LAB 4

Implementing Radix Sort

Problem1. Take non negative integer numbers from the file input or from user input and sort those numbers using Radix sort. The program should also show the each pass of the sorting. You can also use a different data structure such as linked list for sorting.

Algorithm

1. In the first iteration the elements are picked up and kept in various pockets checking their unit's digit.

2. The numbers are collected from pocket 9 to 0 and again they are given as input to the sorter.

3. In the second iteration, the ten's digit is sorted.

4. Repeat through step 2.

5. In the final iteration the digit in hundred's position is sorted.

6. Repeat through step 2.

```
Radix(A,n){

 for(j=0; j < k; j++) {
  for(i=0; i < n; i++) {
    determine the jth digit of A[i] ((A[i]/(10^j))%10), call it d
    append A[i] onto pocket d
  }
  for(d=0; d < 10; d++)
    place pocket d in A (after pocket d-1, before pocket d+1)
}
```

Check http://www.slideshare.net/gkumar007/radix-sort for detail example.

Implementing Hash table using division method

Problem 2 . Create a Simple hash table using division method for a list of string names of your choice and print the hash table with the hash index and names. Use linked list for your program.

In the division method for creating hash functions, we map a key k into one of m slots by taking the remainder of k divided by m.

A key is mapped into one of m slots using the function

$$h(k) = k \bmod m$$

For more information about hash function check
http://homepages.ius.edu/rwisman/C455/html/notes/Chapter11/MulMethod.htm

```
 while(1){

             // Add data from file or user input
             index (Stringnames)
             print( index)
             //create linked list
             link = (node*)malloc(sizeof(node));
             // Copy string names to linked list
            }
// Print linked list with hash values and names

 index( Char * data){
     size = strlen(data);
              while (i <size){
               //convert each char to int and add them up to get the K
                           }
              return k mod m
     }
```