipedia.org/wiki/Sokoban.

2. Programming environment

To the instruction there is attached a basic program which includes:

- Calculation of time increases, which allows to measure the time spent
- Drawing bmp graphic files
- Drawing a pixel, a line, and a rectangle
- Displaying a text

The program uses SDL2 (2.0.3) – http://www.libsdl.org/ library. It is included in the basic project – there is no need to download the sources.
The compilation under a 32-bit Linux system can be done be the following command:

```
g++ -O2 -I./sdl/include -L. -o main main.cpp -lm -lSDL2 -lpthread -ldl -lrt
```

and under a 64-bit system with the following command:

```
g++ -O2 -I./sdl/include -L. -o main main.cpp -lm -lSDL2-64 -lpthread -ldl -lrt
```

To compile the project, the directory containing main.cpp should contain:

- Bitmaps with the needed pictures (cs8x8.bmp, eti.bmp), be sure to preserve proper capitalization of letters.
- The libSDL2.a file (libSDL2-64.a when compiling a 64-bit version)
- The sdl directory attached to the project.

To the project, there are attached scripts that can be used for a compilation (comp for a 32-bit version environment and comp64 for a 64-bit version environment).

The presentation (for the grading) of the project will be in the operating system chosen by a student. The following choices are available:

Linux system: The student is required to check if the project can be properly compiled and runs correctly under the laboratory distribution.
Windows System and compiled using MS Visual C++ studio with the version available at the laboratory.

The successful execution of the program during the presentation is required to obtain points from Project 2.

The program shall not use the C++ stl library.

3. Specification

Basic functionality for 5 points. (all have to be implemented to get any points):

1. Display the board in aesthetic and ergonomic way.
2. Displaying elapsed time during gameplay. Time is reset when a new board is loaded.
3. Move the player character with the arrow keys in the free fields without breaking the walls, the movement is immediate - responding to the events. The character can not enter the walls and crates.
4. Support Esc and n. Where Esc closes the program and n reads the current board again.

Advanced functionality for 10 points:

1. Moving the crates by pushing them. Moving the crate is possible if the field behind it is free. Player can not move two crates at a time (2 points).
2. Animated transition between fields (2 points).
3. Load the board from the file (1 point).
4. Display and handle menu (2 points):
    - load ready-made boards on different levels. The boards' list should be visible in the menu. The boards' filenames cannot be directly embedded in the program code. For example, you can create additional configuration file (txt) that lists available boards.
    - add user boards. User gives the file name for its own board.
    - playing multiple games (one after the other) during one launch of the program.
5. Detect the (successful) completion of current level and display the message (1 point).
6. Store and display (from menu) the best results' list for each level (board). Single result consist of a number of moves and time to complete. The player has the ability to append to the list after solving the board. The list is limited only by memory size (dynamic memory allocation). Best results' list should be persistent (i.e., stored in a file and available between program launches). The menu allows you to display the best results' list for each level in two ways: sorted by time and sorted by the number of moves (2 points).

4. Final remarks

1. The use of the STL library is prohibited.
2. Compiling and launching a program is a prerequisite for awarding any points. The student must check whether the laboratory in which he will present is equipped with the appropriate software.
3. The project (source code) must be realized by the student himself. The student must be able to explain each element of the code and apply minor changes (the teacher may ask).
4. The game should be playable and convenient to use.