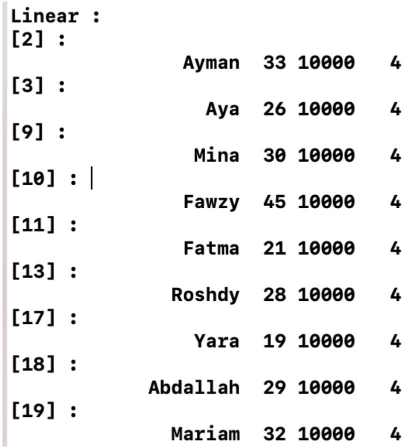
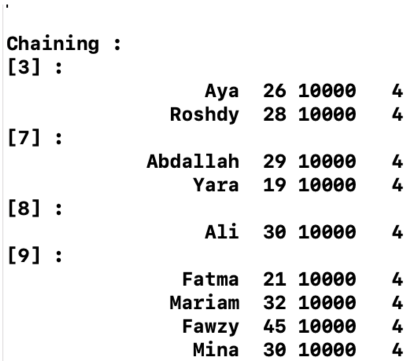


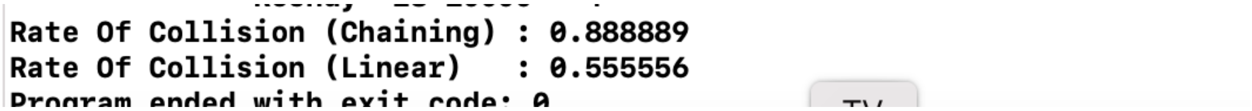
```
Linear :
[2] :
             Ayman  33 10000   4
[3] :
               Aya  26 10000   4
[9] :
              Mina  30 10000   4
[10] :
             Fawzy  45 10000   4
[11] :
             Fatma  21 10000   4
[13] :
            Roshdy  28 10000   4
[17] :
              Yara  19 10000   4
[18] :
          Abdallah  29 10000   4
[19] :
            Mariam  32 10000   4
```

This is the output of the first hash function, and I used it in linear probing, and this function is return key[0]-65, and this function changes the first letter of the name to index, so it can be added to the array.



```
Chaining :
[3] :
               Aya  26 10000   4
            Roshdy  28 10000   4
[7] :
          Abdallah  29 10000   4
              Yara  19 10000   4
[8] :
               Ali  30 10000   4
[9] :
             Fatma  21 10000   4
            Mariam  32 10000   4
             Fawzy  45 10000   4
              Mina  30 10000   4
```

In chaining I used different hash functions, and this hash function used to convert all the letters of the name to numbers and sum it (return sum%size) and then convert the sum to the index that the node has to be put in.



```
Rate Of Collision (Chaining) : 0.888889
Rate Of Collision (Linear)   : 0.555556
Program ended with exit code: 0
```

According to the numbers, we will find that the collision rate of linear is less than the chaining.

- For me I think the first hash function is much better than the second one because it is more faster, and it has less collision rate and this can make it good hash function
- I chose the hash function in chaining because this function adds the ASCII values of each letter in a string. This hash function should do an excellent job of distributing strings uniformly across the hash table slots if the size of the hash table M is small in comparison to the summations that result, as it gives equal weight to all characters in the string. This is an illustration of the folding method of hash function design. Note that the outcome is unaffected by the order of the characters in the string.

- I chose another hash function in linear probing that uses the first letter only of the name and converts it to index, after that it puts it in the returned index.