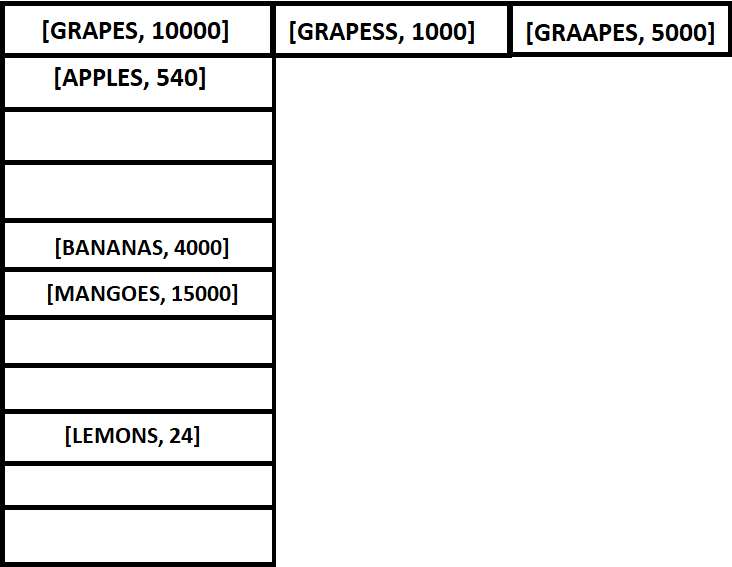
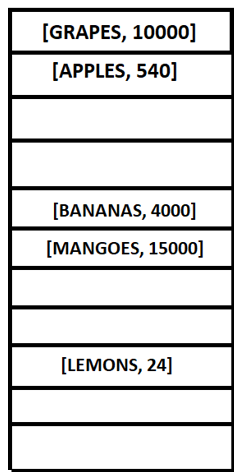
**HASH TABLE IMPLEMENTATION**

**How it works?**

****

**How to implement**

**LIST/ARRAY= [[1], [4], [5], [2] , [3], ]**

****

HASH FUNCTION

**INPUT= [GRAPES, 10000]**

**PROCESS/\_HASH**

**ALOGRITHM:**

**HASH FUNCTION (EXAMPLE):**

**\_Hash ():  
 take to full key variable.**

**Iterate through each variable.**

**Initialize hash =0.**

**Add has to the starting and then add UTF-8/UTF-16 character code to it % the length of the hash table which will already be specified by now.**

**Return hash code.**

**HASH TABLE**

* Take and initial array of length specified
* The has table will have 2 functions
  + Set() – place the elements at the hashed address
  + Get() - will get the has address of the key and lookup for the element at the given address
* **Constructor(length) :**
  + Initialize the array with the length
  + Initialize the array element as blank array/list[]
* **Set( key, value) :**
  + Get the hash code consider it as the index of the array at which the key and value is to be stored.
  + If there is already a **[key, value]** pair at that index :
    - **Array[index][length]** = the list of key and value at the given index
  + else :
    - insert the list of key and value at the given index
* **Get( key ) :**
  + Get the hash code of the address/ index at which the value is residing
  + Iterate through each key of residing at the address
  + if the key matches the desired key :
    - Return the value of the particular key
  + else :
    - return undefined/None/error message