```c
//SHREYAS SAWANT 55 D7A
//To perform Restoring Division Algorithm

#include<conio.h>
#include<stdio.h>
#include<math.h>
#define size 5


int A[size];//Accumulator
int Ad[size];
int M[size];//Divisor
int Q[4];//Dividend
int n=size-1,C[size];//Complement of Divisor
int a,b,t;

void lbs()  //To perform Left Bit Shift
{
    for(int i=0;i<size;i++)
    {
        A[i]=A[i+1];
    }
    A[n]=Q[n-1];
    for(int i=n-1;i>0;i--)
    {
        Q[i]=Q[i-1];
    }
    printf("\n\n\nIteration %d",n-t+1);
    printf("\nAfter left shift");
    printf("\nA: ");
    for(int i=0;i<size;i++)
        printf("%d ",A[i]);
     printf("\nQ: ");
    for(int i=n-1;i>0;i--)
        printf("%d ",Q[i]);
}

void bitcoin()  //To convert to binary
{
    int t=size,i=0,j=0,r1,r2;
    while(t!=0)
    {
        r1=b%2;
        r2=a%2;
        Q[i]=r1;
        M[j]=r2;

        a/=2;
        b/=2;
        if(i<n)
            i++;
        j++;
        t--;
    }
}
void complement()   //To get 2's complement of M
{   int c=1;
    for(int i=n;i>-1;i--)
    {
        if(M[i]==1)
            C[i]=0;
        else
            C[i]=1;
    }

    for(int i=0;i<size;i++)
    {
        if(C[i]+c==2)
        {
            C[i]=0,c=1;
        }
        else
        {
            C[i]=1;break;
        }
    }
    printf("\n2's Complement of Divisor: ");
    for(int i=n;i>-1;i--)
    printf("%d ",C[i]);
    printf("\n\n");
}

void recover()  //To store copy of A,later to be used in recovering A
{
```

```
85      for(int i=0;i<size;i++)
86      {
87          A[i]=Ad[i];
88      }
89      printf("Since MSB of A is 1");
90      printf("\nRestored A: ");
91      for(int i=0;i<size;i++)
92          {printf("%d ",A[i]);}
93      printf("\nNew Q: ");
94      for(int i=n-1;i>-1;i--)
95          {printf("%d ",Q[i]);}
96          printf("\n\n");
97  }
98  void add()       //To perform A-M
99  {
100     int c=0;
101     for(int i=0;i<size;i++)
102     {
103         Ad[i]=A[i];
104     }
105     for(int i=0;i<size;i++)
106     {
107         if(C[i]+A[n-i]+c==2)
108         {
109             A[n-i]=0;c=1;
110         }
111         else if(C[i]+A[n-i]+c==3)
112         {
113             A[n-i]=1;c=1;
114         }
115         else if(C[i]+A[n-i]+c==1)
116         {
117             A[n-i]=1;c=0;
118         }
119         else
120         {
121             A[n-i]=0;c=0;
122         }
123     }
124     printf("\n\nPerforming A-M");
125     printf("\nA-M= ");
126     for(int i=0;i<size;i++)
127         printf("%d ",A[i]);
128     printf("\n\n");
129 }
130
131
132 void rda()  //To iterate and perform the algorithm
133 {   t=n;
134     lbs();
135     while(t!=0)
136     {
137         add();
138         if(A[0]==1)
139          {
140              Q[0]=0;
141              recover();
142          }
143          else
144          {  printf("Since MSB of A is 0");
145             printf("\nA: ");
146             for(int i=0;i<size;i++)
147             {
148                 printf("%d ",A[i]);
149             }
150              Q[0]=1;
151              printf("\nNew Q: ");
152             for(int i=n-1;i>-1;i--)
153             {printf("%d ",Q[i]);}
154             printf("\n\n");
155          }
156         t--;
157         if(t!=0)
158             lbs();
159     }
160
161 }
162 int main()
163 {
164     printf("Enter positive numbers less than 16");
165
166     printf("\nEnter dividend ");
167     scanf("%d",&b);
168     printf("Enter divisor ");
```

```
169         scanf("%d",&a);
170         if(a==0)
171         {
172             printf("\nINVALID\n");return 0;
173         }
174         bitcoin();
175 
176         printf("\nDividend (Q) in binary is: ");
177         for(int i=n-1;i>=0;i--)
178             printf("%d ",Q[i]);
179 
180         printf("\n\nDivisor (M) in binary is: ");
181         for(int i=n;i>=0;i--)
182             printf("%d ",M[i]);
183 
184         printf("\n\nAccumulator (A) : ");
185         for(int i=0;i<size;i++)
186             printf("%d ",A[i]);
187 
188         printf("\n");
189         complement();
190         rda();
191         int c=0;
192         printf("\n\nQUOTIENT:  ");
193         for(int i=n-1;i>=0;i--)
194         {
195             printf("%d ",Q[i]);
196             c+=(int)pow(2,n-1-i)*Q[n-1-i];
197         }
198         printf("= %d",c);
199         c=0;
200         printf("\nREMAINDER: ");
201         for(int i=0;i<size;i++)
202         {
203             printf("%d ",A[i]);
204             c+=(int)(pow(2,i)*A[n-i]);
205         }
206         printf("= %d",c);
207         printf("\n\n");
208 
209     }
210 
211 
212 
213 
214 
```