Table of contents

The software:

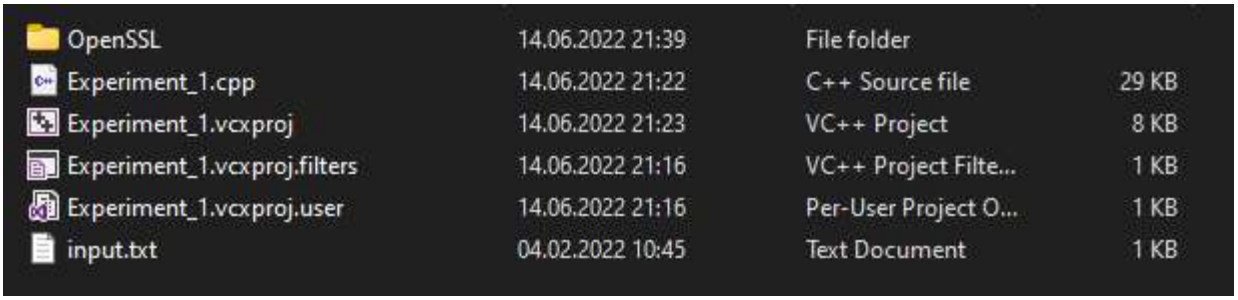
Check that you have Visual Studio 2019 software.

a) If you do not have a particular version, check (especially older versions) that the following instruction points execute smoothly.
b) If you are using a different development environment (e.g. dev-cpp) you should connect the OpenSSL DLLs explained on the visual studio example in point 3. (In case of errors)

Archive with programs:

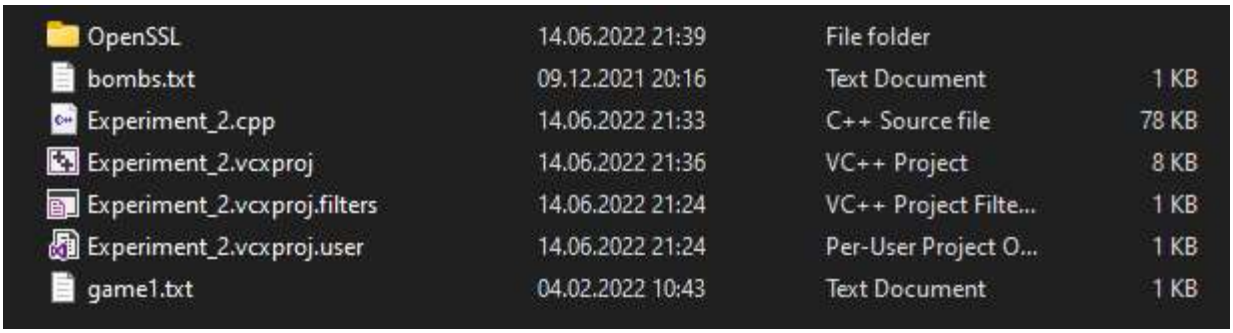
Extract the attached files to any location of your choice and proceed to check the contents.

a) In the case of the first task, depending on the variant, there should be a Visual Studio 2019 solution file with an input.txt text file. In addition, the obfuscated code should contain a directory with the OpenSSL library.



1 Content of Experiment_1 folder for task 1 in the obfuscated variant

b) For the second task, there should be a solution with the OpenSSL directory.

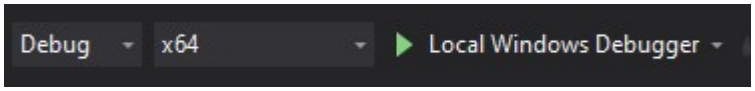


2 Contents of Experiment_2 directory for task 2

If the files are incomplete, check that everything has been extracted or download the archive again.

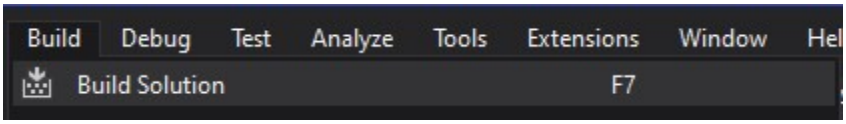
Checking the operation of the programs:

1. Run the Experiment_1.sln file
2. Set the project option to Debug and platform to x64



3 Settings 3.2

3. Build the project using the F7 key or by hovering over the tab at the top *Build* and *Build Solution*



4 Settings 3.3

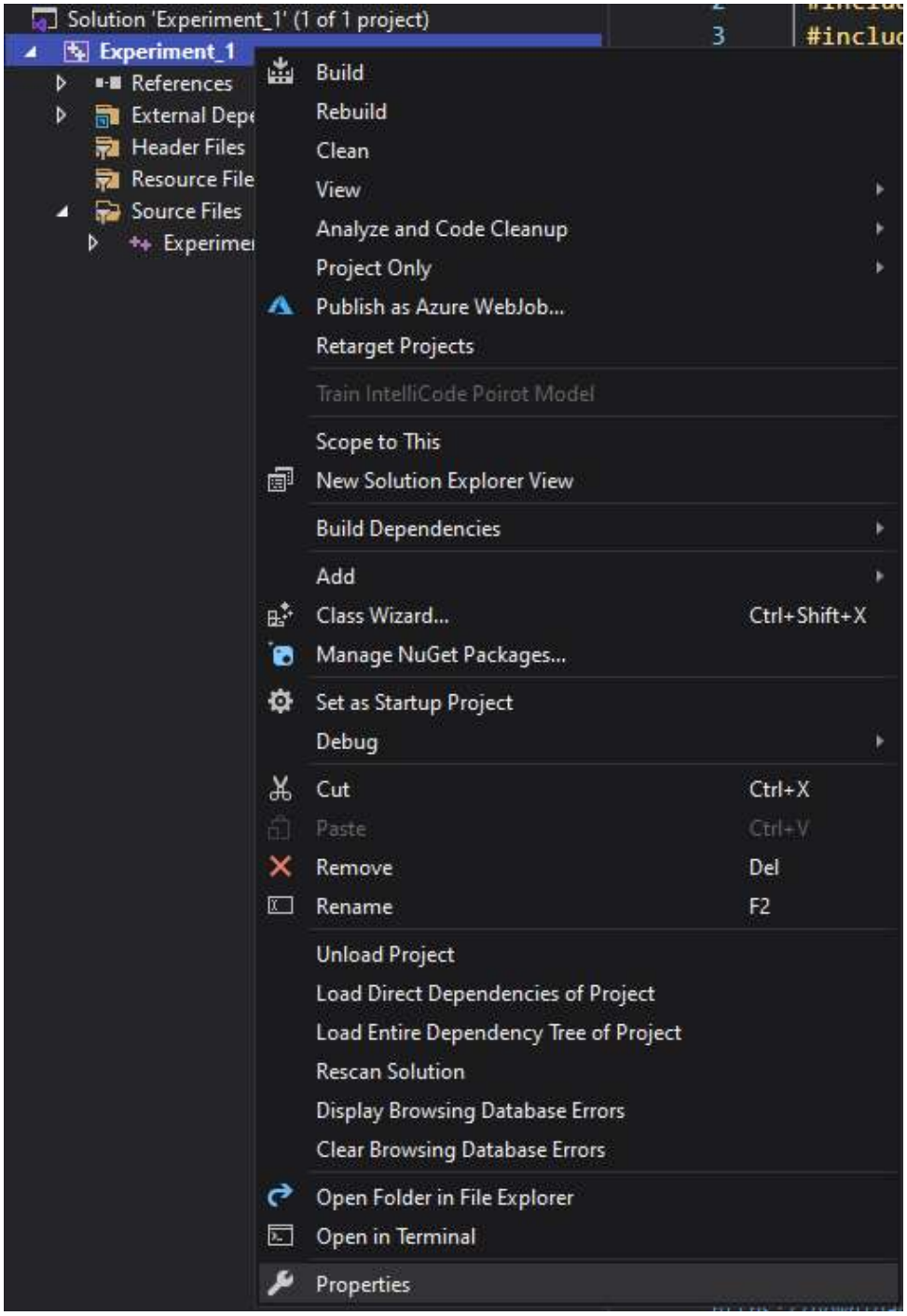
4. Start the program with F5 or by clicking on the green triangle pointing to the right (image 3)

5. If the program starts without any errors, everything works as it should.
   Repeat steps 1-4 for program 2 to check if everything works as it should.

In case of errors:

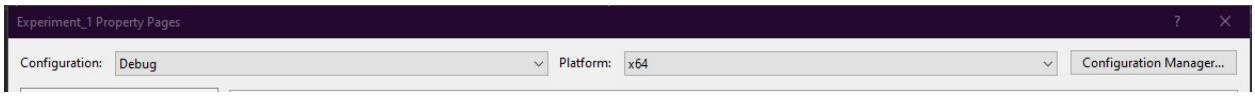
a) In case of missing aes.h header:

1. Enter the solution options by right-clicking on " Experiment_1" in the solution explorer on the left.



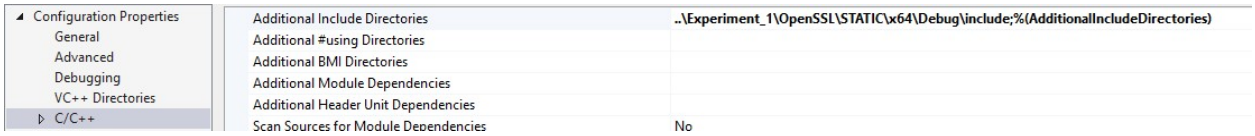
5 Open project properties

2. Ensure that "Configuration" and "Platform" is set to "Debug" and "Active(x64)"



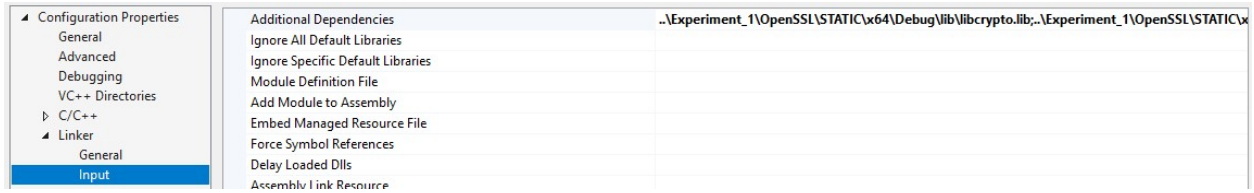
6 Check platform settings

3. Go to "C/C++" and then "General" and under "Additional Include Directories" delete the content and paste
   ..\Experiment_1\OpenSSL\STATIC\x64\Debug\include;%(AdditionalIncludeDirectories)



7 Place where to connect libaries

4. Confirm with enter and click the left mouse button on the "Apply" button
5. Go to the "Linker" tab and then "Input"
6. In the "Additional Dependencies" tab, delete the content and paste
   ..\ Experiment_1\OpenSSL\STATIC\x64\Debug\lib\libcrypto.lib;
   ..\ Experiment_1\OpenSSL\STATIC\x64\Debug\lib\libssl.lib;%(AdditionalDependencies)



8 Miejsce gdzie należy wkleić kolejną zawartość

7. Confirm with enter and left click on the "Apply" button
8. Exit the settings

   Note: For program 2 change *Experiment_1* to *Experiment_2*

b) In case of incorrect variable name definitions
   1. Copy the contents of the code using "CTRL + C" shortcut
   2. Paste into the main file in Visual Basic Studio using "CTRL + V"
   3. Agree to use UNICODE by Microsoft Visual Studio

Instructions for programs:

Have a piece of paper or notepad available to record the start and end time of the task and the answer to the task.

For RandomNumbers [Experiment_1]:

Description: The file takes data from the input.txt file and creates something using it.

a) Read and analyze the source code
b) Analyze the input.txt file, which the program takes as input
c) Solve the following problems and write down the solution times. The time allocated to the task is 1 hour:
    1. What is the end result of the program?
    2. What process takes place during the execution of the program. Give the name of the algorithm if you can.

For Minesweeper [Experiment_2]:

Addition: The program is a simple version of the game Minesweeper. It implements typical functionality like flagging fields where a bomb might be, or uncovering fields if there are no bombs around.

a) Read and analyze the source code
b) Analyze the files bombs.txt and game1.txt
c) Solve the following problems and record the solution times. The time allocated to the task is 2 hours:
    1. Change the code to load the bomb distribution from the file game1.txt
    2. Make the map uncovered from the beginning, but the gameplay continues normally
    3. In case of more time:
        a) Set in the code the player wins after any field is discovered.
        b) Make hovering over a bomb using Expose mode to turn that bomb into a flagged bomb.

After completing the tasks, you will need to answer the end of the experiment questionnaire provided with the files.