The team integrated code by a top-down integration method. First, we have created a based on the JavaScript file to be able to create subclasses below it, it would start at the top where the user would input any variables to change in our project 4. Integration of code from other members would start once a base code is finished, such as a stack where once the basic functionality is implemented subclasses would be added below it such as peek, push, and pop to allow the team to implement the function after the back of the stack is finished. Our integration always begins with subclasses that are created after a superclass is made by the team. Like the implementation with the queue, a superclass queue is made then subclasses that would take in information for the queue would change the variables that it is given to enqueue a value or dequeue a value from the superclass using the subclasses to change the variable given to it. The linked list data structure works the same way where our integration strategy would first begin with the linked list superclass that would take in an index and an integer to be able to put into the linked list. Then the subclasses would take over and insert and remove based on the user’s input of the variable to remove or replace using the subclasses created for the project. Finally, for a binary search tree, there is a main binary search tree superclass that would take in information and use the recursive subclasses to be able to change the variables to create a binary search tree for the user. From all that implantation of our data structure, our integration of code as a team is a top-down integration method