Batch: A3 Roll No.:16010121051
Experiment / assignment / tutorial No.4
Grade: AA / AB / BB / BC / CC / CD /DD
Signature of the Staff In-charge with date

TITLE : To study and implement Non Restoring method of division

AIM : The basis of algorithm is based on paper and pencil approach and the operation involve repetitive shifting with addition and subtraction. So the main aim is to depict the usual process in the form of an algorithm.

Expected OUTCOME of Experiment: (Mention CO/CO's attained here)

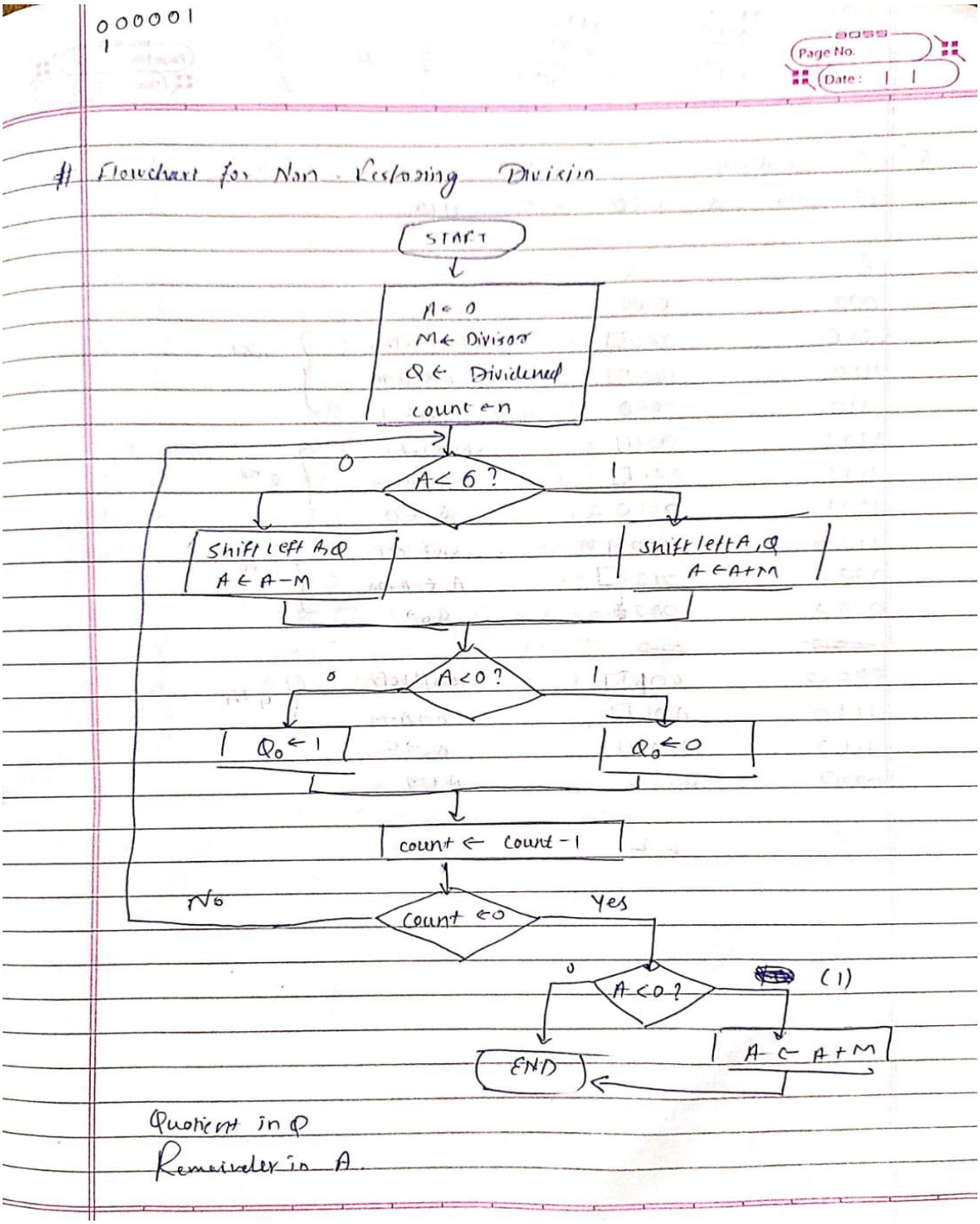
Books/ Journals/ Websites referred:

1. Carl Hamacher, Zvonko Vranesic and Safwat Zaky, "Computer Organization", Fifth Edition, TataMcGraw-Hill.
2. William Stallings, "Computer Organization and Architecture: Designing for Performance", Eighth Edition, Pearson.
3. Dr. M. Usha, T. S. Srikanth, "Computer System Architecture and Organization", First Edition, Wiley-India.

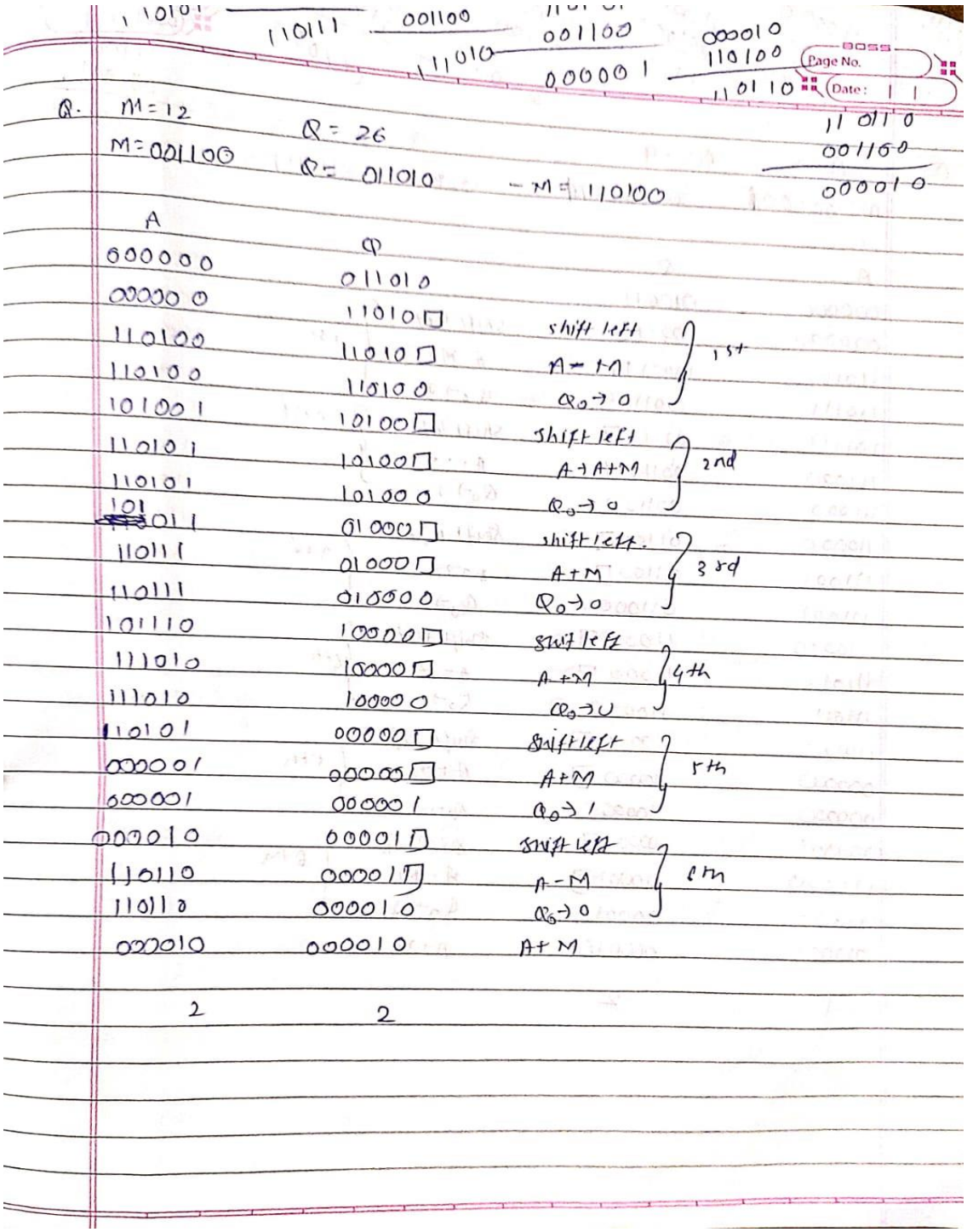
Pre Lab/ Prior Concepts:

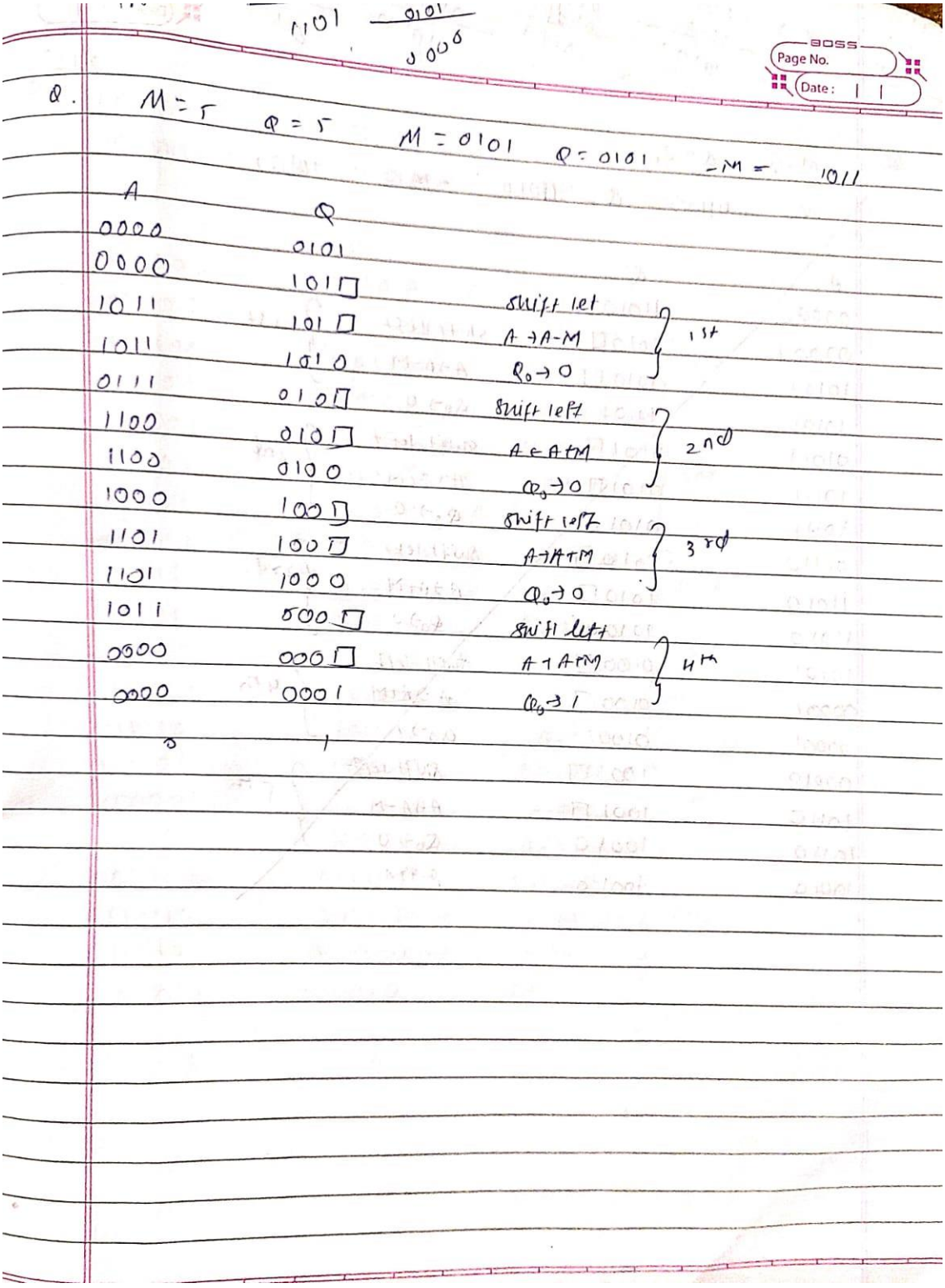
The Non Restoring algorithm works with any combination of positive and negative numbers.

Flowchart for Non Restoring of Division:



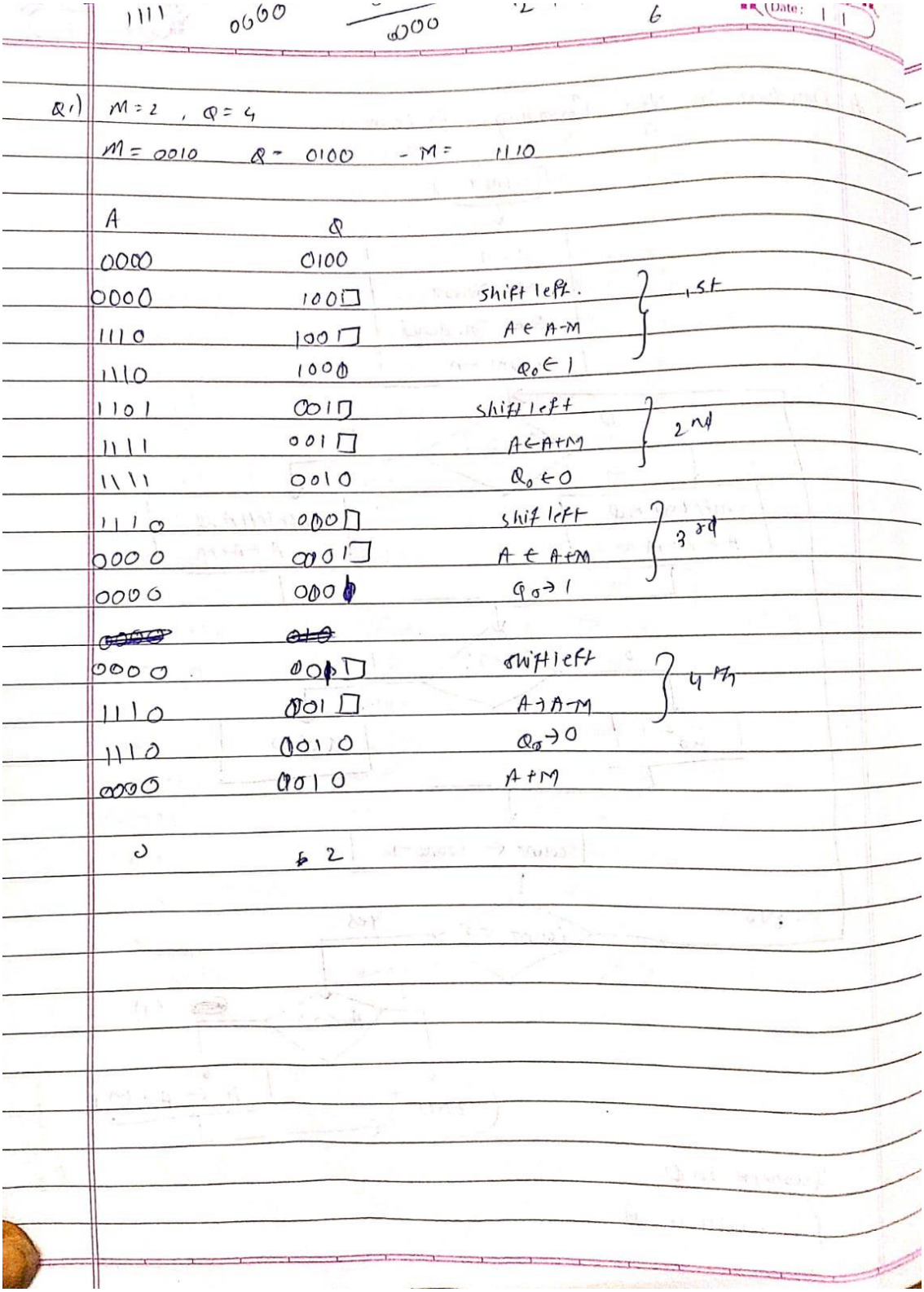
Example:

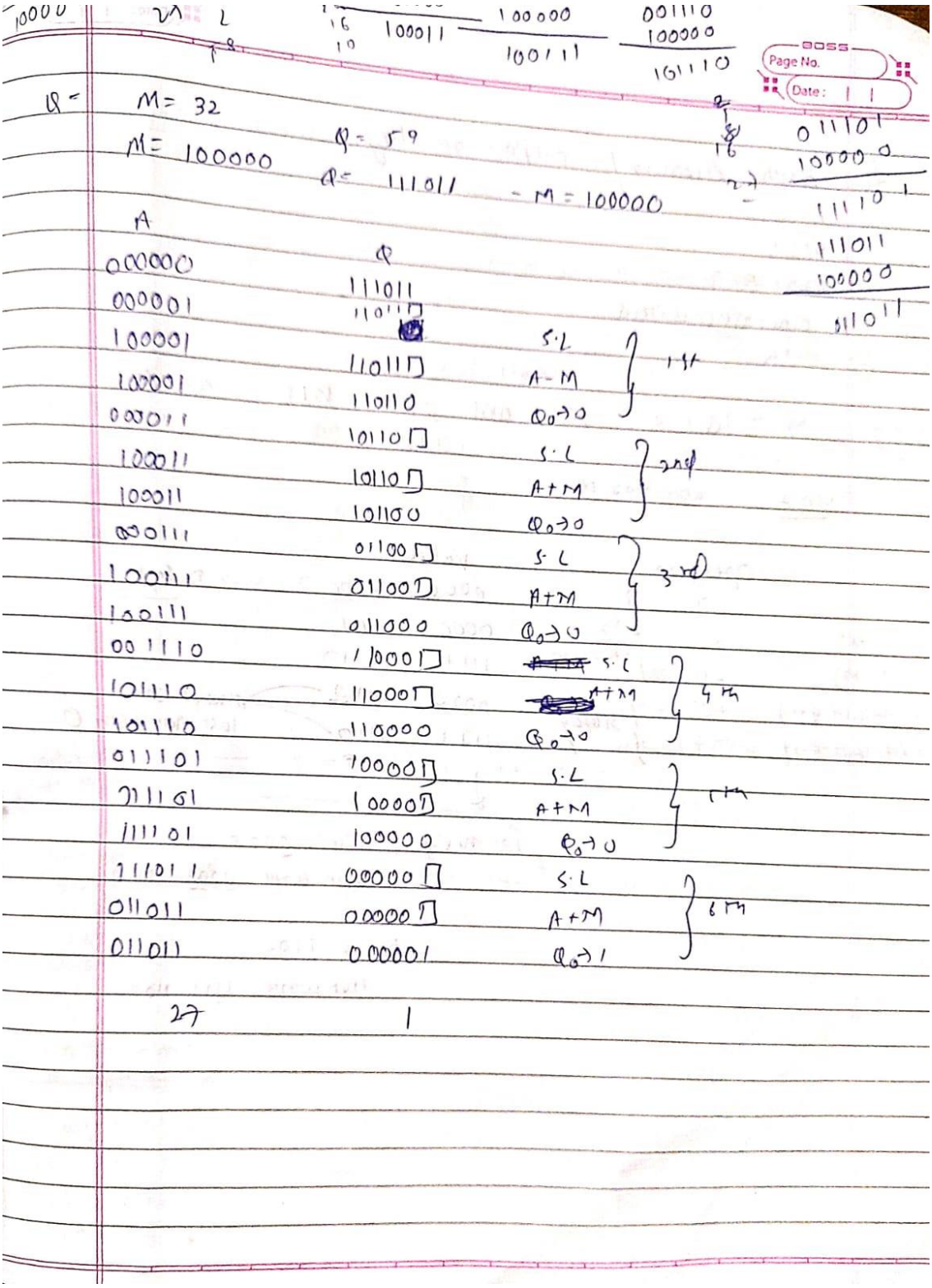

Q. M=12 Q=26
M=001100 Q=011010 -M=110100
A Q
000000 011010
shift left
A=A+M
Q0→0
1st
shift left
A→A+M
Q0→0
2nd
shift left
A+M
Q0→0
3rd
shift left
A+M
Q0→0
4th
Shift left
A+M
Q0→1
5th
Shift left
A-M
Q0→0
6th
A+M
2 2



Q. M = 5 Q = 5 M = 0101 Q = 0101 -M = 1011

A	Q		
0000	0101		
0000	1010	Shift left	1st
1011	1010	A → A-M	
1011	1010	$Q_0 \rightarrow 0$	
0111	0101	Shift left	2nd
1100	0101	A ← A+M	
1100	0100	$Q_0 \rightarrow 0$	
1000	1001	Shift left	3rd
1101	1001	A → A+M	
1101	1000	$Q_0 \rightarrow 0$	
1011	0001	Shift left	4th
0000	0001	A → A+M	
0000	0001	$Q_0 \rightarrow 1$	


Q1) M = 2 , Q = 4
M = 0010 Q = 0100 -M = 1110
A Q
0000 0100
Shift left.
A ← A-M
Q0 ← 1
1st
Shift left
A ← A+M
Q0 ← 0
2nd
shift left
A ← A+M
Q0 → 1
3rd
Shift left
A → A-M
Q0 → 0
A+M
4th


M = 32
M = 100000
Q = 59
Q = 111011 - M = 100000
A
Q
S.L
A-M
Q0→0
1st
S.L
A+M
Q0→0
2nd
S.L
A+M
Q0→0
3rd
S.L
A+M
Q0→0
4th
S.L
A+M
Q0→0
5th
S.L
A+M
Q0→1
6th
27
1

Implementation:

```
#include <math.h>
#include <stdio.h>
//NON RESTORING DIVISION
int main()
{
int a[50],a1[50],b[50],d=0,i,j;
 int n1,n2, c, k1,k2,n,k,quo=0,rem=0;
   printf("Enter the number of bits\n");
   scanf("%d",&n);
 printf("Enter the divisor and dividend\n");
 scanf("%d %d", &n1,&n2);

 for (c = n-1; c >= 0; c--)//converting the 2 nos to binary
 {
  k1 = n1 >> c;

  if (k1 & 1)
    a[n-1-c]=1;// M
  else
   a[n-1-c]=0;

   k2 = n2 >> c;
```

```
 if (k2 & 1)
   b[2*n-1-c]=1;// Q
 else
  b[2*n-1-c]=0;

}

for(i=0;i<n;i++)//making complement
{
   if(a[i]==0)
     a1[i]=1;
   else
     a1[i]=0;
}

a1[n-1]+=1;//twos complement ie -M

if(a1[n-1]==2)
{
      for(i=n-1;i>0;i--)
   {
        if(a1[i]==2)
      {
         a1[i-1]+=1;
```

```
            a1[i]=0;
          }
       }
   }
  if(a1[0]==2)
    a1[0]=0;

  for( i=0;i<n;i++)// putting A in the same array as Q
  {
     b[i]=0;

  }

printf("A\tQ\tPROCESS\n");

  for(i=0;i<2*n;i++)
{
    if(i==n)
        printf("\t");

    printf("%d",b[i]);
}
 printf("\n");
```

```
for(k=0;k<n;k++)//n iterations
{
   for(j=0;j<2*n-1;j++)//left shift
    {
     b[j]=b[j+1];


    }


    for(i=0;i<2*n -1;i++)
    {
       if(i==n)
           printf("\t");
       printf("%d",b[i]);
    }printf("_");


    printf("\tLEFT SHIFT\n");


       if(b[0]==0)
       {
                for(i=n-1;i>=0;i--)//A=A-M
                {
                   b[i]+=a1[i];


                       if(i!=0)
```

```
  {
     if(b[i]==2)
           {
              b[i-1]+=1;
              b[i]=0;
           }
     if(b[i]==3)
           {
              b[i-1]+=1;
              b[i]=1;
           }
          // printf("%d",b[i]);
    }
}
     if(b[0]==2)
        b[0]=0;

     if(b[0]==3)
        b[0]=1;

for(i=0;i<2*n -1;i++)
{
   if(i==n)
     printf("\t");
```

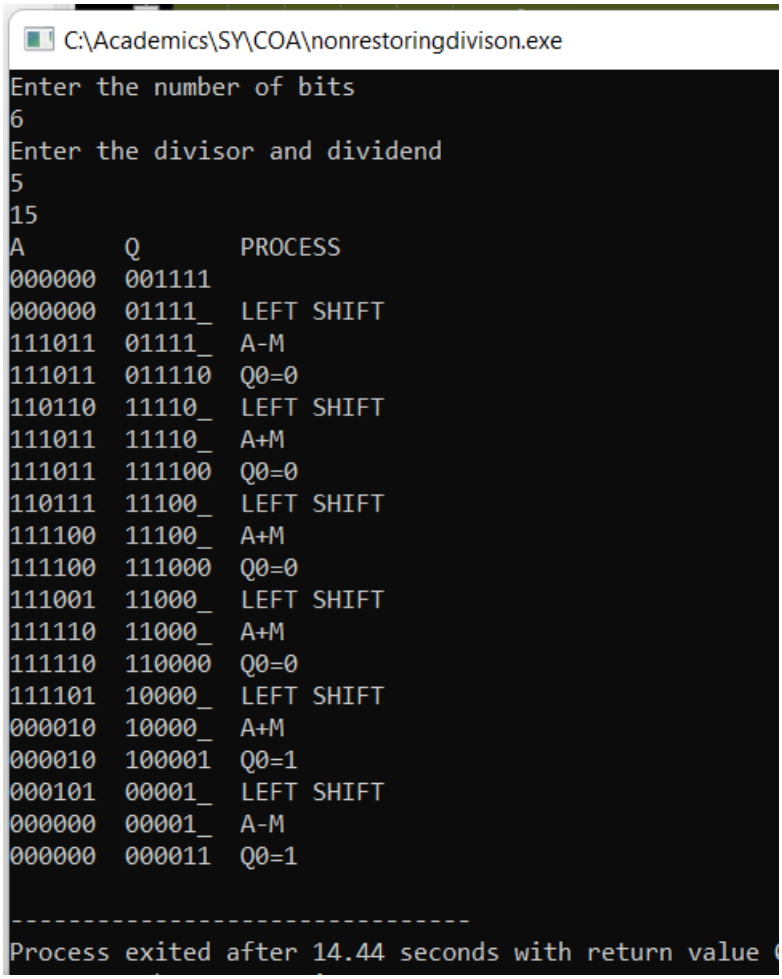
```
            printf("%d",b[i]);
        }printf("_");

        printf("\tA-M\n");
}

else
{
        for(j=n-1;j>=0;j--)//A=A+M
          {
            b[j]+=a[j];

            if(j!=0)
          {
            if(b[j]==2)
                {
                  b[j-1]+=1;
                  b[j]=0;
                }
            if(b[j]==3)
                {
```

```
                b[j-1]+=1;
                b[j]=1;
              }
    }

      if(b[0]==2)
         b[0]=0;

      if(b[0]==3)
         b[0]=1;
    }

    for(i=0;i<2*n -1;i++)
{
    if(i==n)
       printf("\t");



    printf("%d",b[i]);
}printf("_");

printf("\tA+M\n");
```

```
}



if(b[0]==0)//A==0?
{
   b[2*n-1]=1;
   for(i=0;i<2*n ;i++)
     {
        if(i==n)
            printf("\t");



        printf("%d",b[i]);
     }

     printf("\tQ0=1\n");
}



if(b[0]==1)//A==1?
{
```

```
        b[2*n-1]=0;
        for(i=0;i<2*n ;i++)
          {
            if(i==n)
               printf("\t");


            printf("%d",b[i]);
          }

          printf("\tQ0=0\n");

      }

}}
```

Output:



Conclusion:

Non restoring divison was implemented successfully.

Post Lab Descriptive Questions

1. **What are the advantages of non restoring division over restoring division?**

The advantage of using non-restoring arithmetic over the standard restoring division is that a test subtraction is not required; the sign bit determines whether an addition or subtraction is used.

Date: _____ **Signature of faculty in-charge**