COURSE PROJECT DOCUMENTATION

CS101 Projects 2015

Minesweeper

GROUP CUSE

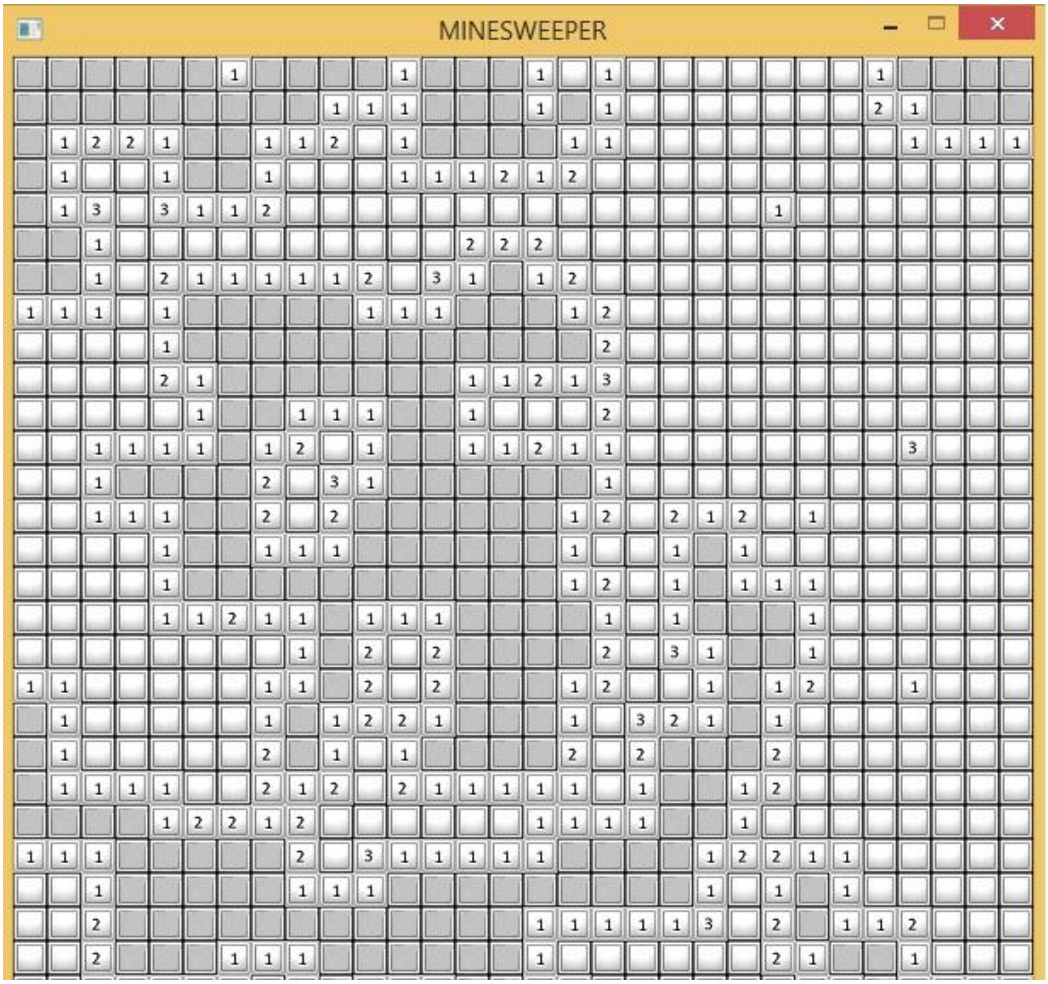
Siddhartha Shering -140110002
Pankil Nimje -140110092
Aditi Sharma -140110012
Shardul Kanvinde -140110009

Table of Contents

1. Introduction

Minesweeper is a single player puzzle game that involves logical thinking.The game was created by Microsoft company to develop analytical skills of people. The game consists of a rectangular grid consisting of multiple tiles some containing mines and rest without mines. The player has to clear the grid by uncovering the tiles but without uncovering the ones containing the mines.

This is how the game window looks like when we have made a few successful moves.



2. Problem Statement

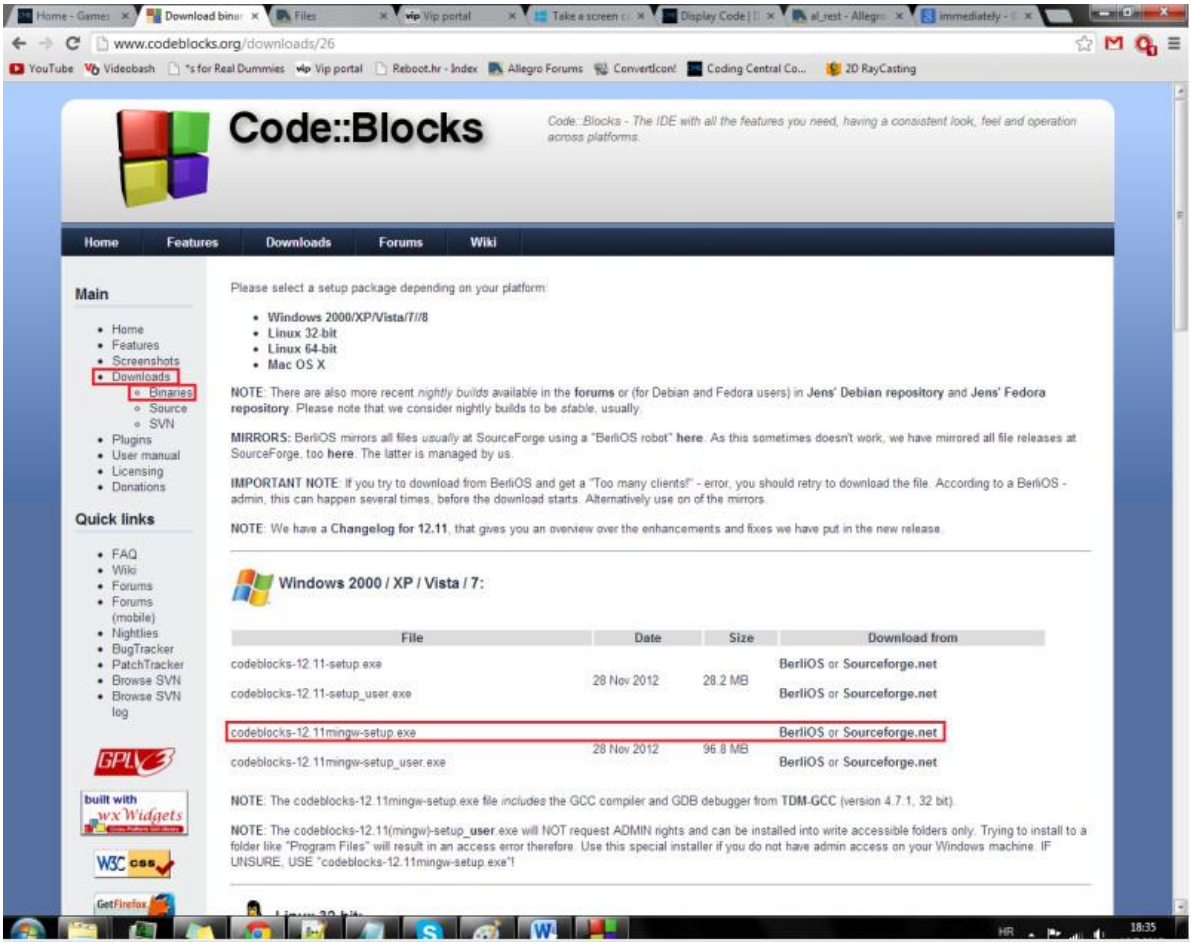
- Minesweeper consists of a rectangular grid having multiple tiles. Some of the tiles contain mines and the rest don't have mines.
- To win, the player has to uncover all the tiles that do not have mines without opening the tiles that have mines.
- Our goal is to create minesweeper game using C++.
- We aim to make it via allegro for graphics.

3.1 Requirements

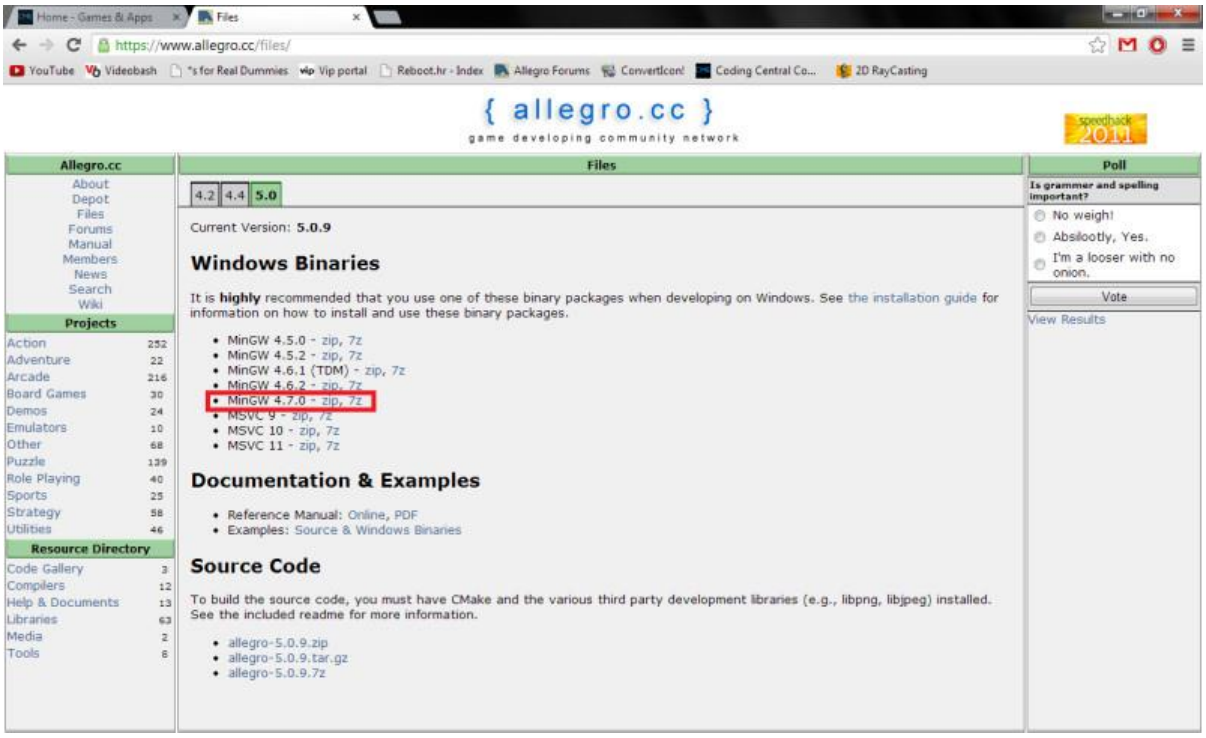
1) A computer with allegro installed in it. We have integrated allegro and code blocks IDE.
2) The user should have windows or linux OS system installed.

3.2 Installing Allegro5 with Code Blocks

1) Download and install the latest Code::Blocks binary release with GCC compiler (if you don't prefer some other compiler) from http://www.codeblocks.org

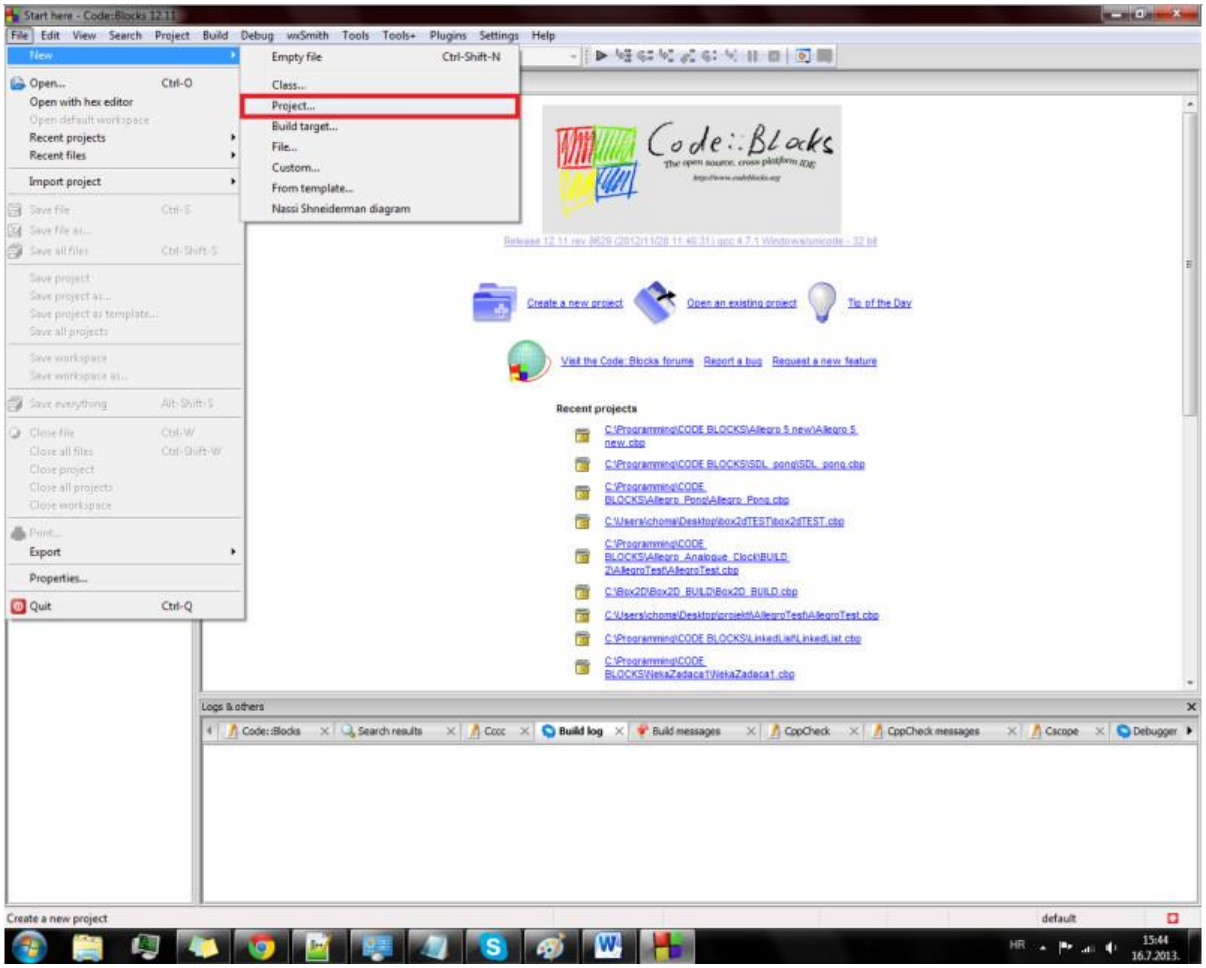

www.codeblocks.org/downloads/26
Code::Blocks
Code::Blocks - The IDE with all the features you need, having a consistent look, feel and operation across platforms.
Home
Features
Downloads
Forums
Wiki
Main
Home
Features
Screenshots
Downloads
Binaries
Source
SVN
Plugins
User manual
Licensing
Donations
Quick links
FAQ
Wiki
Forums
Forums (mobile)
Nightlies
BugTracker
PatchTracker
Browse SVN
Browse SVN log
Please select a setup package depending on your platform:
Windows 2000/XP/Vista/7//8
Linux 32-bit
Linux 64-bit
Mac OS X
NOTE: There are also more recent nightly builds available in the forums or (for Debian and Fedora users) in Jens' Debian repository and Jens' Fedora repository. Please note that we consider nightly builds to be stable, usually.
MIRRORS: BerliOS mirrors all files usually at SourceForge using a "BerliOS robot" here. As this sometimes doesn't work, we have mirrored all file releases at SourceForge, too here. The latter is managed by us.
IMPORTANT NOTE: If you try to download from BerliOS and get a "Too many clients!" - error, you should retry to download the file. According to a BerliOS - admin, this can happen several times, before the download starts. Alternatively use on of the mirrors.
NOTE: We have a Changelog for 12.11, that gives you an overview over the enhancements and fixes we have put in the new release.
Windows 2000 / XP / Vista / 7:
File
Date
Size
Download from
codeblocks-12.11-setup.exe
28 Nov 2012
28.2 MB
BerliOS or Sourceforge.net
codeblocks-12.11-setup_user.exe
BerliOS or Sourceforge.net
codeblocks-12.11mingw-setup.exe
28 Nov 2012
96.8 MB
BerliOS or Sourceforge.net
codeblocks-12.11mingw-setup_user.exe
BerliOS or Sourceforge.net
NOTE: The codeblocks-12.11mingw-setup.exe file includes the GCC compiler and GDB debugger from TDM-GCC (version 4.7.1, 32 bit).
NOTE: The codeblocks-12.11(mingw)-setup_user.exe will NOT request ADMIN rights and can be installed into write accessible folders only. Trying to install to a folder like "Program Files" will result in an access error therefore. Use this special installer if you do not have admin access on your Windows machine. IF UNSURE, USE "codeblocks-12.11mingw-setup.exe"!

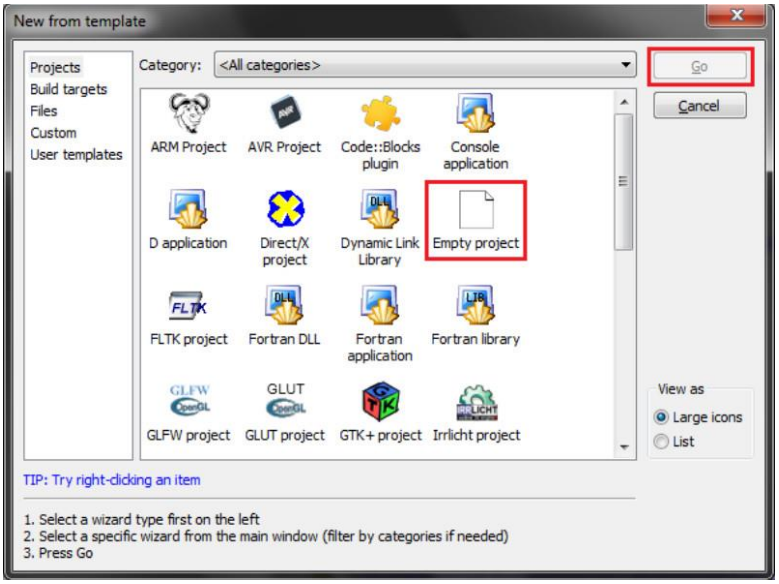
2) Download the latest Allegro 5 binaries from http://www.allegro.cc/files

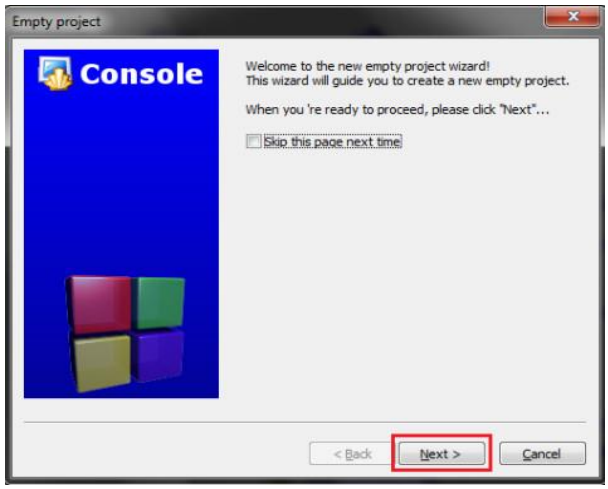


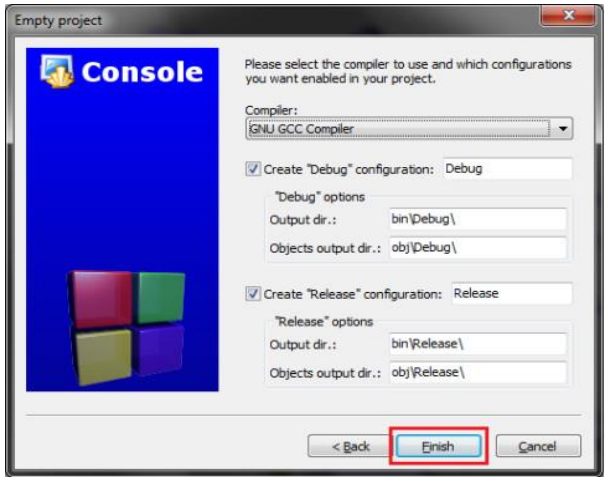
3) Extract the content of allegro-5.x.x-mingw-x.x.x.zip/7z you just downloaded to a location where you won't have to move things a lot, or even forget where they are, like C:\ALLEGRO.

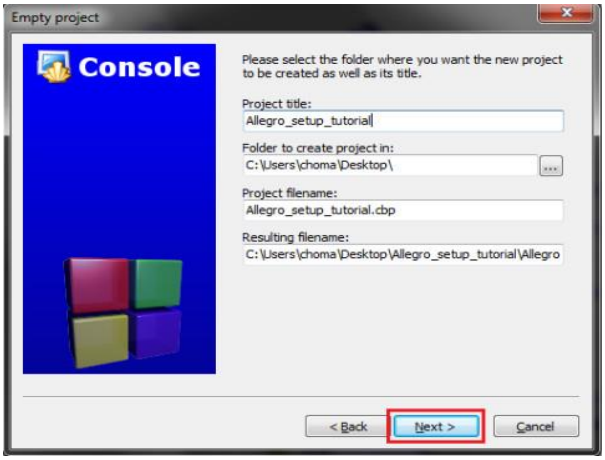
4) Open Code::Blocks and start a new empty project. Complete the wizard. File->New->Project->Empty project.


File Edit View Search Project Build Debug wxSmith Tools Tools+ Plugins Settings Help
New
Open... Ctrl-O
Open with hex editor
Recent projects
Recent files
Import project
Save file Ctrl-S
Save file as...
Save all files Ctrl-Shift-S
Save project
Save project as...
Save project as template...
Save all projects
Save workspace
Save workspace as...
Save everything Alt-Shift-S
Close file Ctrl-W
Close all files Ctrl-Shift-W
Close project
Close all projects
Close workspace
Print...
Export
Properties...
Quit Ctrl-Q
Empty file Ctrl-Shift-N
Class...
Project...
Build target...
File...
Custom...
From template...
Nassi Shneiderman diagram
Code::Blocks
Create a new project
Open an existing project
Tip of the Day
Visit the Code::Blocks forums
Report a bug
Request a new feature
Recent projects
Logs & others
Code::Blocks
Search results
Cccc
Build log
Build messages
CppCheck
CppCheck messages
Cscope
Debugger
Create a new project
default
HR
15:44
16.7.2013.

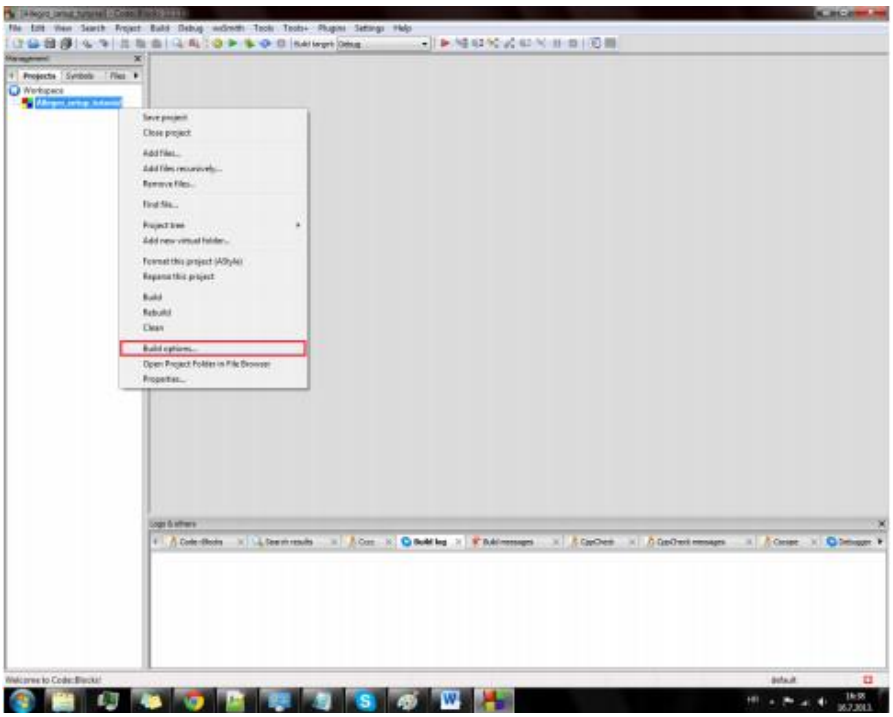

New from template
Projects
Build targets
Files
Custom
User templates
Category:
<All categories>
Go
Cancel
ARM Project
AVR Project
Code::Blocks plugin
Console application
D application
Direct/X project
Dynamic Link Library
Empty project
FLTK project
Fortran DLL
Fortran application
Fortran library
GLFW project
GLUT project
GTK+ project
Irrlicht project
View as
Large icons
List
TIP: Try right-clicking an item
1. Select a wizard type first on the left
2. Select a specific wizard from the main window (filter by categories if needed)
3. Press Go


Empty project
Console
Welcome to the new empty project wizard!
This wizard will guide you to create a new empty project.
When you 're ready to proceed, please click "Next"...
Skip this page next time
< Back
Next >
Cancel

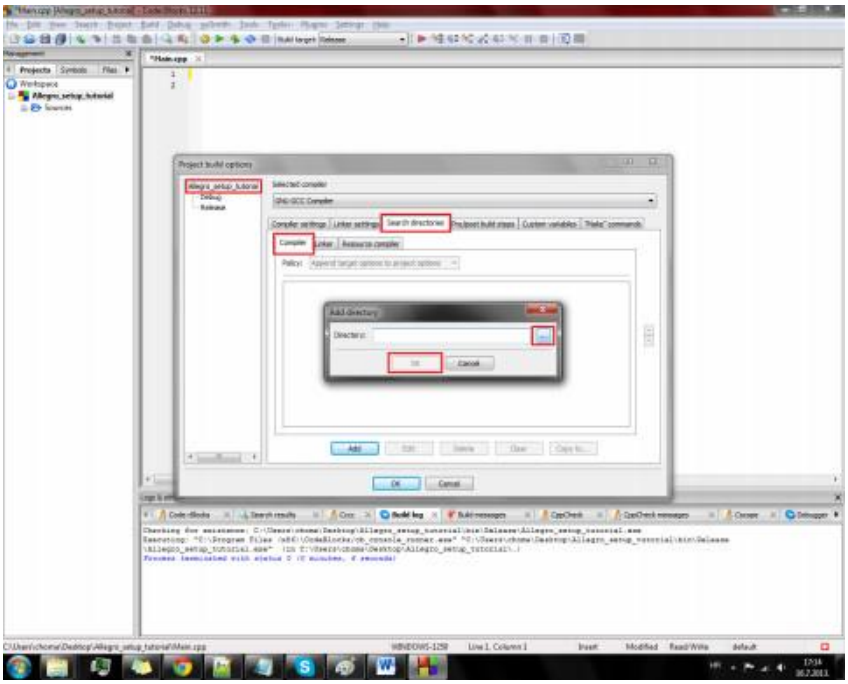

Empty project
Console
Please select the compiler to use and which configurations you want enabled in your project.
Compiler:
GNU GCC Compiler
Create "Debug" configuration:
Debug
"Debug" options
Output dir.:
bin\Debug\
Objects output dir.:
obj\Debug\
Create "Release" configuration:
Release
"Release" options
Output dir.:
bin\Release\
Objects output dir.:
obj\Release\
< Back
Finish
Cancel



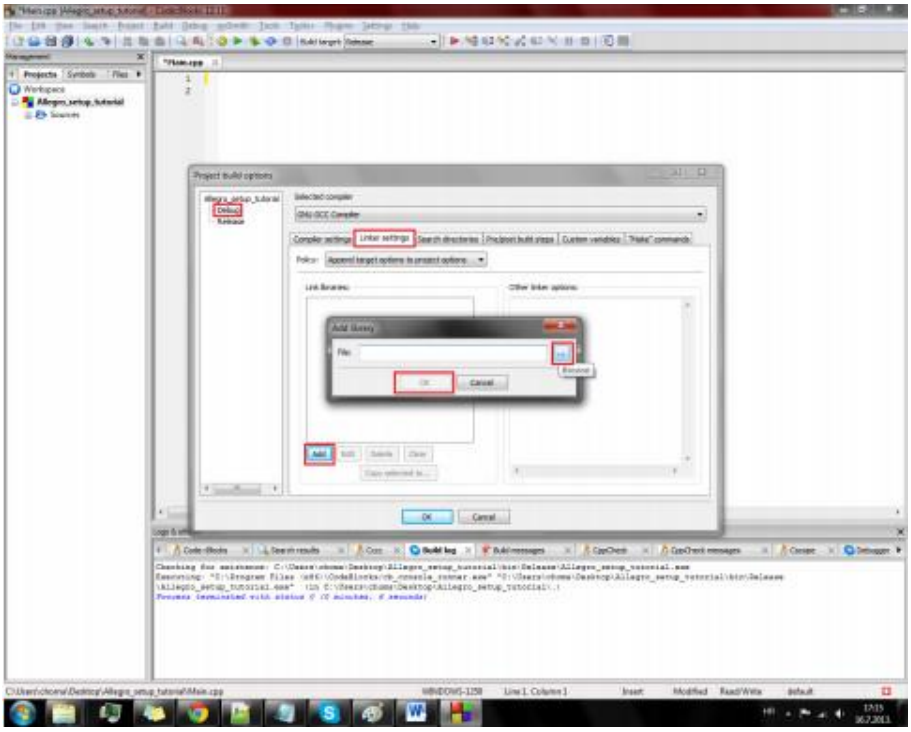
5) Right-click your project file in the workspace on the left side of the screen, and go to Build Options.



6) Click your project name on the left side of the window, go to Search Directories->Compiler->Add, find the Allegro include folder (in my case C:\ALLEGRO\include), click Yes, than OK



7) Now select Debug, and go to Linker Settings->Add. Find the Allegro lib folder (in my case C:\ALLEGRO\lib), and inside it select liballegro-5.x.x-monolith-mt-debug.a (I have the 5.0.9 version, yours may be different).Click Yes, than OK.



8) Now select Release, and repeat step 7, but find the file named liballegro-5.x.x-monolith-mt.a inside the lib folder, instead of the liballegro-5.x.x-monolith-mt-debug.a in step 7.

9) That's all when it comes to setting up Code::Blocks. Now create Main.cpp and paste this code inside:

```
#include <allegro5/allegro.h>     //Main Allegro header
#include <allegro5/allegro_native_dialog.h>     //Header for dialogue boxes
        int main()
{
                        //This is the display on which you draw everything
    ALLEGRO_DISPLAY *display;
                        //If Allegro fails to initialize show a message box
```

```
        if(!al_init())
al_show_native_message_box(NULL, NULL, NULL, "Could not initialize Allegro 5",
NULL, NULL);
                                //Define the display
    display = al_create_display(800, 600);
                                 //If Allegro fails to create a display show a message box
        if(!display)
 al_show_native_message_box(NULL, NULL, NULL, "Could not create Allegro 5
display", NULL, NULL);
                                //Show the application window for 5 seconds
                                        //(if you don't, the application will shut down  ,
                                                immediately after you launch it)
al_rest(5.0f);
                                    //Deallocate the memory used for the display creation
al_destroy_display(display);
return 0;
```

10) Press Ctrl+F9 to build the project, but don't run it yet, it won't work until you copy some .dll files to your project folder. To do that go to your Allegro folder, and from the bin directory (in my case C:\ALLEGRO\bin) copy allegro-5.x.x-monolith-mt-debug.dll and allegro-5.x.x-monolith-mt.dll.

Paste allegro-5.x.x-monolith-mt-debug.dll to your project's Debug folder.

Paste allegro-5.x.x-monolith-mt.dll to your project's Release folder.

If everything went OK, you should be able to run your application without any errors (Press F9 in C::B).You should see a 800x600 white window for 5 seconds.

4. Implementation

Functionality

A. Update function

Function which updates display of blocks. We have defined two arrays storagemap and dispaymap. The mines are allocated to the storagemap and we update the displaymap accordingly since finally displaymap array is displayed.

B. Allocating Mines

We check whether a mine exists at the random location suggested by the rand function. If there is no mine present previously we allocate that position to a mine.

C. Loading images

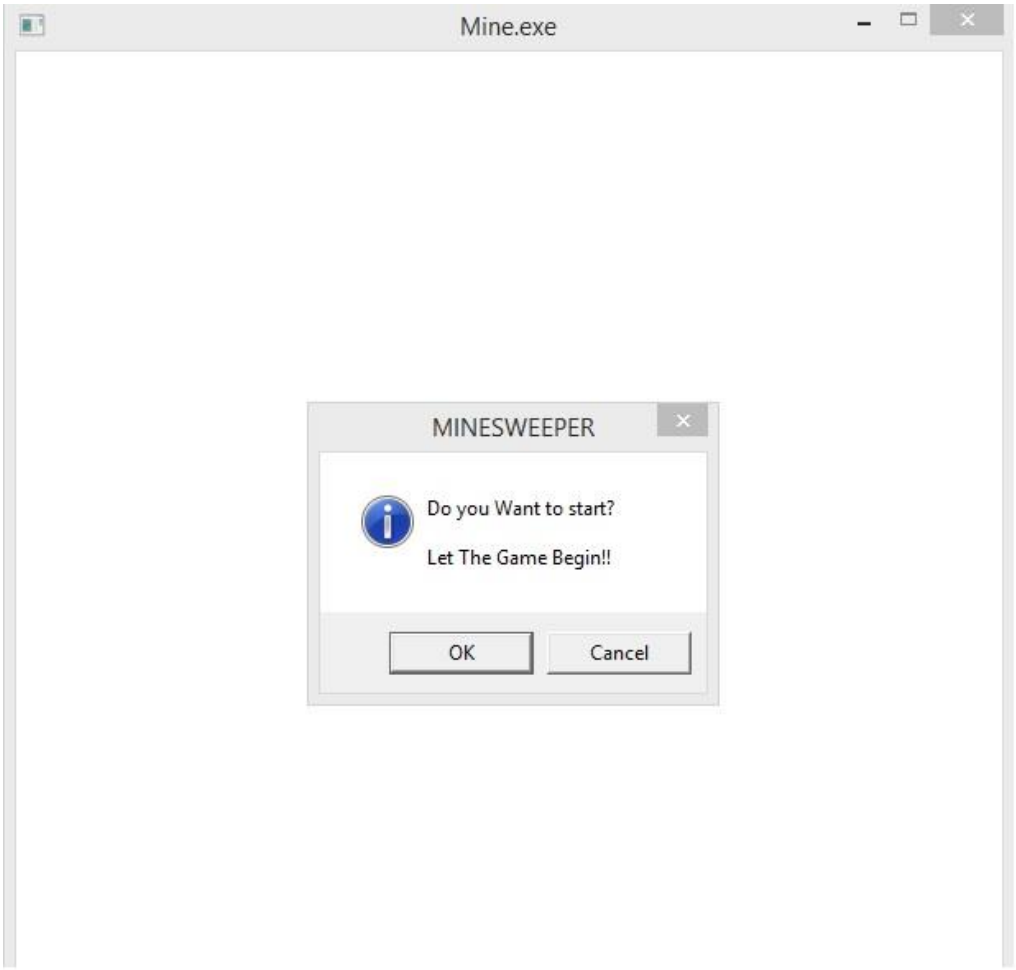
This part has been written as per allegro's syntax. Here we give the locations of each image being used by allegro on the hard drive so that the IDE knows where to get the image from.

D. Opening all the tiles with no mines in contact with the opened tile The user drags the slider on the progress bar.

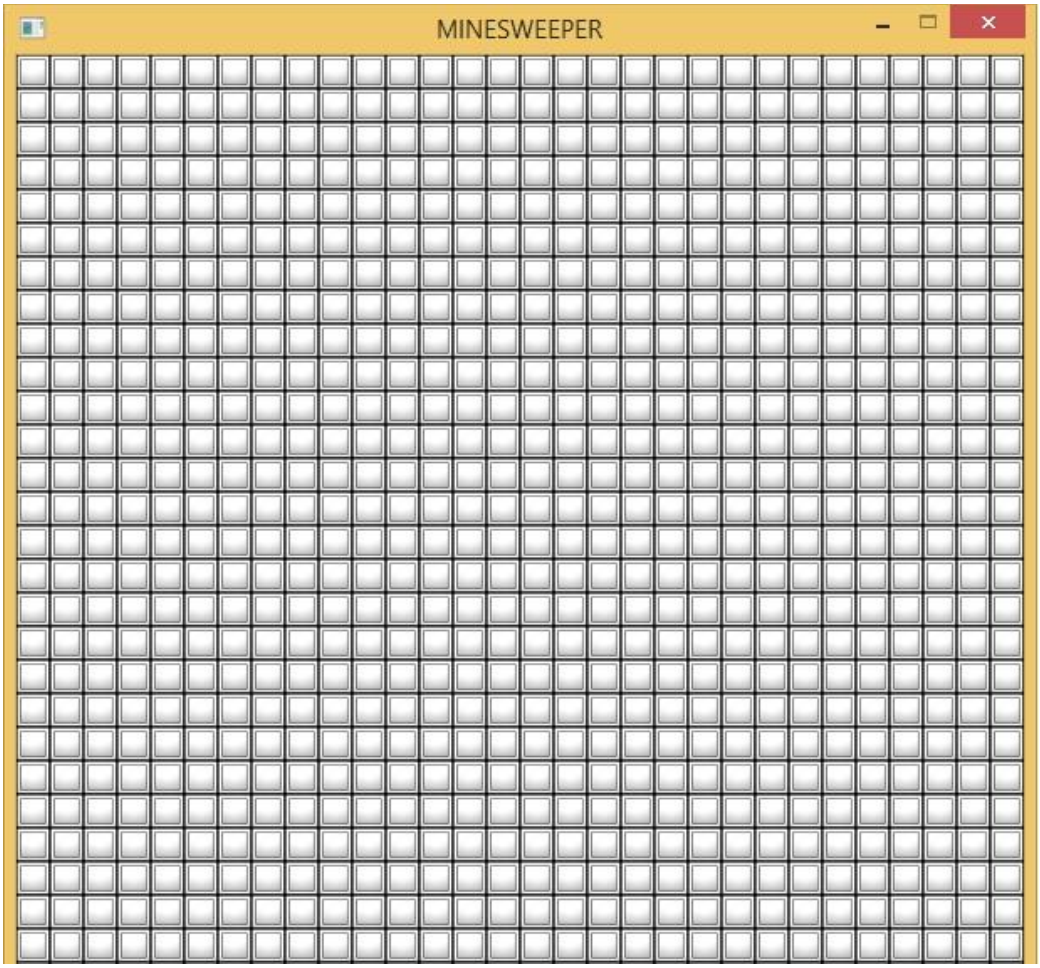
E. Displaying all the mines on the screen once a mine is clicked upon.

5. Testing Strategies and Data

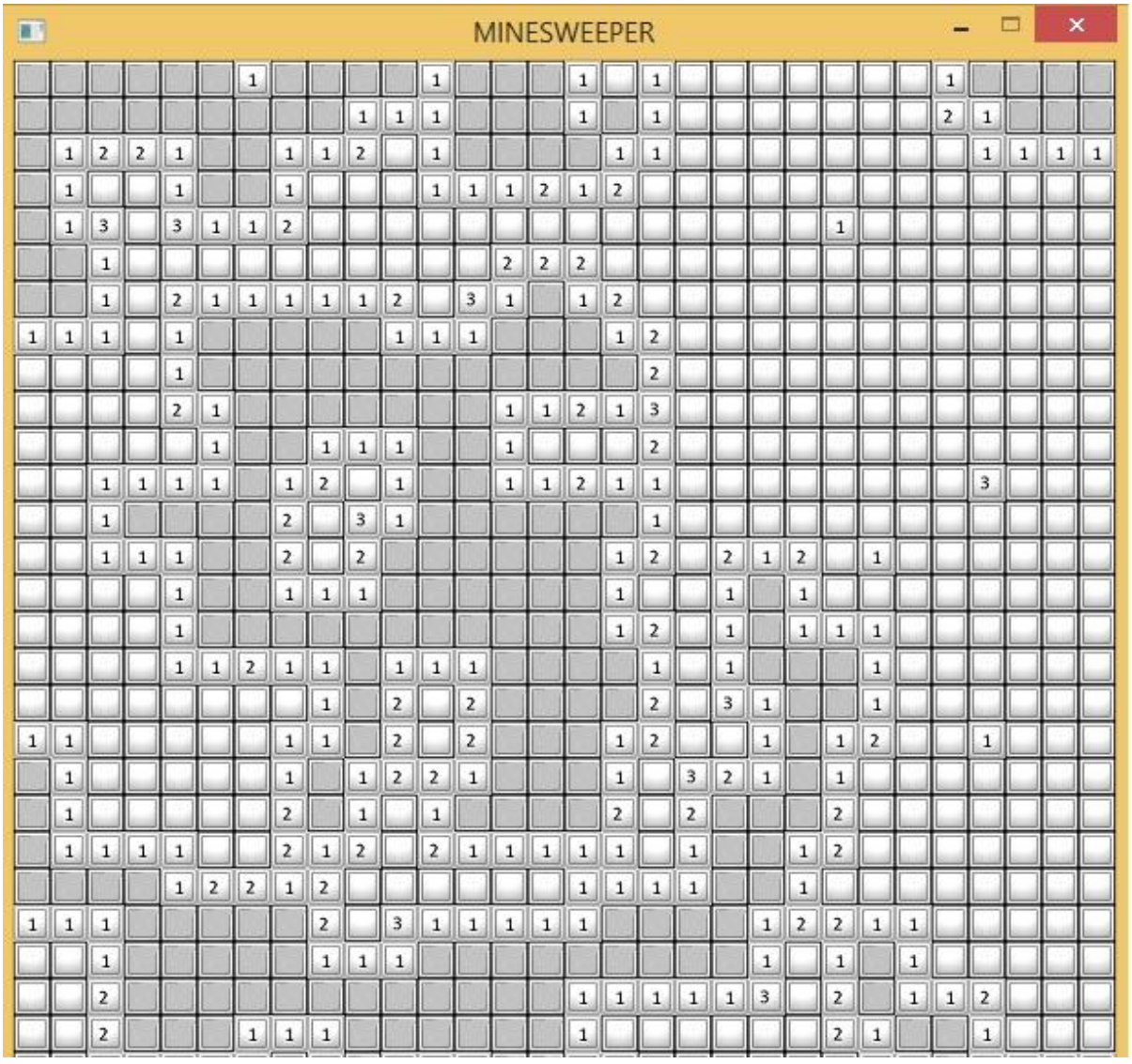
1. This is the window that opens when the user runs the program.



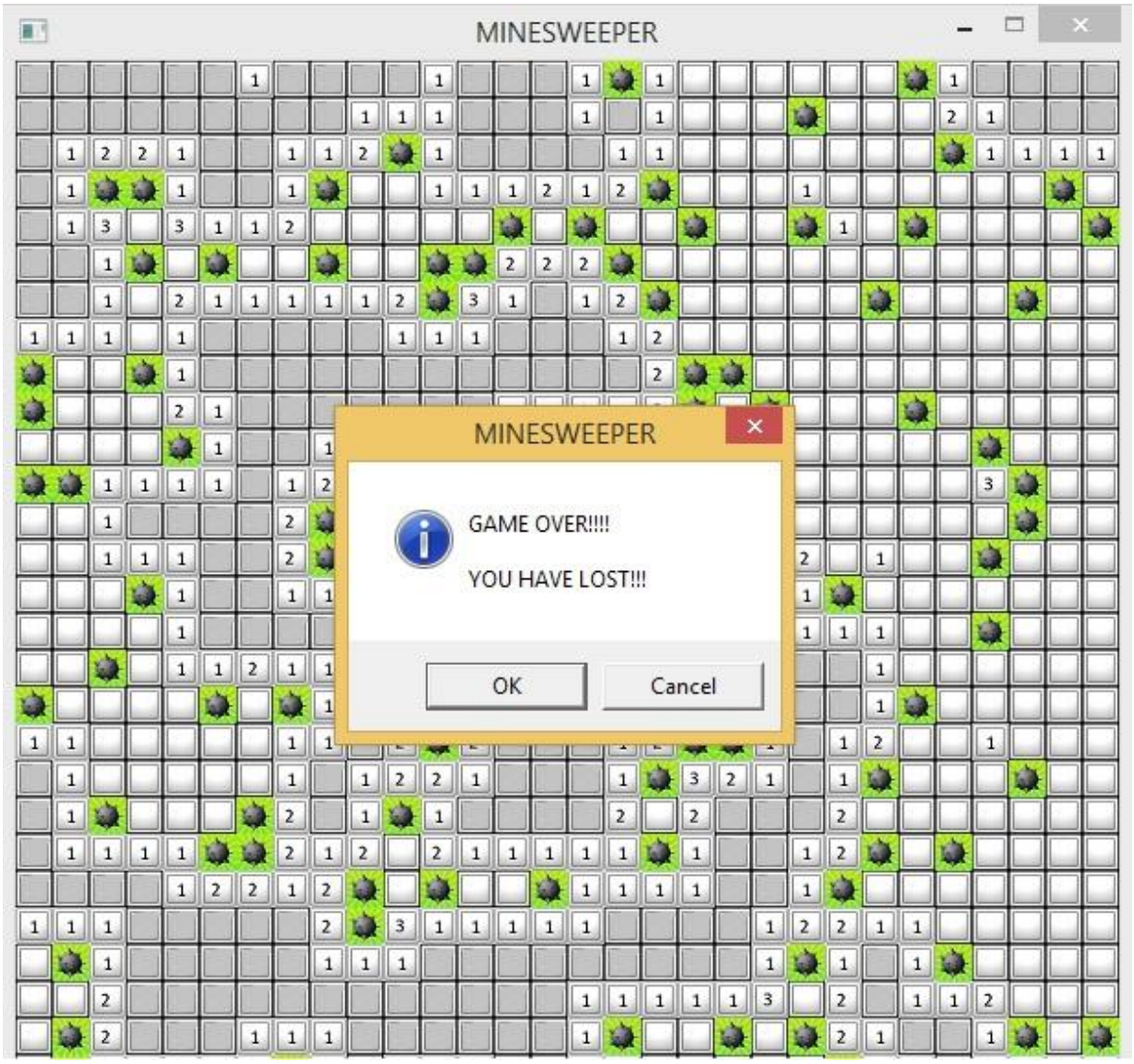
2. Once the user selects ok, the game screen pops up



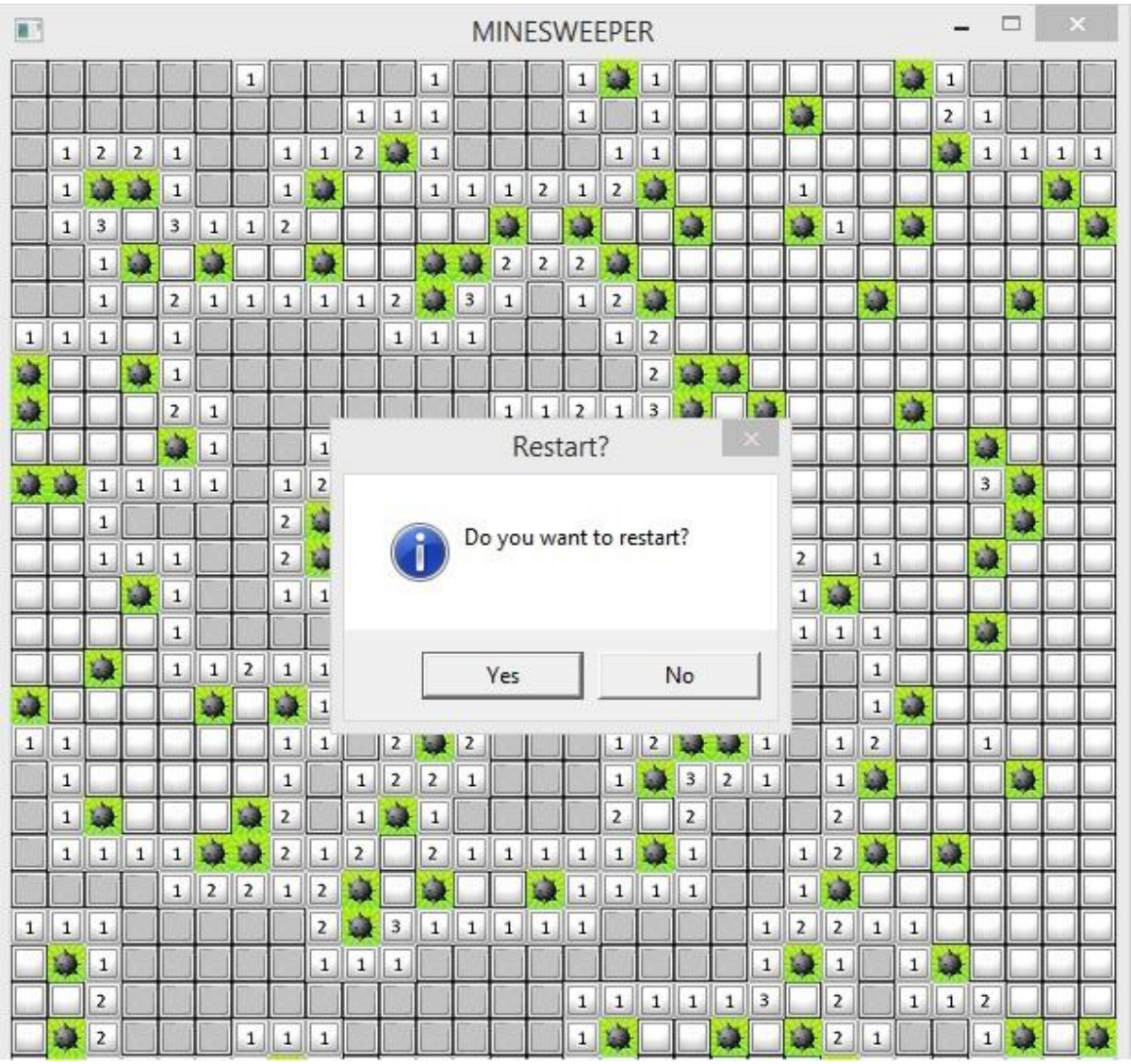
3. When the user clicks on certain tiles which did onot contain mines.



4. When the player clicks on a tile containing the mine and henceforth loses.



5. The ”Restart game” window



6. Discussion of System

A) What worked as per plan?

1. The entire algorithm worked as we had decided.
2. The gaming window also opens as we had desired (that too with improved graphics)
3. The function which is responsible for re-launching the game if lost and if the player wants to play again worked as planned.

B) What we added more than discussed in SRS?

1. The quantum improvement in our project was using allegro instead of simplecpp which we had mentioned as a challenge in the SRS.
2. Additional features like input via both keyboard and mouse (on the welcome and exit screen) has been added.

7. Future Work

Many additional features can be added such as:

1. Flags can be introduced in the game. If you think there is a mine in a particular tile you can place a flag on it as a mark.

2. Sound can be added, whenever a mine bursts.

3 .Difficulty levels can be added.

8. Challenges faced and conclusion:

The main challenge was learning how to work with allegro .Understanding the functions and syntax used in this high level gaming GUI was not easy. Also, managing time while balancing other course work during the semester. Things don't always work as planned so coming up with alternatives was also a big part of this venture.

We overcame the above by watching and learning through the tutorials available for allegro on youtube and dividing work efficiently such that every member had enough time for his/her allotted work.

It was a fun learning experience as we got to learn a complex though highly useful interface which can be used for making many other games in future also.

9. References

1. The Youtube channel(for allegro)

   https://www.youtube.com/user/CodingMadeEasy/featured

2. Previous year projects.

3. Other webpages to find the syntax for commands in allegro.