**WEEK – 9**

**NAME : SANMAT SANJAYAKUMAR PAYAGOUDAR**

**SRN : PES1UG20CS385**

**SECTION : G**

/\*   
1)Write a program to insert a mobile number(10 digit) along with the name of   
customer into a hash table. Add functions to delete based on phone number   
Add function to search a particular phone number and display ythe details   
accordingly.Handle the collision using seperate chainingad linear probing   
techniques   
\*/

**PROGRAM -1 :**

**HASHING**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

struct node

{

char name[20];

int ph\_no;

struct node \*next;

};

struct hash

{

struct node \*head;

int count;

};

void insert(struct hash \*ht, int size, int ph\_no, char \*name);

void delete(struct hash \*ht, int size, int ph\_no);

void search(struct hash \*ht, int sz, int ph);

void display(struct hash \*ht, int size);

void main()

{

int sz, ch, num, i;

char name[20];

struct hash \*ht;

printf("Enter the size of table : ");

scanf("%d", &sz);

ht=(struct hash \*)malloc(sz\*(sizeof(struct hash)));

for(i=0;i<sz;i++)

{

ht[i].head=NULL;

ht[i].count=0;

}

while(1)

{

printf("\n1:Insert a PHONE NUMBER\n2:Delete a PHONE NUMBER\n3:Search a PHONE NUMBER\n4:Display\n");

printf("Enter your choice : ");

scanf("%d", &ch);

switch(ch)

{

case 1: printf("Enter the PHONE NUMBER : ");

scanf("%d", &num);

printf("\nEnter the name : ");

scanf("%s", name);

insert(ht, sz, num, name);

break;

case 2: printf("Enter the PHONE NUMBER to be deleted : ");

scanf("%d", &num);

delete(ht, sz, num);

break;

case 3: printf("Enter the PHONE NUMBER to be searched : ");

scanf("%d", &num);

search(ht, sz, num);

break;

case 4: display(ht, sz);

break;

}

}

}

void insert(struct hash \*ht, int size, int ph\_no, char \*name)

{

int index;

struct node \*temp;

temp=(struct node \*)malloc(sizeof(struct node));

temp->next=NULL;

temp->ph\_no=ph\_no;

strcpy(temp->name, name);

index=ph\_no%size;

temp->next=ht[index].head;

ht[index].head=temp;

ht[index].count++;

}

void delete(struct hash \*ht, int size, int ph\_no)

{

struct node \*prev, \*temp;

int index;

index=ph\_no%size;

prev=NULL;

temp=ht[index].head;

while(temp->ph\_no!=ph\_no)

{

prev=temp;

temp=temp->next;

}

if(temp==NULL)

printf("Element not found\n");

else

{

if(prev==NULL)

{

ht[index].head=temp->next;

}

else

{

prev->next=temp->next;

}

}

}

void display(struct hash \*ht, int size)

{

int i;

struct node \*temp;

printf("\n");

for(i=0;i<size;i++)

{

printf("%d(%d): ", i, ht[i].count);

if(ht[i].head!=NULL)

{

temp=ht[i].head;

while(temp!=NULL)

{

printf("PHONE NUMBER - %d, NAME - %s -> ", temp->ph\_no, temp->name);

temp=temp->next;

}

}

printf("\n");

}

}

void search(struct hash \*ht, int sz, int ph)

{

int index;

struct node \*temp;

index=ph%sz;

temp=ht[index].head;

while((temp!=NULL)&&(temp->ph\_no!=ph))

temp=temp->next;

if(temp!=NULL)

{

printf("\nFound with\n");

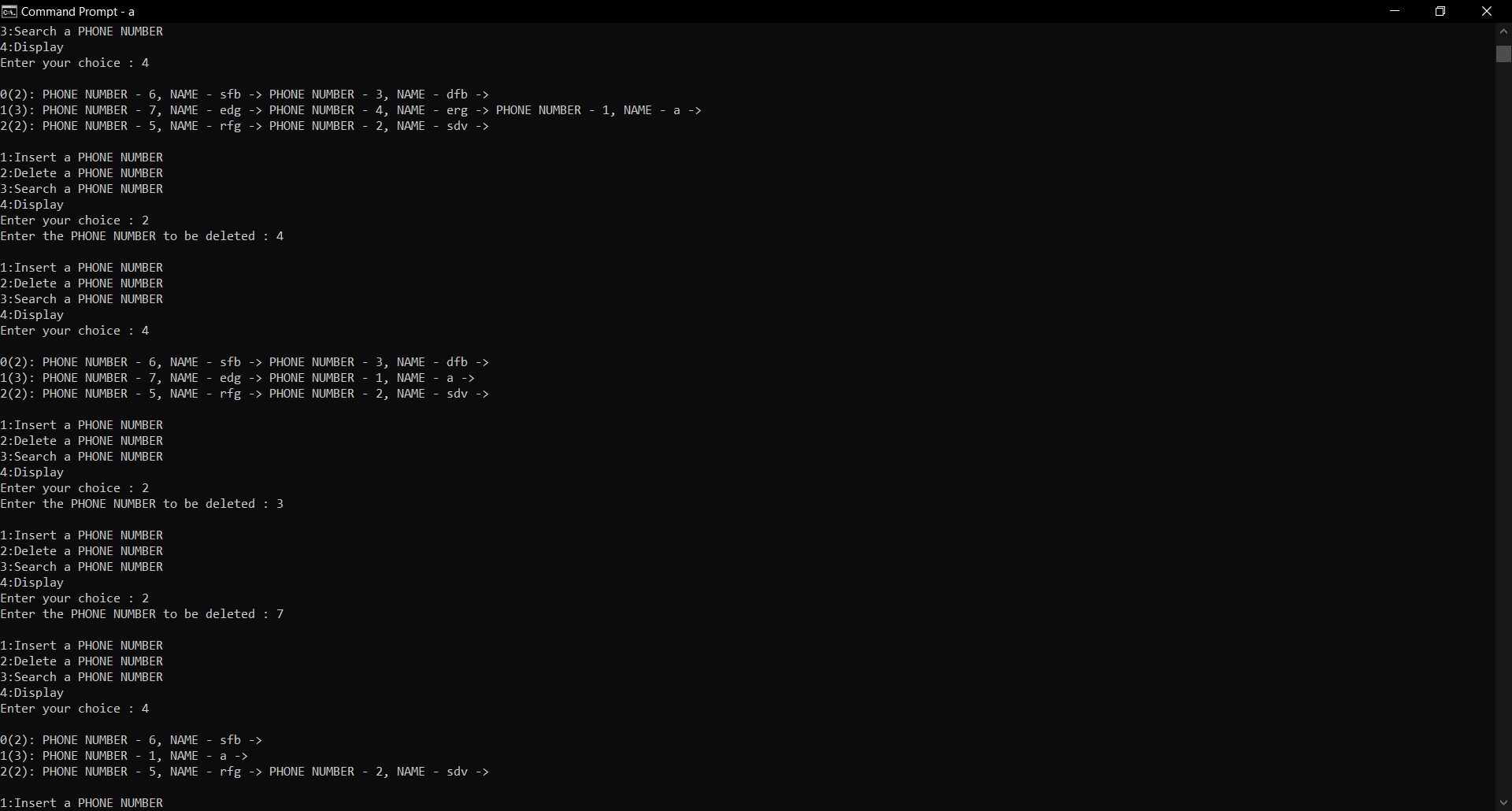
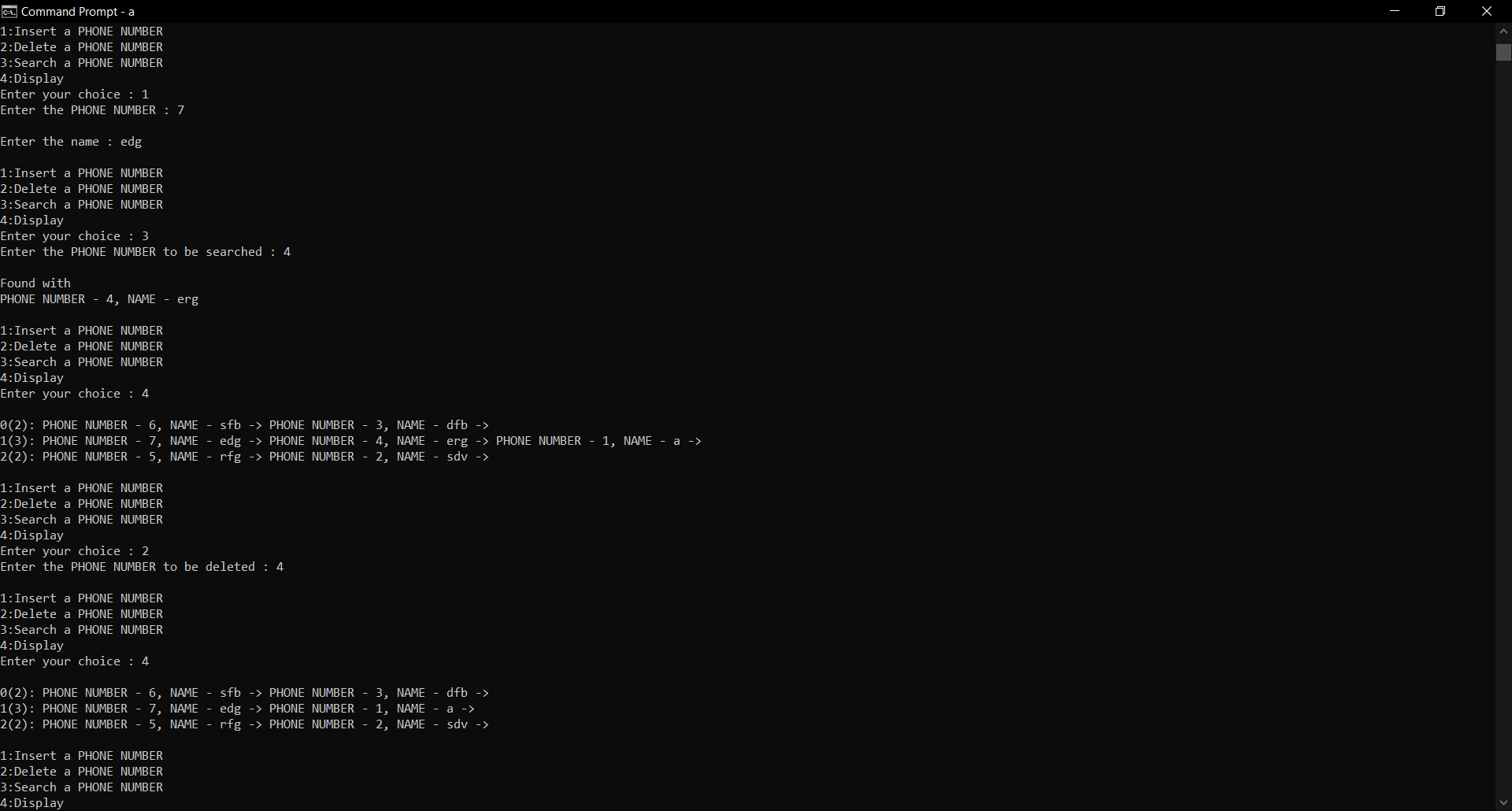
printf("PHONE NUMBER - %d, NAME - %s\n",temp->ph\_no, temp->name);

}

else

printf("Record not found\n");

}



**PROGRAM-2 :**

**LINEAR PROBING**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

struct element

{

int ph\_no;

char name[10];

int mark;

};

void insert(struct element \*ht, int size, int ph\_no, char \*name, int \*count);

int delete(struct element \*ht, int size, int ph\_no, int \*count);

void display(struct element \*ht, int size, int \*count);

void search(struct element \*ht, int size, int ph\_no);

int main()

{

struct element \*ht;

int ch, sz, i, num, no\_elements;

char name[10];

printf("Enter the table size : ");

scanf("%d", &sz);

ht=(struct element \*)malloc(sz\*(sizeof(struct element)));

for(i=0;i<sz;i++)

{

ht[i].mark=0;

}

no\_elements=0;

while(1)

{

printf("\n1:Insert a PHONE NUMBER\n2:Delete a PHONE NUMBER\n3:Search a PHONE NUMBER\n4:Display\n");

printf("Enter your choice : ");

scanf("%d", &ch);

switch(ch)

{

case 1: printf("Enter the PHONE NUMBER : ");

scanf("%d", &num);

printf("\nEnter the name : ");

scanf("%s", name);

insert(ht, sz, num, name, &no\_elements);

break;

case 2: printf("Enter the PHONE NUMBER to be deleted : ");

scanf("%d", &num);

delete(ht, sz, num, &no\_elements);

break;

case 3: printf("Enter the PHONE NUMBER to be searched : ");

scanf("%d", &num);

search(ht, sz, num);

break;

case 4: display(ht, sz, &no\_elements);

break;

}

}

}

void insert(struct element \*ht, int size, int ph\_no, char \*name, int \*count)

{

int index;

if(\*count==size)

{

printf("Table is full\n");

}

index=ph\_no%size;

while(ht[index].mark==1)

{

index=(index+1)%size;

}

ht[index].ph\_no=ph\_no;

strcpy(ht[index].name, name);

ht[index].mark=1;

(\*count)++;

}

int delete(struct element \*ht, int size, int ph\_no, int \*count)

{

int index, i=0;

if(\*count==0)

printf("Table is empty\n");

index=ph\_no%size;

while(ht[index].ph\_no!=ph\_no && i<=\*count)

{

index=(index+1)%size;

i++;

}

if(ht[index].ph\_no==ph\_no)

{

ht[index].mark=0;

(\*count)--;

}

else

{

printf("Not found in the table\n");

}

}

void display(struct element \*ht, int size, int \*count)

{

int i, index;

for(i=0;i<size;i++)

{

printf("\n%d : ", i);

if(ht[i].mark!=0)

printf("PHONE NUMBER - %d, NAME - %s, MARK - %d -> ", ht[i].ph\_no, ht[i].name, ht[i].mark);

}

printf("\nNumber of elements : %d", \*count);

}

void search(struct element \*ht, int size, int ph\_no)

{

int index;

index=ph\_no%size;

while(ht[index].ph\_no!=ph\_no)

index=(index+1)%size;

if(ht[index].ph\_no==ph\_no)

{

printf("\nFound with\n");

printf("PHONE NUMBER - %d, NAME - %s\n", ht[index].ph\_no, ht[index].name);

}

else

printf("Record not found\n");

}

