**SIDEKICK Writeup**

Brandon McKinley

A computer screen shot of a program code

AI-generated content may be incorrect.

A screen shot of a computer program

AI-generated content may be incorrect.I intended **Sidekick** to be a Drone Command application or simulation. With the ability for a user to keep the info of each drone on a battlefield or grid on hand for easy access. I had the thought to use ROS to simulate an environment, but I chose to focus on pure C++ for simplicity of learning and practicing in C++. It could be considered a simulation or a game. With version 1, I realized that I could just keep adding features to this application and it could be worked on endlessly. So, with version 2, I decided to focus on learning how to randomly generate a map size and objects to include their placement at random, then focused on the user menu and the drones themselves. The drone classes inherit from the base drone class while having their own functions specific to them. I was wanting the map to be different each time and not exactly the same for each run of the program.

I think that my vectors and pointers played a large role in my program because I found that I needed to access various types of data in each of my files. For example, when working with my drones, I needed to be able to access specific members of new drone objects that the user is allowed to create. Pointers where kind of tricky to figure out but I believe I was using them correctly, because I wanted to access information for a specific object without changing that information when it is passed through my functions. Such as the map handling the removal of the drones.

A screen shot of a computer program

AI-generated content may be incorrect.Since my data sets would be small and based on the drones battery, I used a simple bubble sort function to sort them based on their battery life. It also stores the sorted drones into their own vector.

A screenshot of a computer code

AI-generated content may be incorrect.

I think my map class became the main carrier of the program because I needed it to keep track of its own size, the objects within it, such as the drones or buildings, and handle what objects fall within a certain type. I ended up using an enumerator for storing simple objects such as what was the grid content and what objects would be generated on the map. I used this because I only need simple constants that were always public and I didn’t need them to have a specific data type.

I learned that a better more modern way of truly generating random numbers, compared to srand, was using the <random> library, this allowed me to create a seed and engine to distribute the size of the map with a minimum size of 10x10 or a max of 40x40. This allowed me to get the random map size I wanted each time a user would run the program. This was to allow an environment size to be different each time. This was also applied to the generate objects function within the map class to achieve similar results but also includes the random placement of the objects based on the generated map size. I also learned how complicated a program could get when implementing various classes that need to work together in order to achieve the desired outcome, so using resources like YouTube or stackoverflow was helpful. Im sure there are simpler ways of doing what I did and having pointers to help grab the data my program needed was very helpful. I think in the future, I will be able to have a more detail and clear path for the next program because I struggled with sticking with my plan and not adding to many features as I went along.