Tutorial 1 - Introduction to C++

Computer Graphics

Adapted From: Kartic Subr and Brian Seipp

January 23, 2022

Part 1: Hello World!

Make sure that you have a C++ compiler installed g++ installed (that should be available out of the box in Linux, Mac and DICE machines). Run a program that prints 'Hello World' through the terminal. Extend the program to collect your name from the terminal and print your name as well. If you are not familiar with C++ you can follow the step here.

Part 2: Pointers and memory

An important feature of C++ is the ability for fine grain memory management. As such an important concept is the addresses of in memory where data is stored - pointers. This in C++ is primarly controlled via **Address-of operator (&)** and **Dereference operator (*).** The reason why this is important is when it comes to optimizations of different operations. For example when passing information to a function, one can pass the address of the data, rather than copy the data instead. You can read more about the topic here. Make sure you understand pointers.cpp and pointers2.cpp. Try experimenting with the two operators(& and *) and make sure you understand, when are the two operators used.

Part 3: Life!

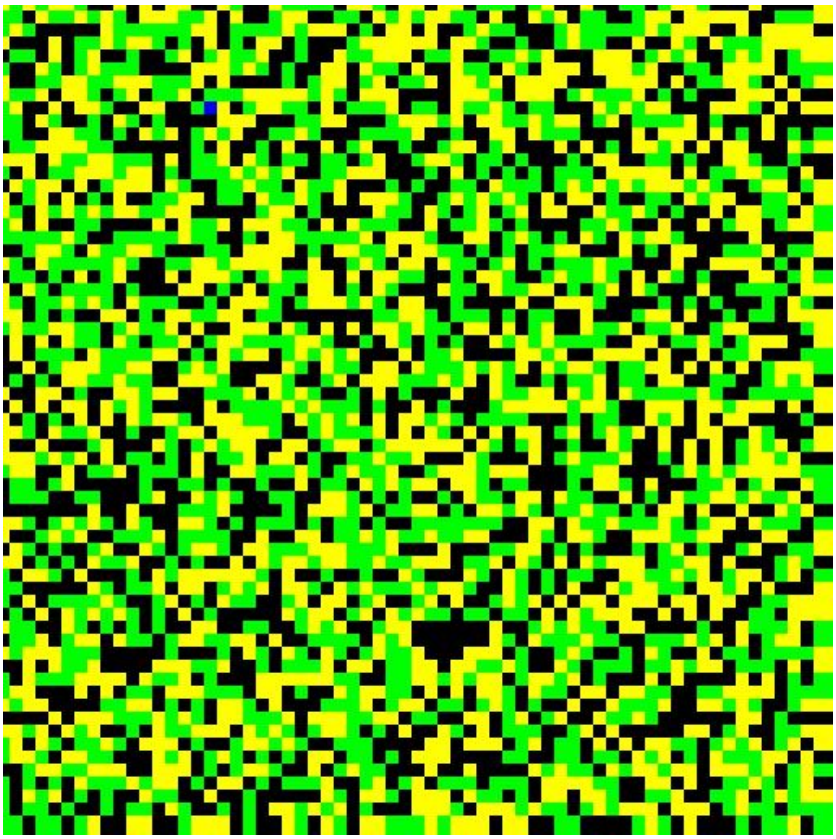
You open your eyes and discover you are in a totally new world! Even stranger you find that you can only move in 4 directions: up, down, left, or right! Ever the opportunist you begin to look for gold so you can buy a ride off this strange planet, but your stomach begins to grow and you realize you better look for food too!

This project will see you implement several components in a game of life. The goal of the game is to explore as much of the world as possible, mining for gold as you encounter it or eating delicious food so you don't starve to death. This game is implemented in 3 files. The first is driver.cpp. This is the main driver for the program. This has been fully implemented and can be left as is. Be sure to take a look to understand how main programs work as well as good examples of includes and how to allocate memory and instantiate new classes in c++.

The second is the agent.cpp file. The agent is responsible for tracking the health and loot you gather as you explore the world. Additionally the agent knows its own position in the world. As such the agent also takes actions to move around. These take the form of random or planned actions. You will be implementing the random and planned actions the agent will be taking. Additionally you will be implementing the eat and mine functions for when you encounter food or gold respectively. Finally you must allow the agent to start in a random

location each time you run the simulation. Further details for each function can be found by referring to the agent.h file including specifications on any validation you might need to do.

The third set of files is the world.cpp file. The world is is what the agent will be exploring. Through this class the driver will ask the agent to take random or planned actions. The world will provide additional validation such as ensuring that the agent does not walk outside its bounds. You will be responsible for implementing both the take_random_step function and take_planned_step functions in the world class. Additionally you will implement the createWorld function. This function populates the 2 dimensional world with values that correspond to the items (or lack thereof) you could find in the world. Finally you will implement parts of the log function. The log function will print an image that represents the world as your agent explores it. You will be responsible for specifying the RGB channel values to identify the various items in the world as well as the agent position. Finally you will implement the correct size of grid cells in the image as they correspond to the world. Further details of this are available in the world.h file. The resulting log will look something like this:



Where the yellow squares represent gold, the green food, the black nothing, and the blue box the agent. Each time your agent takes an action this function will be called by the run function, and an image generated. Try making a gif out of the resulting images and see how your agent explored the environment!