Experiment No:2.6

Digit extraction hashing :

```
#include <iostream>
#define max 100
using namespace std;
class digitext
{
public:
        int d,ud[10],n,c,a[max];
        digitext()
        {
                cout<<"Enter the number of memory locations";
                cin>>n;
                for(int i=0;i<n;i++)
                {
                        a[i]=0;
                }
                c=0;
                cout<<"\nEnter the number of digits to be extracted";
                cin>>d;
                cout<<"\nEnter the digit positions";
                for(int i=0;i<d;i++)
                {
                        cin>>ud[i];
                }
        }
        void menu()
        {
                int s;
                do
                {
                        cout<<"\nEnter your choice\n1. Insertion\n2. Display\n3. Exit\n4. Search\n";
                        cin>>s;
                        switch(s)
                        {
                                case 1:
                                        insert();
                                        break;
                                case 2:
                                        display();
                                        break;
                                case 3:
                                        break;
                                case 4:
                                        search();
                                        break;
                        }
```

```
        }while(s!=3);
}
void insert()
{
        int t,un,x,x1,x2=0,j=0;
        cout<<"\nEnter the number: ";
        cin>>un;
        if(c!=n)
        {
                for(int k=0;k<d;k++)
                {
                        x=un;
                        for(int i=1;i<=ud[k];i++)
                        {
                                x1=x%10;
                                x=x/10;
                        }
                        x2=x2*10+x1;
                }
                while(x2>0)
                {
                        x1=x2%10;
                        j=j*10+x1;
                        x2=x2/10;
                }
                t=j%n;
                while(a[t]!=0)
                {
                        t=t+1;
                        if(t==n)
                        {
                                t=0;
                        }
                }
                a[t]=un;
                c++;
        }
        else
        {
                cout<<"\nMemory is full";
        }
}
void display()
{
        if(c!=0)
        {
                cout<<"\n";
                for(int j=0;j<n;j++)
                {
                        cout<<a[j]<<endl;
```

```
                }
        }
        else
        {
                cout<<"\nMemory is empty\n";
        }
}
void search()
{
        int t,un,x,x1,x2=0,j=0;
        cout<<"\nEnter the number to search: ";
        cin>>un;
        if(c!=0)
        {
                for(int k=0;k<d;k++)
                {
                        x=un;
                        for(int i=1;i<=ud[k];i++)
                        {
                                x1=x%10;
                                x=x/10;
                        }
                        x2=x2*10+x1;
                }
                while(x2>0)
                {
                        x1=x2%10;
                        j=j*10+x1;
                        x2=x2/10;
                }
                t=j%n;
                if(a[t]==un)
                {
                        cout<<"\nFound at "<<t+1<<"th position\n";
                }
                else
                {
                        j=t;
                        t++;
                        while(a[t]!=un && t!=j)
                        {
                                t=t+1;
                                if(t==n)
                                {
                                        t=0;
                                }
                        }
                        if(a[t]==un)
                        {
                                cout<<"\nFound at "<<t+1<<"th position\n";
```

```
                                        }
                                        else
                                        {
                                                cout<<"\nElement not found\n";
                                        }
                                }
                        }
                        else
                        {
                                cout<<"\nMemory is empty\n";
                        }
                }
};
int main()
{
        digitext o;
        o.menu();
        return 0;
}
```

OUTPUT:

Enter number of locations: 10

Enter the number of digits to be extracted: 3

Enter the positions of digits to be extracted: 1 3 4

Digit Extraction Menu: 1. Insert an element. 2. Display. 3. Search an element. 4. Exit. Enter your choice: 1

Enter an element: 12345

Number added!... Digit Extraction Menu: 1. Insert an element. 2. Display. 3. Search an element. 4. Exit. Enter your choice: 1

Enter an element: 23456

Number added!... Digit Extraction Menu: 1. Insert an element. 2. Display. 3. Search an element. 4. Exit. Enter your choice: 1

Enter an element: 34567

Number added!... Digit Extraction Menu:

1. Insert an element. 2. Display. 3. Search an element. 4. Exit. Enter your choice: 1

Enter an element: 45678

Number added!... Digit Extraction Menu: 1. Insert an element. 2. Display. 3. Search an element. 4. Exit. Enter your choice: 2

Location Element 0 1 2 3 4 5 12345 6 23456 7 34567 8 45678 9

Digit Extraction Menu: 1. Insert an element. 2. Display. 3. Search an element. 4. Exit.