Term work : 05

Consider a supermarket scenario where sales manager wants to search for the customer details usig a customer id. customer information like (custid, custname and custphno) are stored as a structure and custid will be used as hash key. Develope and execute a program in C using suitable data structures to implement the following operations:

a. Insertion of a new data entry.

b. search for customer information using custid.

c. display the records (Demonstrate collision and its handling using linear probing methods)

Aim:

To learn the implementation of hashing is solving problems

Theory:

Hashing is an important data structure which is designed to use a special function called the Hash function which is used to map a given value with a particular key for faster access of elements. The efficiency of mapping depends of the efficiency of the hash function used.

Let a hash function H(x) maps the value x at the index x% 10 in an array eg. if the list of values is [11, 12, 13, 14, 15] it will be stored at positions {1, 2, 3, 4, 5} in the array or Hash Table respectively.

count = 0 ;

Source code :

```
# include <stdio.h>
# include <stdlib.h>

# define SIZE 10

struct customer {
        int custid;
        char custname [30];
        double custphno;
    };
struct Record {
        struct customer info;
        int empty;
    };
int Hashfn (int key){
        return (key % SIZE);
    }
int search ( int key, struct Record ht []){
        int count, temp, pos;
        temp = Hashfn(key);
        pos = temp;
        for ( count = 1; count <= size ; count ++){
            if ( ht[pos] .empty == 1)}
            return -1;
            }
        if ( ht [pos] .info ) . custid == key) {
```

```
        return pos;
        }
    pos = ( temp + count) % SIZE;
    }
    return -1;
}
void InSHT LP( struct customer cust, struct Record ht [ ]){
        int count, post, temp;
        int key = cust custid;
        temp = Hashfn (key);
        pos = temp;
        for( count = 1; count <= SIZE; count ++){
            if (ht [post . empty = =1){
                ht [pos]. info = cust;
                ht [pos]. empty = -1;
                printf (" In Record Inserted into Hash Table \n");
                return;
            }
        if ((ht [pos]. info). custid == key){
          printf ("\n Duplicate Record cannot be Inserted \n");
        }
          pos= ( temp t count) % SIZE;
        }
      printf ("\n Hash Table is Full \n");
    }
void Display (struct Record ht [ ]){
int count;
printf ("\n Hash Table");
  for ( count = 0; count < SIZE; count ++){
      printf (" \n [%.d] : \t", count);
```

```
if (ht [count]. empty ==-1){
printf ("\n Customer. ID:%d \t Name : %s\t Phone : %lf ",
  [ht[count].info ). custid, ht [count].info ). custname,
  (ht [count]. info . custphno);
  }
  else
    printf ("\n No Hash Entry \n");
   }
  }
}
int main(){
  int count, key, option ;
  struct Record ht[SIZE];
  struct customer cust ;
  for ( count = 0; count < SIZE ; count ++){
      ht [count]. empty = 1;
   }
   while (1){
    printf ("\n 1. Insert a Record \n");
    printf ("\n 2. Search a Record \n");
    printf (" \n 3. Display all Records \n");
    printf ("\n 4. Exit \n");
    printf (" Enter your option :");
    scanf ("%d", &option);
    switch (option) {
     case1 : printf ("\n Enter customer id, name, ph :");
    scanf ( " %d %s %f ", &cust. custid, &cust. custname,
    &cust. custphno );
```

```
INSHT- LP (cnt, ht);
break;
case 2: printf ("\n Enter the key to search : ");
    scanf ("%d", skey);
    count = search ( key. ht);
    if (count == -1)~~}~~{
        printf (" \n Record Not Found \n");
     }
      else {
         printf ("\n Record Found at Index : %d\n", count);
      }
     break;
 case 3 : Display (ht);
             break;
 case4 : exit (1);
  }
}
 return 0;
}
```

REFERENCES:

Books

1. Richard F Lilberg , Behrouz A Forouzan , Data structures : A Pseudo code Approach with C, Lingage 2007.
2. Horowitz , Sahni , Anderson - Fread , Fundamentals and Date structures in C, Universe Press 2nd Edition .

E - Resources :

1. https : // geeksforgeeksorg /

CONCLUSION.

In this term work , we learn about hashing , basic operations of hashing and their implementation to solve problems. we also learnt basic problem solving techniques and programming paradesigns .

TERM WORK : 06

consider a warehouse where the items have to be arranged in an ascending order. Development and execute a program in C using suitable data structures to implement warehouse such that items can be traced easing.

AIM:

To learn the implementation of linked list in solving problems.

THEORY

A linked list is a sequence of data structures which are connected together via links. Linked list is a sequence of links which contains items. Each link contains a connection to another link. It is the second most used data structure after array.

Basic operations of linked list.

Insertion - Adds an elements at the beginning of the list.

Deletion - Deletes an element at the beginning of the list.

Display - Displays the complete list.

Delete - Deletes an element using the given key.

SOURCE CODE:

```
#include <stdio.h>
struct node
 {
   int data ;
   struct node * next ;
  };
 void Diplay (struct node * head)
 {
   struct node * temp ;
   temp = head ;
   while ( temp != NULL)
  {
   printf ( " %d ", temp -> data);
   temp = temp -> next ;
  }
}
struct node * Add (struct node * head, int value)
 {
   struct node * newnode , * prev * curr ;
   newnode = (struct node * lmalloc ( size of ( struct node)) ;
   newnode -> date = value ;
   newnode -> next => NULL ;
   if ( newnode == NULL)
   {
      printf (" Error : Could not allocate memory : ") ;
   }
   else
    {
     if (head == NULL)
head = new node ;
else
{
   if (newnode -> data   < head -> data)
   {
    newnode -> next = head;
     head = newnode;
   }
  else
   {
     curr  = head -> next ;
      prev = head ;
    while (curr != NULL && newnode -> data -> curr -> data)
     {
        prev = prev -> next ;
        curr = curr -> next;
     }
     prev -> next = newnode ;
     newnode -> next = curr ;
     }
   }
  }
   return head ;
  }
  void main ()
  {
```

```
int choice, value;
struct node * head =NULL;

for (;;)
{
  printf ("\n Enter 1. Add 2.Display 3.Exit ");
  printf ("\n Enter choice :");
  scanf ("%d", &choice);
  switch (choice)
 }
  case 1: printf ("\n Enter value ");
  ~~Ea~~          scanf (" %d", &value);
           head= Add (head. value);
           break;
   case 2: if (head = = NULL)
           {
           printf (" List is empty");
           }
           Display (head;
           break;
   Case 3: exit(1);
           break;

     }

   }
}
```

REFERENCE :

Books :

1. Richard F Wilberg , Behrouz A Fourozan , Data structures . A Psuedo code Approach with c, gngage 2007.
2. Horowitz , Sahni , Andarson - Fread Fundamentals of Data structures inc , Universe Press 2nd Edition .

E-Resources.

1- http://geeksforgeets.org/

CONCLUSION:

In this term work , we learnt about linked list basic operations of linked list and their implementations to solve problems. We also learnt basic problem solving techniques and programming paradigms .