ENGR:2730 Computers in Engineering
Christensen and Chu, Spring 2021, Homework #5

Deadline: Friday, April 2, 2021 by 11:59pm
Early turn in bonus deadline: Wednesday, March 31, 2021 by 11:59pm

SFML reading for this assignment: Events explained, Keyboard, mouse and joysticks, Text and fonts, Shapes, Playing sounds and music.

In this homework, you will add new features to your Xonix game that you started working on in Homework 4. Copy your hw4 folder from your homework directory and paste it back into your homework directory to create a new folder called hw5. **IMPORTANT: Please do not to add your cmake-build-debug or .idea directories to SVN when you copy hw4 to hw5.**

IMPORTANT: You must have permission to use any fonts, sounds and/or images that you use in your programs. This means that you need to cite where you got the work, whose work it is, and the license saying that you can use it. Thus, you should go to one of the many web sites with free material and cite the permission/license that they grant you to use their work.

Here is an example, if I want to use a font for my program, I may go to the web site "1001 Free Fonts" and want to use the Fondy Script font. The website says this font is free for personal use. This means that you can use it for your homework, however, you would need to add the following comment to your code to use it.

// Fondy Script by Måns Grebäck, https://www.1001freefonts.com/, License: Free for personal use.

We will deduct points from your homework if you do not cite your permission to use a work in a comment in your program. Please feel free to ask questions about proper attribution / usage if you have any!

Part 1. [25pts] Add sounds your Xonix game to play background music and to make a game over sound. Feel free to add more sounds to your game. **DO NOT ADD .WAV FILES TO SVN and DO NOT ADD SOUND FILES LARGER THAN 100KB TO SVN.** Only add .ogg or .flac files to SVN. NOTE: .ogg and .flac files are compressed and are much smaller than .wav files. You should use Audacity (see note below) to convert sound file formats into .ogg or .flac and to trim them to a shorter length of time.

1. SFML supports the audio file formats WAV, OGG/Vorbis and FLAC. Due to licensing issues, MP3 is not supported. Make a "sounds" directory in your hw5 project and save your music in that directory.
2. Add **#include <SFML/Audio.hpp>** to the top of your main program.
3. Add a sound (e.g., https://www.zapsplat.com) to your game using the SoundBuffer, Sound and Music objects. Read the SFML 2.5 tutorial how to do this https://www.sfml-dev.org/tutorials/2.5/audio-sounds.php. Sound objects are short, i.e., approximately 1-3 seconds. Music objects are larger and stream from disk. Do a web search to find free game sounds. Make sure to look at the license and cite were the sound comes from. The license may have requirements that you have to follow to use the sound.
4. Modify your CMakeLists.txt to copy the sound files from your project directory to your build directory. See how this is done for the texture image files as an example. Remember to reload your CMakeLists.txt file after you make changes to it.
5. Make sure to check that your sound file was loaded correctly. If the sound fails to load, your program should return EXIT_FAILURE (same as with textures in hw4).
6. Once your program works with the sound (and your sound file is smaller than 100KB), check it into SVN.

NOTE: Audacity is installed on the college of engineering computers including on the VDI. It is a free program that you can install on your Mac or Windows computer to edit audio files. Audacity allows you to change the bit-rate of sound files, shorten the length of sound files, concatenate sound files and export sound files to .ogg format. Changing the bit-rate of sound files from 44 kHz to 8 kHz will make sound files smaller.

Part 2. [25pts] Add score keeping to your game.

1. Do a web search to find a web site with free fonts (e.g., https://www.1001freefonts.com/). Select a font. Download it. Make a directory in your hw5 project called "fonts" and move your font file into the fonts directory.
2. Modify your CMakeLists.txt to copy the font file from your project directory to your build directory. See how this is done for the image files as an example. Remember to reload your CMakeLists.txt file after you change it.
3. Follow the SFML tutorial https://www.sfml-dev.org/tutorials/2.5/graphics-text.php to learn how to add text to your program.
4. Make sure to check that your font files were loaded correctly. If the font file fails to load, your program should return EXIT_FAILURE (same as with textures in hw4).
5. Add #include <string> at the top of your program to use the std::to_string(score) command which converts some integer score to a string. Remember you can concatenate two strings together with the plus operator.

Part 3. [25pts] Remove all the continue commands from the your Xonix program without changing the logical operation of the program. Continue commands breaks structured programming and makes your code more difficult for others to read. In nearly all cases, code is more understandable to others if it does not use continue commands. You should not use break statements to remove continue statements either. There are different ways to remove continue commands. Sometimes you can replace the continue with an if-statement and other times is better to change a for-loop to a while-loop. For each continue statement that you remove, please choose the method that you prefer to remove it with the intent of making your code more understandable to others. Note: it is very easy to break your code when you remove a continue statement. Thus, it is recommended that you check your working program into SVN before removing the continue command. This way you can revert back to your previous working program if you make a mistake.

Part 4. [25pts] Add your own feature to your game. **IMPORTANT: YOU MUST ADD COMMENTS TO YOUR CODE EXPLAINING WHAT YOU ADDED FOR THIS PART.** You are welcome to add more sounds or change out the images, but this will not give you credit for this part of the assignment. Please get prior approval from a professor or TA if you are unsure if your planned additions will qualify for full credit. **EXAMPLES** of acceptable additions include, but are not limited to:

1. Add a welcome and instruction screen at the beginning of the game. The display should have good instructions. The program should pause on this screen and prompt the user to continue onto the game. HINT: You can use a state variable to change the state of your game. HINT: Use a state machine (See Book Section 4.12).
2. Add multiple levels to the game that get harder. One way to make the level harder is to start the original board with more than three unique gems. HINT: Use a state machine (See Book Section 4.12).
3. Add a high score page after the game ends each time. The program will need to keep track of the previous 5 scores either in memory (cleared each time the game runs) or in a file (which would be persistent between game runs). HINT: Use a state machine (See Book Section 4.12).
4. You can add a button to give hints.

Submission Instructions

You must make a CLion project called "hw5" under your SVN homework directory. Check your homework into SVN.

Hint: you can see the current version of your submission by opening this link in a web browser:
https://class-svn.engineering.uiowa.edu/cie/projects/spring20/