1. **Introduction**

For our final project we decided to go with a platformer game. For context, everyone in our group has some sort of background in gaming. Despite all coming from different gaming backgrounds we decided that we would all be interested in creating one. The simplest and best middle ground: a 2D platformer. Our game is based on the popular steam game called Jump King. The concept behind the game is having a main character that has the purpose of jumping his way to the top. What’s at the top? Who knows (it’s a Princess). The main character has to stretch through different environments and mechanics to make his way to the top.

1. **Decisions Made**

The first decision we made before anything was what engine we were going to run the game on. Many in the group wanted to use unity since it was a tool that would be used in the real world. This game with its own challenges since we would have make adaptations to using code in C#. Additionally, we would have to learn how to use the software completely. The other choice was to use GameMaker, a web browser. This would be exponentially easier since there were more resources available. We decided to use unity despite being difficult for two reasons. First, yes it would be more difficult but we wanted to get real world experience since many of us are curious in game development. Second, we decided that we wanted to consider “learning” unity as a part of the project. A lot of our initial time went into learning the software and bypassing a relatively steep learning curve. Coming out of this project we are all glad to be able to have learned a sophisticated game software like Unity.

Another decision that we made as a group was the methodology for sharing code. Although github may seem like an obvious answer, there was much more to consider that will be discussed later on. In the end we ended up falling back on github for immediately sharing code.

A few smaller decisions included, how to pan the camera with the character, allowing the character to walk, how many sections / levels to create, what different materials or implementations should we include. We decided to pan the camera based on the length of each segment. Everytime the character moves we look for the position of the camera and the position of the main character. If the character goes above the top boundary we shift the camera upwards by a fixed amount and vice versa for hitting the bottom boundary. We wanted to make a good variety of levels but at the same time not to many as to overwhelm ourselves as it would require us to brainstorm different ideas again while also looking for appropriate assets for the idea. We decided on making 10 total level or sections. We decided on a variety of different implementations to add more to the game rather than simply jumping. For material we decided on normal ground, ice, and snow. We came up with two terrain / weather mechanics: wind, which appears in all the ice areas, and low gravity for the level where we enter space. In our final levels we decided to implement moving platforms in the last sections of the game. For our last level we created a game completed segment. For completionists, we added a coin base system that involves going around the game and collecting all the coins.

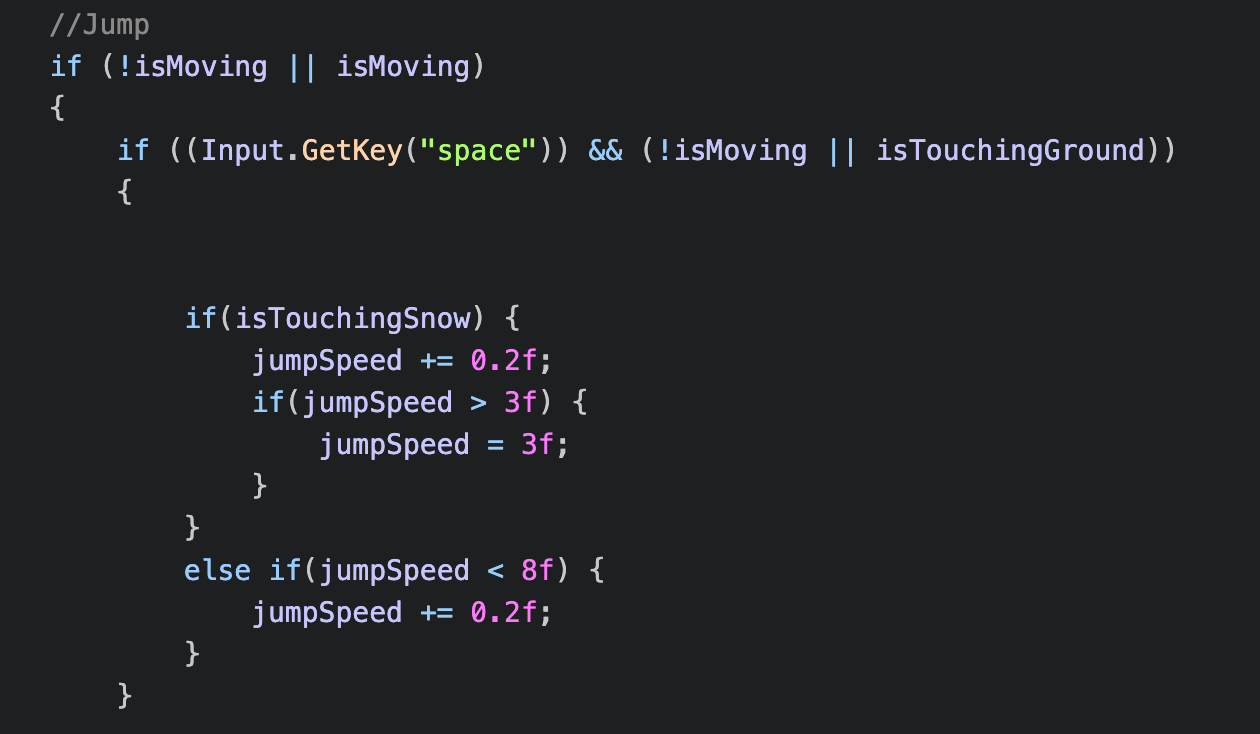
1. **Challenges**

The biggest challenge of the entire project was quite frankly using github. Specifically, using github with unity. Starting this project we had no idea how we would be able to share our code. Unity makes the process complicated because our repository would include user setting folder and files. Additionally, some files are too large for github. To overcome this error we had to (along with an enormous amount of research) we downloaded something called git lfs. After creating our template, git lfs would add two files to our repository: .git ignore and .git attributes. The .git ignore choses which files to exclude from our repository. The .git attributes looks at which files are too large and essentially compresses them for git distribution.

The second most relevant challenge was learning how to use Unity. No one in our group knew how to use Unity prior to this project. We all had to spend time learning the different in’s and out’s of the program while figuring out the differences between C++ and C#. On top of that, this was all of our first times experimenting with game development.

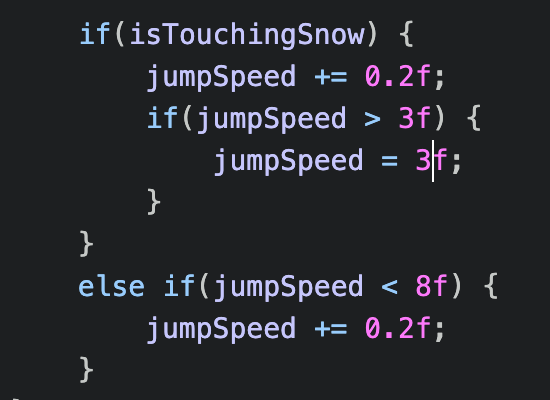
1. **Core Algorithms**

One of the most important sections of code is of course none other than the jump code. The jump code determines how long the user has been holding the space bar and then allows the user to perform an actual jump once they release the jump button. Most of the code will loop through these statements again and again. For example how long we hold space continues to affect the speed of the jump.

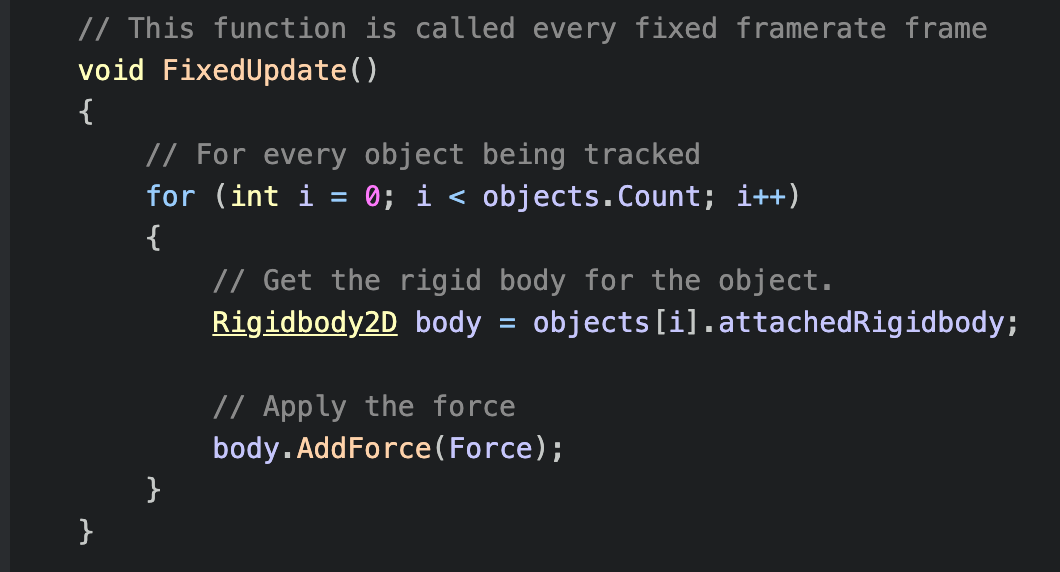


The large sum of code that we implemented was creating interactions between different surfaces and weathers. For example, the way a character interacts with ice surface makes it so the user just slides.

Snow makes it so that the user has a much weaker jump than normal:



Wind in icy sections affects how the user moves either against or with the wind:



And gravity makes it so the user stays in the air much longer than normal.