Vidyavardhini's College of Engineering and Technology
Department of Artificial Intelligence & Data Science

Experiment No. 8
Implement Restoring algorithm using c-programming
Name: Apurv Kini
Roll Number: 65
Date of Performance:
Date of Submission:

Aim: To implement Restoring division algorithm using c-programming.
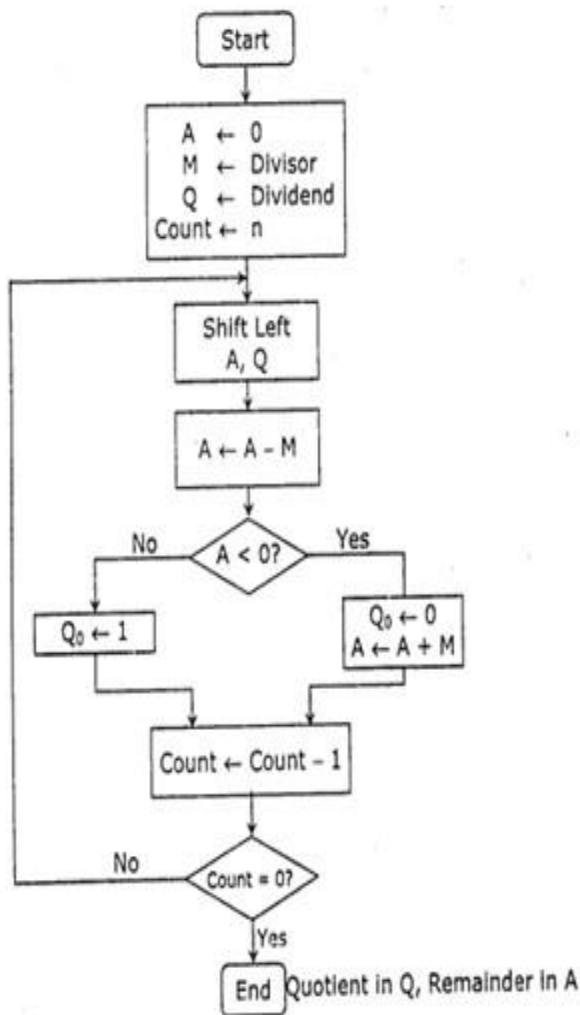
Objective -
1. To understand the working of Restoring division algorithm.
2. To understand how to implement Restoring division algorithm using c-programming.

Theory:
1) The divisor is placed in M register, the dividend placed in Q register.
2) At every step, the A and Q registers together are shifted to the left by 1-bit
3) M is subtracted from A to determine whether A divides the partial remainder. If it does, then Q0 set to 1-bit. Otherwise, Q0 gets a 0 bit and M must be added back to A to restore the previous value.
4) The count is then decremented and the process continues for n steps. At the end, the quotient is in the Q register and the remainder is in the A register.

Flowchart


```
Start
  │
  ▼
A ← 0
M ← Divisor
Q ← Dividend
Count ← n
  │
  ▼
Shift Left
A, Q
  │
  ▼
A ← A - M
  │
  ▼
  A < 0?
 No      Yes
  │        │
  ▼        ▼
Q₀ ← 1   Q₀ ← 0
         A ← A + M
  │        │
  └───┬────┘
      ▼
Count ← Count - 1
      │
      ▼
   Count = 0?
 No     Yes
         │
         ▼
End  Quotient in Q, Remainder in A
```

Perform 8 ÷ 3 by restoring division technique.

	A Register	Q Register	
Initially	0 0 0 0 0	1 0 0 0	
Shift	0 0 0 0 1	0 0 0 ☐	First Cycle
Subtract M	1 1 1 0 1		
Set Q₀	① 1 1 1 0		
Restore(A+M)	0 0 0 1 1		
	0 0 0 0 1	0 0 0 ⓪	
Shift	0 0 0 1 0	0 0 ⓪ ☐	
Subtract M	1 1 1 0 1		
Set Q₀	① 1 1 1 1		Second Cycle
Restore(A+M)	0 0 0 1 1		
	0 0 0 1 0	0 0 ⓪ ⓪	
Shift	0 0 1 0 0	0 ⓪ ⓪ ☐	
Subtract M	1 1 1 0 1		
Set Q₀	⓪ 0 0 0 1		Third Cycle
Shift	0 0 0 1 0	0 0 ⓪ ①	
Subtract M	1 1 1 0 1	⓪ ⓪ ① ☐	
Set Q₀	① 1 1 1 1		Fourth Cycle
Restore(A+M)	0 0 0 1 1		
	0 0 0 1 0	⓪ ⓪ ① ⓪	
	Remainder	Quotient	

$8 \div 3$: Remainder in A, Quotient in Q

Program-

```c
#include<stdlib.h>
#include<stdio.h>  int
acum[100]={0}   ;
 void add(int acum[],int b[],int
n);  int q[100],b[100];  int main()
{
int x,y;
 prin ("Enter the Number :");
 scanf("%d%d",&x,&y);
int i=0;
 while(x>0||y>0)
{
if(x>0)
{
```

```c
  q[i]=x%2;
x=x/2;
 }
 else
 {
 q[i]=0;
 }
 if(y>0)
 {
 b[i]=y%2;
y=y/2;
 }
 else
 {
 b[i]=0;
 }
 i++;
 }

 int   n=i;     int
bc[50];      prin
("\n");
for(i=0;i<n;i++
)
 {
 if(b[i]==0)
 {
 bc[i]=1;
 }
 else
 {
 bc[i]=0;
 }
 }
 bc[n]=1;
for(i=0;i<=n;i++)
 {
 if(bc[i]==0)
 {
 bc[i]=1;
 i=n+2;
 }
 else
 {
 bc[i]=0;
```

```c
    }
  }
  int l;
  b[n]=0;
  int k=n; int
  n1=n+n-1;
  int j,mi=n-
  1;
  for(i=n;i!=0;i--)
  {
  for(j=n;j>0;j--)
  {
  acum[j]=acum[j-1];

  }
  acum[0]=q[n-1];
  for(j=n-1;j>0;j--)
  {
  q[j]=q[j-1];
  }

  add(acum,bc,n+1);
  if(acum[n]==1)
  {
  q[0]=0;
  add(acum,b,n+1);
  }
  else
  {
  q[0]=1;
  }
  }
  prin ("\nQuoient   : ");

  for(  l=n-1;l>=0;l--)
  {
  prin ("%d",q[l]);

  }
  prin ("\nRemainder : ");
  for( l=n;l>=0;l--)
  {
  prin ("%d",acum[l]);
  }
  return 0;
```

```
}
void add(int acum[],int bo[],int n)
{
int i=0,temp=0,sum=0;
for(i=0;i<n;i++)
{
sum=0;
sum=acum[i]+bo[i]+temp;
if(sum==0)
{
acum[i]=0;
temp=0;
}
else if (sum==2)
{
acum[i]=0;
temp=1;
}
else if(sum==1)
{
acum[i]=1;
temp=0;
}
else if(sum==3)
{
acum[i]=1;
temp=1;
}
}
}
```

Output –

```
Input:
15 7

Output:
Enter the Number :

Quoient: 0010
Remainder: 00001
```

Conclusion –

In this experiment, we learned about the division algorithm in computer architecture which is the Restoring Algorithm.