Hash Table Collisions

Hash table collisions occur when the index calculated by the hash function already has data residing.

I suggest two ways to handle it:

1) Linear Probing:

Once an index is calculated and it's found to be populated, we then start searching for the very next available index that is available. For example, if the index calculated is 6, and it's found to be occupied, we then start checking at index 7 and even that is occupied we evaluate index 8 and if it's found to be free, we add it to index 8 and then add some information to acknowledge that. Locality of reference can make this operation efficient, but it can get a bit cumbersome to implement.

2) Chaining:

This method uses a linked list at each index location, every index location stores a pointer to the head of a linked list. Now to insert data to a particular index calculated by the hash function we'd simply add the data to the end of the linked list. Now let's say a collision has occurred we'd not have to worry as we'd simply add it to the linked list, hence multiple data points can exist at the same index. Now, when we want to search for something, we simply index into that position and traverse the linked list to find that data. This is comparatively more efficient than linear search.