EXP 8 **HASHING** **Date:**

Aim: Implementation of the following hash functions.

1. **Division method**
2. **Trunication method**
3. **Midsquare method**
4. **Folding method**

Code:

```
#include <stdio.h>
#include<math.h>
void display(int m,int *hash)
{
   printf("\n                    HASH TABLE\n");
   for(int i=0;i<m;i++)
   {
      if(hash[i]==0)
      {
         printf("h[%d]=\n",i);
      }
      else{

      printf("h[%d]=%d\n",i,hash[i]);
      }
   }
}
void division(int *keys,int n)
{
   int hash[10]={0};
   int k;
   int m=8;
   for(int i=0;i<n;i++)
   {
      hash[keys[i]%m]=keys[i];
   }
   display(m,&hash[0]);

}

void trunication(int *keys,int n)
{
   int hash[10]={0};
   for(int i=0;i<n;i++)
   {
      hash[keys[i]/10]=keys[i];
   }
   display(8,&hash[0]);
}

void folding(int *keys,int n)
{
   int sum=0,temp,hash[10]={0};
   for(int i=0;i<n;i++)
   {
      temp=keys[i];
      while(temp!=0)
      {
         sum+=temp%10;
```

```
            temp=temp/10;
        }
        hash[sum]=keys[i];
        sum=0;
    }
    display(10,&hash[0]);
}

void midsquare(int *keys,int n)
{
    int hash[10]={0};
    int no_of_digits,e,temp,a;
    for(int i=0;i<n;i++)
    {
        no_of_digits=(log10(keys[i])+1); // no_of-digits stores the value of the
number of digits of the number
        a=round(no_of_digits/3); // we divide no of digits by 3 inorder to make 3
sections of the number
        e=no_of_digits - 2*a;
        temp=keys[i];
        temp=temp/pow(10,a);
        temp=pow(temp % (int)pow(10,e),2);
        hash[temp]=keys[i];
    }
    display(10,&hash[0]);
}

int main()
{
    int n,o,con;
    int keys[100];

    do{
        printf("\nEnter the no of keys: ");
    scanf("%d",&n);

    printf("\nEnter the keys: ");
    for(int i=0;i<n;i++)
    {
        scanf("%d",&keys[i]);
    }
        printf("\nEnter 1 for division method\nEnter 2 for trunication\nEnter 3 for
Midsquare method\nEnter 4 for folding method\nEnter your choice: ");
        scanf("%d",&o);
        switch (o)
        {
        case 1:
            division(&keys[0],n);
            break;
        case 2:
            trunication(&keys[0],n);
            break;
        case 3:
            midsquare(&keys[0],n);
            break;
        case 4:
            folding(&keys[0],n);
            break;
        default:
```

```
            printf("\nWrong choice");
            break;
        }
        printf("\nDo you want to continue(1/0): ");
        scanf("%d",&con);
    }
    while(con==1);
    return 0;
}
```

Output:

```
Enter the no of keys: 5

Enter the keys: 72 18 43 36 6

Enter 1 for division method
Enter 2 for trunication
Enter 3 for Midsquare method
Enter 4 for folding method
Enter your choice: 1

                HASH TABLE
h[0]=72
h[1]=
h[2]=18
h[3]=43
h[4]=36
h[5]=
h[6]=6
h[7]=

Do you want to continue(1/0): 1

Enter the no of keys: 5

Enter the keys: 72 18 43 36 6

Enter 1 for division method
Enter 2 for trunication
Enter 3 for Midsquare method
Enter 4 for folding method
Enter your choice: 2

                HASH TABLE
h[0]=6
h[1]=18
h[2]=
h[3]=36
h[4]=43
h[5]=
h[6]=
h[7]=72

Do you want to continue(1/0): 1

Enter the no of keys: 3

Enter the keys: 123 312 739
```

```
Enter 1 for division method
Enter 2 for trunication
Enter 3 for Midsquare method
Enter 4 for folding method
Enter your choice: 3

                HASH TABLE
h[0]=
h[1]=312
h[2]=
h[3]=
h[4]=123
h[5]=
h[6]=
h[7]=
h[8]=
h[9]=739

Do you want to continue(1/0): 1

Enter the no of keys: 5

Enter the keys: 1 11 23 61 44

Enter 1 for division method
Enter 2 for trunication
Enter 3 for Midsquare method
Enter 4 for folding method
Enter your choice: 4

                HASH TABLE
h[0]=
h[1]=1
h[2]=11
h[3]=
h[4]=
h[5]=23
h[6]=
h[7]=61
h[8]=44
h[9]=

Do you want to continue(1/0): 0
```