Our project will take in data from the Divvy and run it through our program to determine when bikes should be moved from an overpopulated station to an underpopulated one. The data gives us a list of bikes and their check in and check out times, as well as which stations they move from and to. This will allow us to track how many bikes are at each station throughout the day. After reading in the given data we will read through the array of the information with a loop with a “clock counter” that will increment the time passed. Whenever the time equals the check-out or check-in time of a bike that bike will be moved to the appropriate vector. If a station is too full or too empty the bikes will be redistributed as needed.

This project mostly implements vectors. Because we had large amounts of data to continuously read through, a vector was appropriate as we could access each element by using an incremented loop. Although it is difficult to remove or add data in the middle of a vector this was not a problem from this project as we only cared how many bikes were in a station vector, not which bikes they were. The only vector which specific data was need from a bike was the vector of the initial data. Because this data was only used to move bikes within station vectors, it was unchanged. Because of this vectors proved to be the best choice of data structure.

Using the Code:

When you run the code the functions should run through without in user interaction.

IDEs:

Xcode

CodeBlocks

OSs:

IOS

Windows