Spike: Task 4.P
Title: Gridworld Multi-Threaded

Author: Mitchell Wright, 100595153

Goals / deliverables:

Produce a working Gridworld game, according to the provided specification sheet and develop and understanding of threads and concurrent loops.

Items created during task:

- Code, see: \03 - Spike – Gridworld Multi-threaded\GridWorld\

Technologies, Tools, and Resources used:
List of information needed by someone trying to reproduce this work

- Visual Studio 2022
- SourceTree
- GitHub
- Lecture 2.1 – Game Loops & Software Architecture

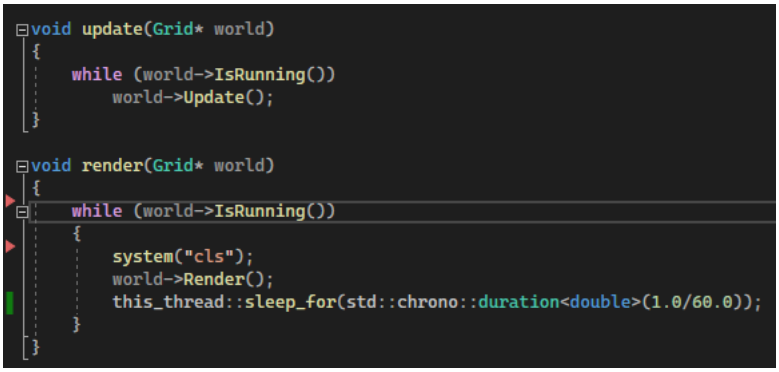
Tasks undertaken:

- Implement Loops
- Use Threads
- Complain About System(“cls”);
- Commit to Git

What we found out:

1. Implement Loops:

   The separate loops in this program were pretty easy to make. Update is essentially unchanged and render has a quick nap at the end of each loop to prevent flickering.

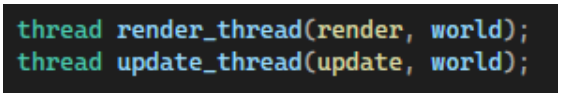


```
void update(Grid* world)
{
    while (world->IsRunning())
        world->Update();
}

void render(Grid* world)
{
    while (world->IsRunning())
    {
        system("cls");
        world->Render();
        this_thread::sleep_for(std::chrono::duration<double>(1.0/60.0));
    }
}
```

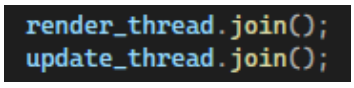
2. Use Threads:

   In order to create a thread, this syntax is needed.



```
thread render_thread(render, world);
thread update_thread(update, world);
```

   Then, once the threads are created, they need to be joined, so that the functions can be run.



```
render_thread.join();
update_thread.join();
```

   Then once the functions in the threads have finished the program ends, much the same as in the previous spike.

3. Complain About System(“cls”):

   It’s really slow.

4. Commit to Git: