Quest Six – Jailer Vs. Player

Programmer: Natalia Burbano

Completed: 10/11/2019

Standard Code

In this program, the user is playing a game against the computer. The player has a token that is placed on a hypothetical gameboard that contains a total of 16 spaces. There is a "start" place, which I called "Space 0" in my program, and then the rest of the spaces consist of the numbers 1 through 15, increasing in numerical order. For this game to function, it requires the generation of random numbers. In order to ensure this, I initialized time as my random seed value in the beginning of my code. The game begins by the computer randomly generating a random value between 1 and 15, which will determine the placement of the silver coin. Then, the user inputs their desired location for their gold coin. However, the program does not allow the user to place their gold coin in the same spot as the computer's silver coin. The user then "rolls the die", which just means a random number between 1 and 6 is generated. Throughout the game, the value of the die that is rolled and the location of the token are outputted to update the player. With the use of a while loop, the program will continue to roll the die until the player's token lands on either of the coins' spot. Also, if the token passes Space 15, its location is automatically reset to 0-the start location. The game will only end once the token lands on a space occupied by either the computer's randomly placed silver coin or the user's gold coin. If it lands on the gold coin, the player wins. If it lands on the silver coin, the player loses. At the end, a message celebrating the player's victory or failure will be displayed onto the screen.

Extended Code

In the extended version, the program contains an additional silver coin. Now, one of the silver coins must be randomly chosen before the player "rolls the die" and change its location to another random location. To accomplish this, I created a new variable that would store random values, using time as my seed value again. If the random value is even, the first silver coin is chosen to change its location, but if it is odd, the second coin is chosen. For this to work, I had to reset the random value in my variable "chooseCoin" so that it continues to generate random values every time it goes through the loop. The rules of the standard code, in terms of where the coins can be placed, still stand. I added additional while loops to make sure the silver coin that is being assigned a new, random location is not accidentally placed in the same spot as another silver, or gold coin.

Figure 1: Standard code - Player wins

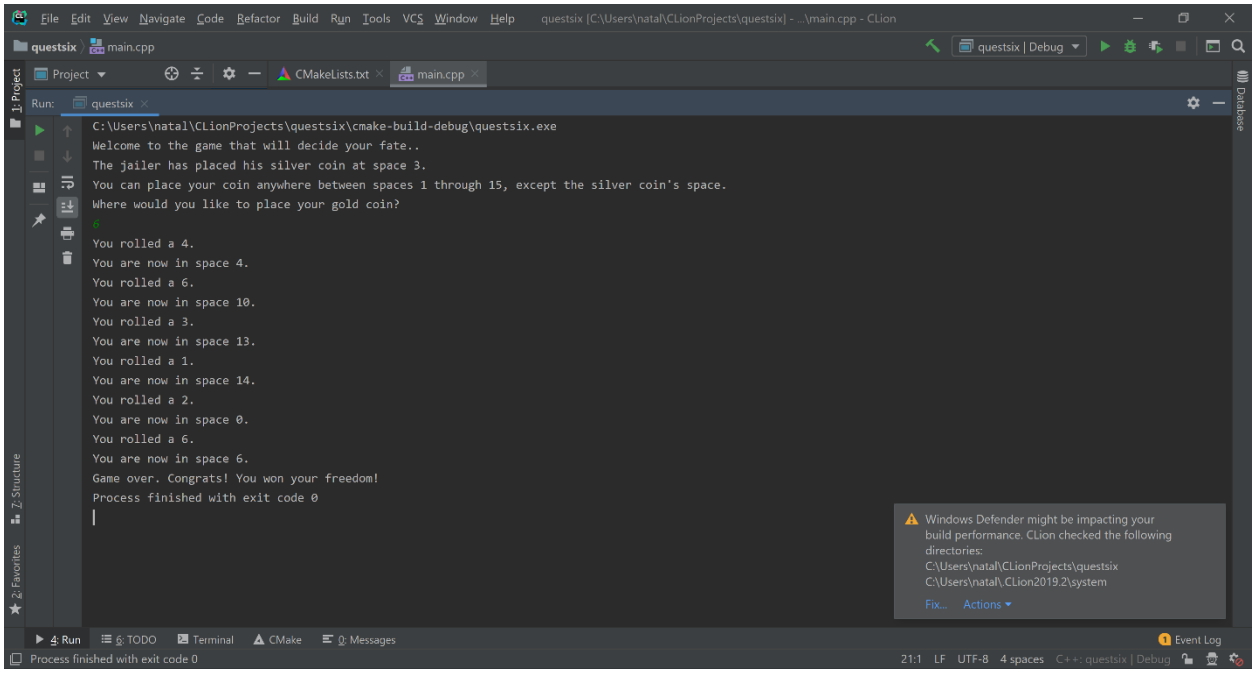


Figure 2: Standard code - Jailer wins

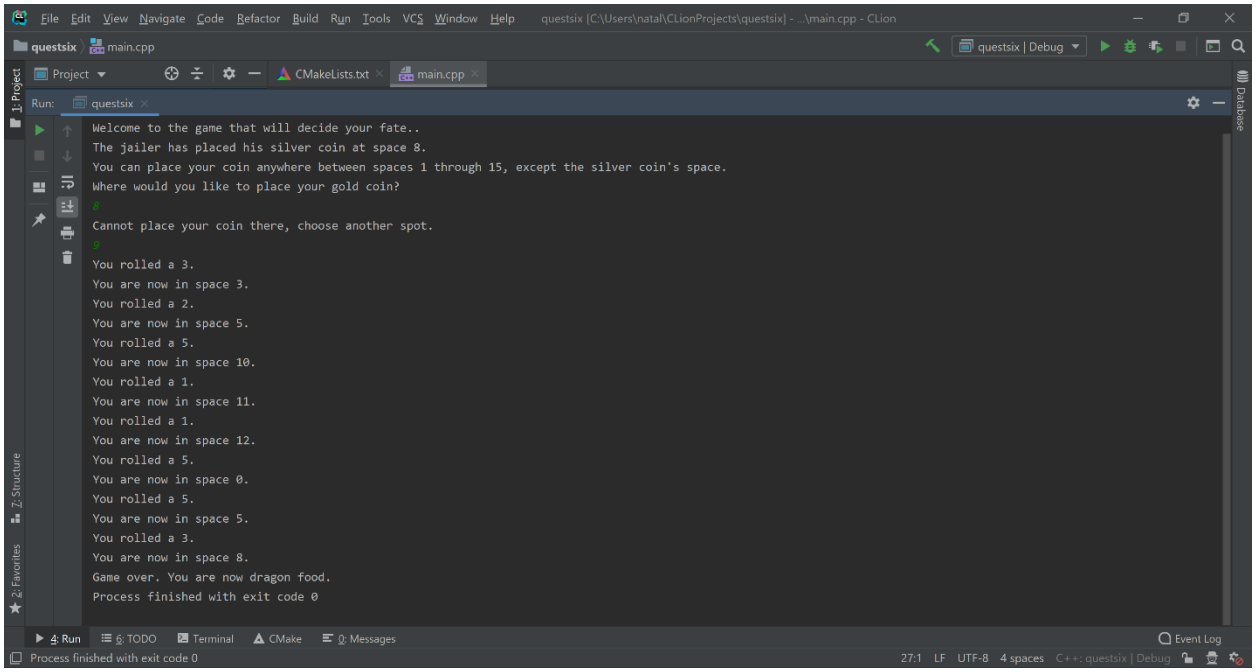
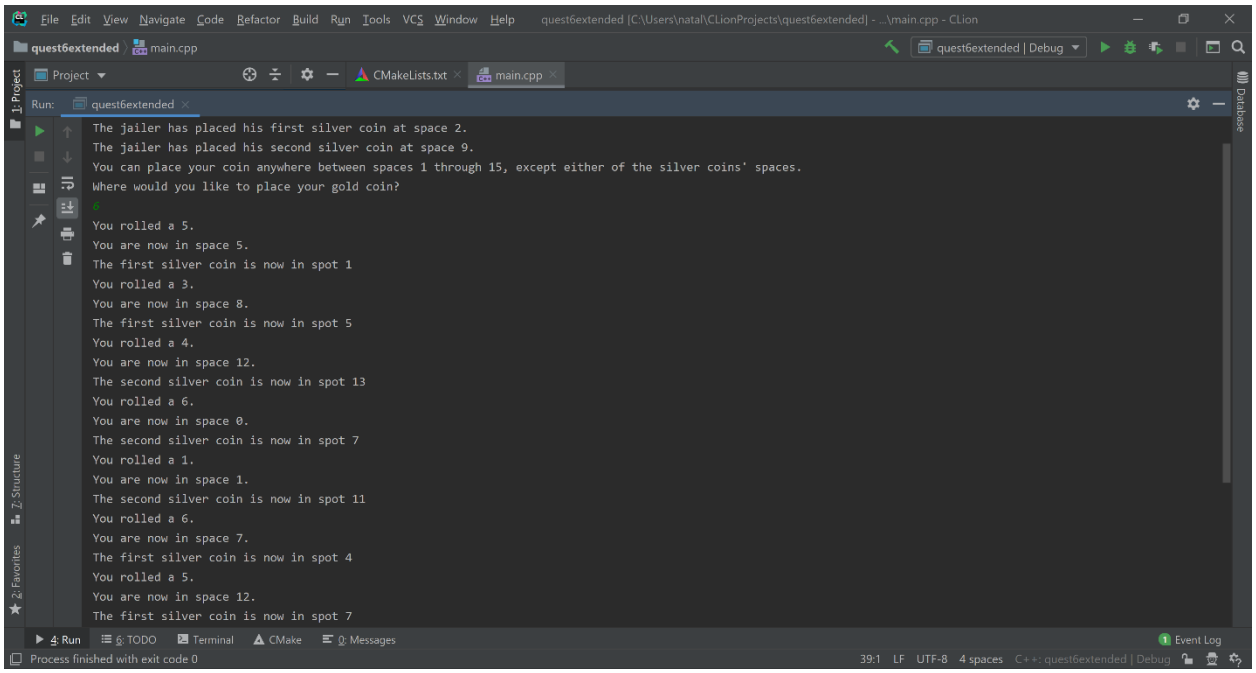


Figure 3 and 4: Extended work - Player wins



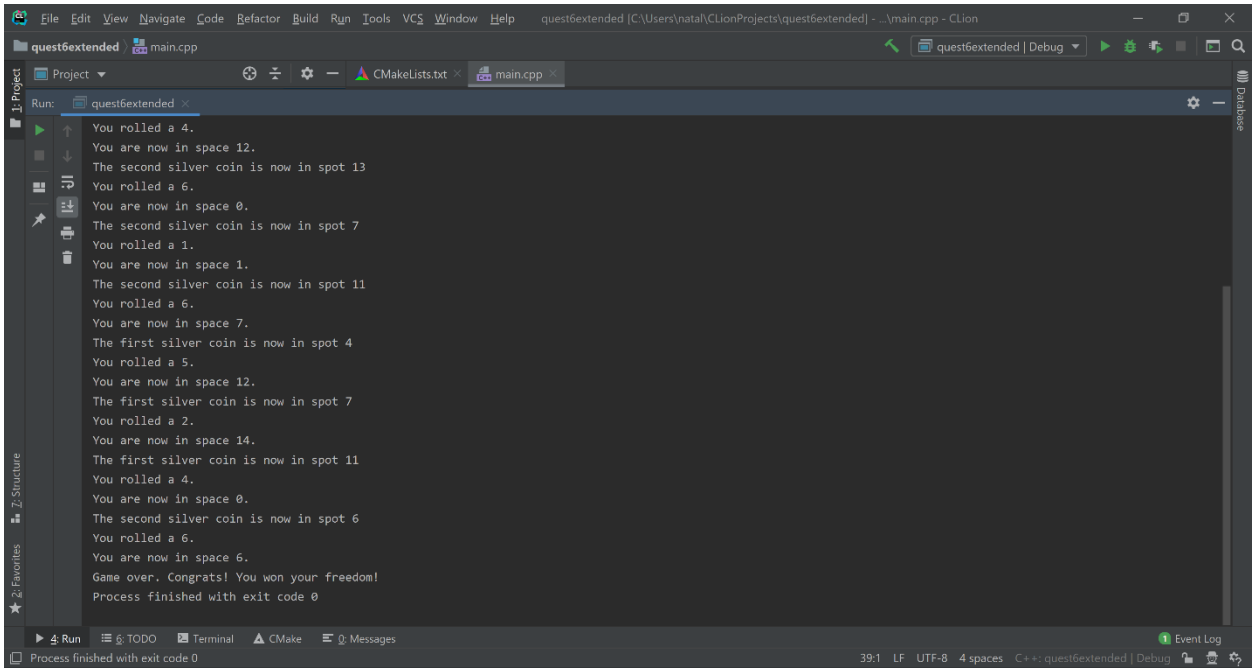
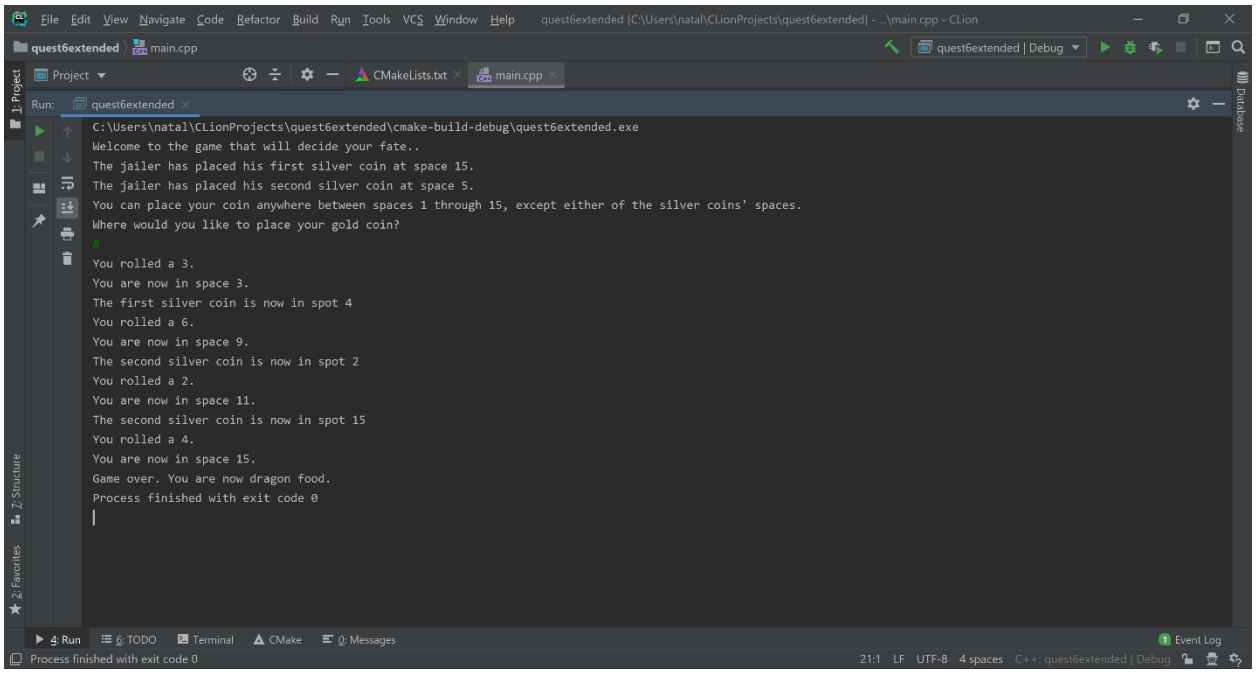


Figure 5: Extended work - Jailer wins


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
quest6extended [C:\Users\natal\CLionProjects\quest6extended] - ...\main.cpp - CLion
quest6extended / main.cpp
quest6extended | Debug
Project
CMakeLists.txt
main.cpp
Run:
quest6extended
C:\Users\natal\CLionProjects\quest6extended\cmake-build-debug\quest6extended.exe
Welcome to the game that will decide your fate..
The jailer has placed his first silver coin at space 15.
The jailer has placed his second silver coin at space 5.
You can place your coin anywhere between spaces 1 through 15, except either of the silver coins' spaces.
Where would you like to place your gold coin?
8
You rolled a 3.
You are now in space 3.
The first silver coin is now in spot 4
You rolled a 6.
You are now in space 9.
The second silver coin is now in spot 2
You rolled a 2.
You are now in space 11.
The second silver coin is now in spot 15
You rolled a 4.
You are now in space 15.
Game over. You are now dragon food.
Process finished with exit code 0
1: Project
2: Favorites
7: Structure
Database
4: Run
6: TODO
Terminal
CMake
0: Messages
Event Log
Process finished with exit code 0
21:1 LF UTF-8 4 spaces C++: quest6extended | Debug