Matthew Bartrum

CS260

The hash table is another way to help our program to run faster. The hash table will store information to array by starting with key. With that we get values from 0 to minus 1 of the total value. An example of this would be if a key is 100, the value of that would be 0 to 99. To fill in this information with string it will turn that sting into a hash code so it will get an index for the array. How it does this is by getting the value of a string and dividing it by the elements in the array. With that information it should find a spot to go into. Hash tables. A big thing that happens while programming with a hash table is collision which is two items going into one place. One solution to this is linear probing. If a collision happens, the item will find an opening spot moving forward until it reaches the end, which at that point it will be sent to the beginning searching for a spot. Another solution to collision will be chaining. With chaining they will have pointers in a link list that would allow this list to add more items at the end of the linked list. Finding the item will be faster than linear probing. Overall, when you make the best hash function it will use the whole array and will give you the fastest returns with the best calculation.

The problem with the hash table could be that we just have a bad hash table that has trouble filling in information. In cases like this we would extend the array to unrealistic numbers that would never be filled. Another problem with the hash table is that it would have so many collisions that many items will take a long time finding a spot in a linear setup.