hashing\_07.cpp

**Compile:** g++ hashing\_07.cpp -o hashing\_07

**Run:** ./hashing\_07

**Program:**

1. **Modulo Division Technique**

#include<iostream>

#include<stdlib.h>

#include<math.h>

#define MAX 100

using namespace std;

intch,c,n,arr[MAX],a1[MAX],key,num;

class hash

{

public:

voidgetdata()

{

cout<<"\nEnter the number of elemets you want to insert :).. ";

cin>>num;

cout<<"enter value of n"<<endl;

cin>>n;

cout<<"Enter data"<<endl;

for(inti=0;i<num;i++)

{

cin>>arr[i];

}

}

void modulo()

{

cout<<"positions are";

for(inti=0;i<num;i++)

{

key=arr[i]%n;

if(a1[key]!=0)

key=linear();

a1[key]=arr[i];

}

}

void display()

{

for(inti=0;i<n;i++)

{

cout<<"elements after allocation is : "<<i<<" "<<"value:\t"<<a1[i]<<"\n"<<endl;

}

}

int linear()

{

for(inti=key;i<n;i++)

{

if(a1[key]!=0)

{

key=key+1;

if(key==n)

key=0;

}

}

return key;

}

};

int main()

{

hash h;

h.getdata();

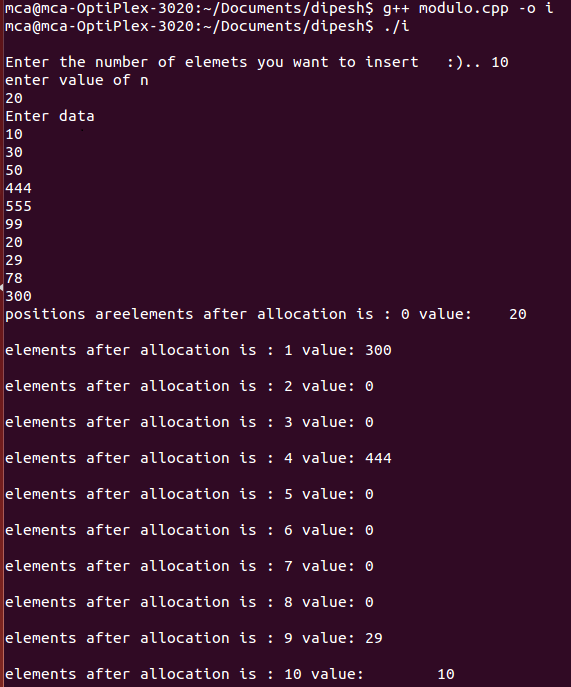
h.modulo();

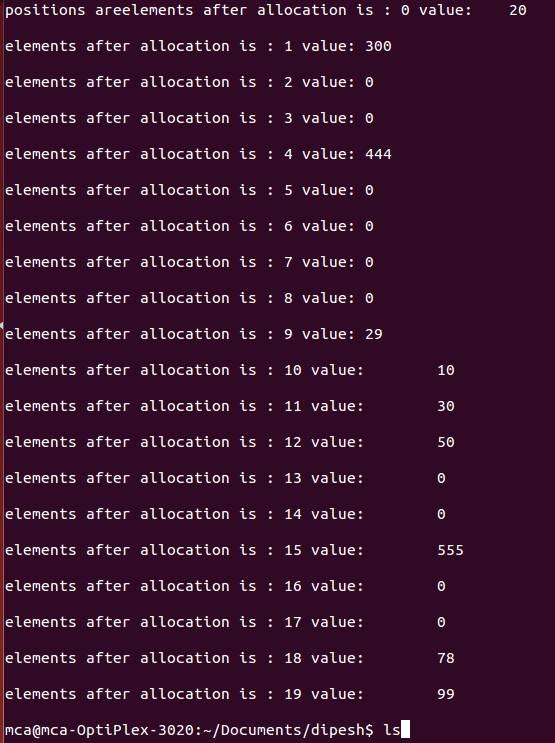
h.display();

return 0;

}

**Output:**

****

****

1. **Digit Extraction Technique**

#include<iostream>

using namespace std;

long arr[11];

long check(long no)

{

if(arr[no]==0)

return 0;

else

return 1;

}

void hash()

{

long temp,no,pos;

for(int i=1;i<=10;i++)

{arr[i]=0; }

cout<<"\n\nEnter 5 numbers of 6 digit : ";

for( int i=1;i<=5;i++)

{

cout<<"\n\n\n\nEnter number "<<i<<" : ";

cin>>no;

temp=(no/100000)\*100;

temp=temp+ ((no/1000)%10)\*10;

temp=temp+(no%100)/10;

cout<<"\n\n Digits extracted from "<<no<<" are : "<<temp;

if(temp%10==0)

{pos=temp%10+1;}

else

{pos=temp%10;}

while(1)

{

if(check(pos)==0)

{

arr[pos]=no;

break;

}

else

++pos;

if(pos>10)

pos=pos%10;

}

}

for(int i=1;i<11;i++)

cout<<"\n\na["<<i<<"] = "<<arr[i];

}

int main()

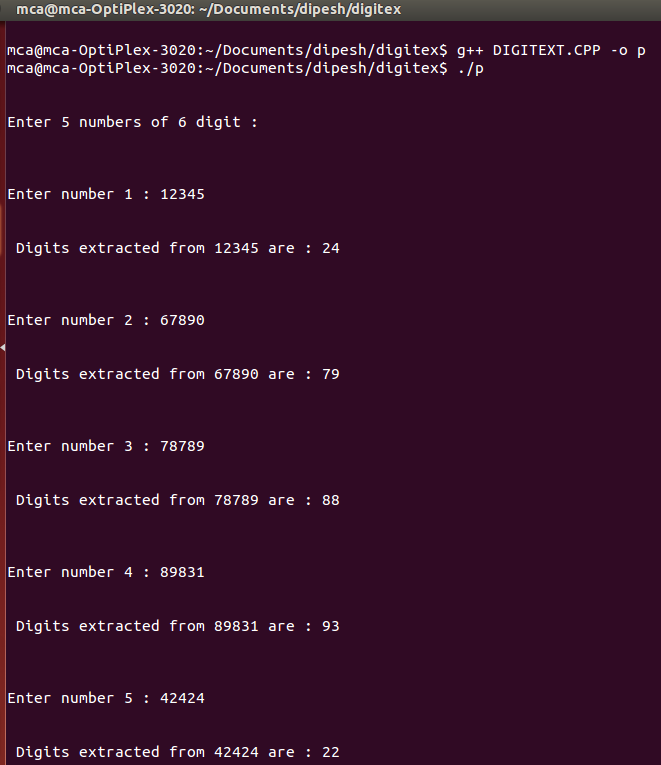
{

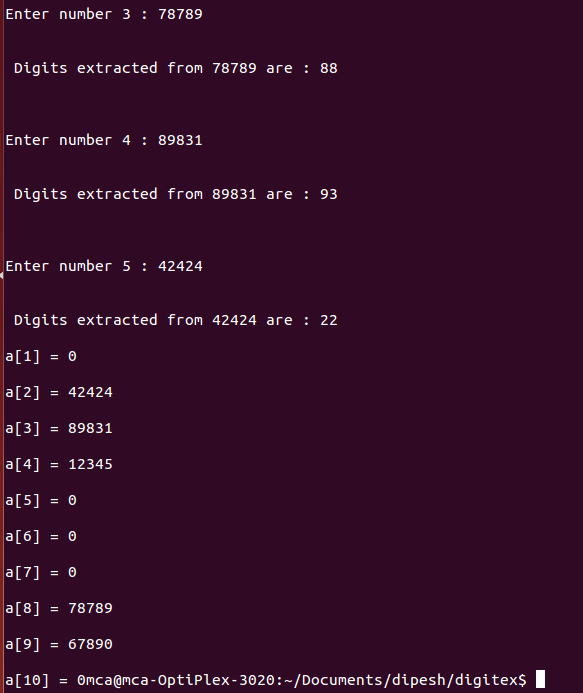
hash();

return 0;

}

**Output:**

****

****

1. **Fold Shift Technique**

#include<iostream>

using namespace std;

longarr[101];

long check(long no)

{

if(arr[no]==0)

return 0;

else

return 1;

}

int main()

{

longtemp,pos,a,b,c;

for(inti=0;i<101;i++)

arr[i]=0;

for(inti=1;i<=5;i++)

{

cout<<"\n\n Enter 6 digit number "<<i<<" ";

cin>>temp;

a=temp/10000;

cout<<a<<" ";

b=(temp/100)-a\*100;

cout<<b<<" ";

c=temp%100;

cout<<c<<" ";

pos=(a+b+c)%100;

while(1)

{

if(check(pos)==0)

{

arr[pos]=temp;

break;

}

else

++pos;

if(pos>99)

pos=pos%100;

}

}

for(inti=1;i<100;i++)

{

if(arr[i]!=0)

cout<<"\n"<<"a["<<i<<"] : "<<arr[i];

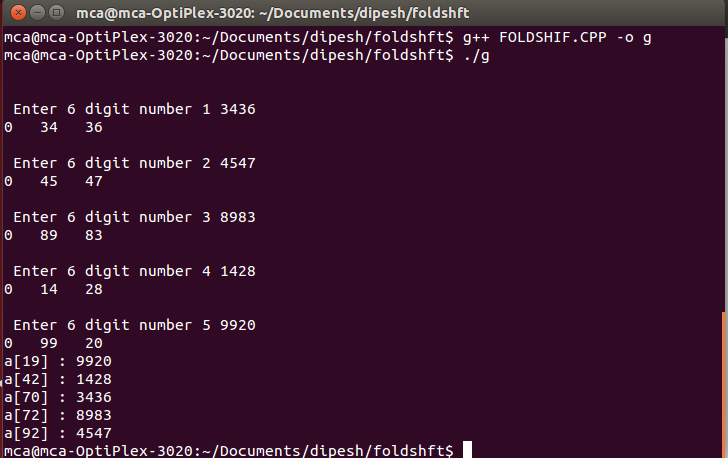
}

cout<<endl;

return 0;

}

**Output:**



1. **Fold Boundary Technique**

#include<iostream>

using namespace std;

longarr[101];

int n;

long check(long no)

{

if(arr[no]==0)

return 0;

else

return 1;

}

long reverse(long no)

{

int count=0;

longans=0;

while(no>0)

{

ans=ans\*10+no%10;

no/=10;

count++;

}

if(count==1)

{

ans=ans\*10;

}

returnans;

}

int main()

{

longtemp,pos,a,b,c;

for(inti=0;i<101;i++)

arr[i]=0;

cout<<"Enter the number of input : "<<endl;

cin>>n;

for(inti=1;i<=n;i++)

{

cout<<"\n\n Enter 5 digit number "<<i<<" :";

cin>>temp;

a=temp/10000;

b=temp/100-a\*100;

c=temp%100;

a=reverse(a);

//b=reverse(b);

c=reverse(c);

pos=(a+b+c)%100;

while(1)

{

if(check(pos)==0)

{

arr[pos]=temp;

break;

}

else

++pos;

if(pos>99)

pos=pos%100;

}

}

for(inti=1;i<100;i++)

{

if(arr[i]!=0)

cout<<"\n"<<"a["<<i<<"] : "<<arr[i];

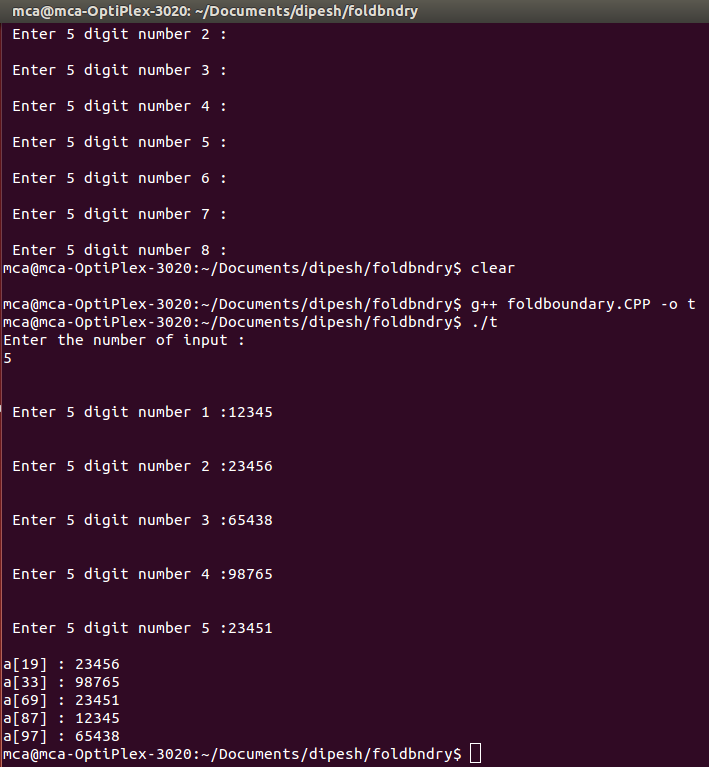
}

cout<<endl;

return 0;

}

**Output:**



1. **Pseudo-Random Technique**

#include<iostream>

using namespace std;

longarr[101];

long y[100];

inta,c,n,no,key;

int linear()

{

for(inti=key;i<no;i++)

{

if(y[key]!=0)

{

key=key+1;

if(key==no)

key=0;

}

}

return key;

}

void pseudo()

{

for(inti=0;i<no;i++)

{

key=(arr[i]\*a+c)%n;

if(y[key]!=0)

{

key=linear();

}

y[key]=arr[i];

//cout<<key<<" : "<<y[key]<<endl;

}

}

int main()

{

cout<<"enter the values of a, c and n"<<endl;

cin>>a>>c>>n;

for(inti=0;i<101;i++)

arr[i]=0;

cout<<"Enter the number of input : "<<endl;

cin>>no;

cout<<"enter numbers"<<endl;

for(inti=0;i<no;i++)

{

cin>>arr[i];

}

pseudo();

cout<<"the array locations after pseudo random are:"<<endl;

//cout<<"hiii";

for(inti=0;i<n;i++)

{

if(y[i]!=0)

cout<<"\n"<<"y["<<i<<"] : "<<y[i];

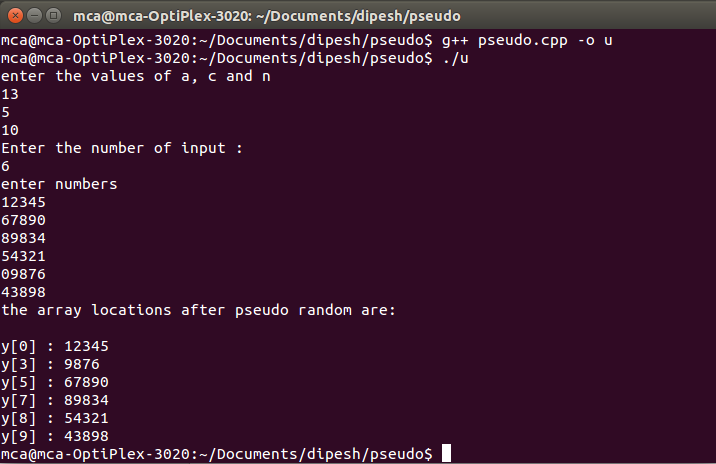
}

cout<<endl;

return 0;

}

**Output:**

****

1. **Direct Technique**

#include<iostream>

#include<cmath>

#define MAX 100

using namespace std;

intn,data[MAX],total,hash[MAX];

class hashing

{

public:

void modulo(){

int temp;

cout<<"enter no of data"<<endl;

cin>>total;

for(inti=0;i<total;i++){

cin>>data[i];

}

cout<<"enter size of array"<<endl;

cin>>n;

for(inti=0;i<total;i++){

temp=data[i];

temp=temp%n;

while(hash[temp]!=0){

temp=temp+1;

temp=temp%n;

}

hash[temp]=data[i];

}

for(inti=0;i<n;i++){

cout<<i<<"\t"<<hash[i]<<endl;

}

}

voiddhash(){

int temp;

cout<<"enter no of data"<<endl;

cin>>total;

for(inti=0;i<total;i++){

cin>>data[i];

}

cout<<"enter size of array"<<endl;

cin>>n;

for(inti=0;i<total;i++){

temp=data[i];

if (temp>n){

temp=temp%n;

while(hash[temp]!=0){

temp=temp+1;

temp=temp%n;

}

hash[temp]=data[i];

}

else{

hash[temp]=data[i];

}

}

for(inti=0;i<n;i++){

cout<<i<<"\t"<<hash[i]<<endl;

}

}

};

int main()

{

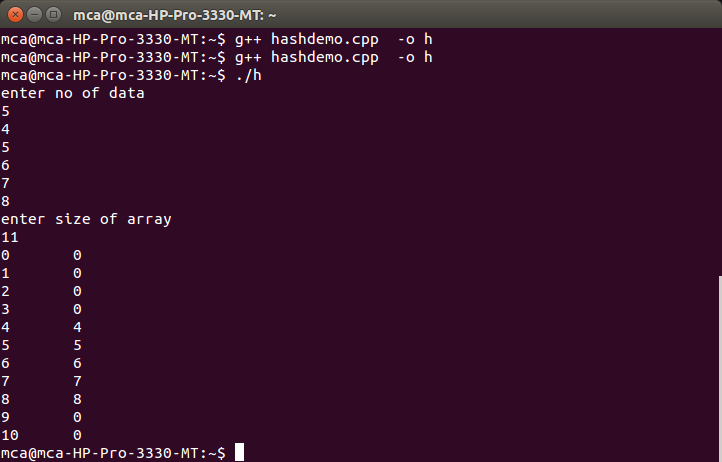
hashing h;

h.dhash();

return 0;

}

**Output:**



1. **Subtraction Technique**

#include<iostream>

#include<cmath>

#define MAX 100

using namespace std;

intn,data[MAX],total,hash[MAX];

class hashing

{

public:

void modulo(){

int temp;

cout<<"enter no of data"<<endl;

cin>>total;

for(inti=0;i<total;i++){

cin>>data[i];

}

cout<<"enter size of array"<<endl;

cin>>n;

for(inti=0;i<total;i++){

temp=data[i];

temp=temp%n;

while(hash[temp]!=0){

temp=temp+1;

temp=temp%n;

}

hash[temp]=data[i];

}

for(inti=0;i<n;i++){

cout<<i<<"\t"<<hash[i]<<endl;

}

}

voidsubt(){

int temp;

cout<<"enter no of data"<<endl;

cin>>total;

for(inti=0;i<total;i++){

cin>>data[i];

}

cout<<"enter size of array"<<endl;

cin>>n;

for(inti=0;i<total;i++){

temp=data[i];

temp=abs(temp-n);

temp=temp%n;

while(hash[temp]!=0){

temp=temp+1;

temp=temp%n;

}

hash[temp]=data[i];

}

for(inti=0;i<n;i++){

cout<<i<<"\t"<<hash[i]<<endl;

}

}

};

int main()

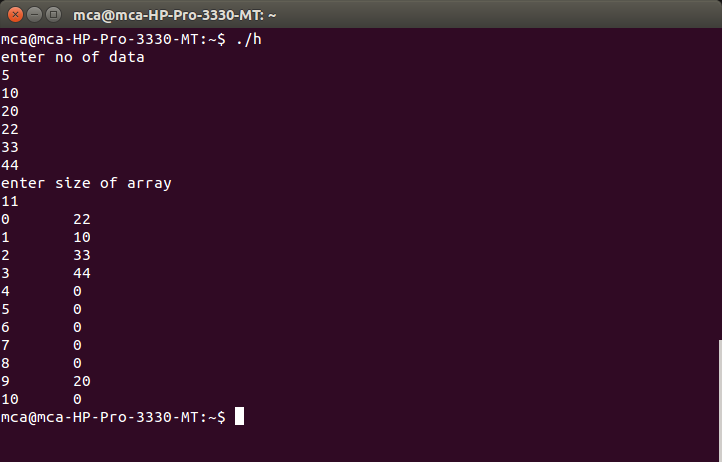
{

hashing h;

h.subt();

return 0;

}

**Output:**