# ASSIGNMENT 6 -U19CS076

```
#include <stdio.h>

#include <conio.h>

#include <math.h>


int a=0,b=0,c=0,com[5]={1,0,0,0,0},s=0;

int anum[5]={0},anumcp[5] ={0},bnum[5]={0};

int acomp[5]={0},bcomp[5]={0},rem[5]={0},quo[5]={0},res[5]={0};


void binary(){


    int r, r2, i, temp;

    for(i = 0; i < 5; i++){

        r = a % 2;

        a = a / 2;

        r2 = b % 2;

        b = b / 2;

        anum[i] = r;

        anumcp[i] = r;

        bnum[i] = r2;

        if(r2 == 0){                    //finding one s complement

            bcomp[i] = 1;

        }
```

```
        if(r == 0){
            acomp[i] =1;
        }
    }
  //part for two's complementing
  c = 0;
  for( i = 0; i < 5; i++){
        res[i] = com[i]+ bcomp[i] + c;
        if(res[i]>=2){
            c = 1;
        }
        else
            c = 0;
        res[i] = res[i]%2;
    }
  for(i = 4; i>= 0; i--){
    bcomp[i] = res[i];
  }
}
void add(int num[]){
    int i;
    c = 0;
    for( i = 0; i < 5; i++){
        res[i] = rem[i]+ num[i] + c;
        if(res[i]>=2){
```

```
            c = 1;
        }
        else
            c = 0;
        res[i] = res[i]%2;
    }
    for(i = 4; i>= 0; i--){
        rem[i] = res[i];
                    }
}
void shl(){//for shift left
    int i;
    for(i = 4; i > 0  ; i--){//shift the remainder
        rem[i] = rem[i-1];
    }
    rem[0] = anumcp[4];
    for(i = 4; i > 0  ; i--){
        anumcp[i] = anumcp[i-1];
    }
    anumcp[0] = 0;


}


void main(){
```

```
int i;
printf("NON-RESTORING DIVISION ALGORITHM");
printf("\nEnter two numbers to multiply: ");
printf("\nBoth must be less than 16");
// for two numbers each below 16
do{
    printf("\nEnter A: ");
    scanf("%d",&a);
    printf("Enter B: ");
    scanf("%d",&b);
}while(a>=16 || b>=16);
binary();
printf("\n\nUnsigned Binary Equivalents are: ");
printf("\nA = ");
for(i = 4; i>= 0; i--){
    printf("%d",anum[i]);
}
printf("\nB = ");
for(i = 4; i>= 0; i--){
    printf("%d",bnum[i]);
}
printf("\nB'+ 1 = ");
for(i = 4; i>= 0; i--){
    printf("%d",bcomp[i]);
}
```

```
    printf("\n\n-->");
    //division part
    shl();
    for(i=0;i<4;i++){
            shl();                  //SHIFT LEFT AS FIRST STEP IN NON-RESTORING
        if(rem[4]==1)
{
                                        //simply add for restoring
             add(bnum);
             if(rem[4]==1)
             {
                        anumcp[0] = 0;
            }
            else
                        anumcp[0]=1;
    }
        else{
            add(bcomp);//subtract b
             if(rem[4]==1)
             {
                        anumcp[0] = 0;
            }
            else
                        anumcp[0]=1;
        }
```

```
    }
    if(rem[4]==1)
    {
            add(bnum);
 }

    printf("\nRemainder is = ");int decimal_val=0;
    for(i = 4; i>= 0; i--){
         printf("%d",rem[i]);
         decimal_val = decimal_val + rem[i] *pow(2,i);
    }
    printf("\nQuotient is = ");int q=0;
    for(i = 4; i>= 0; i--){
         printf("%d",anumcp[i]);
          q = q + anumcp[i] *pow(2,i);
    }

   printf("\nDecimal value of reminder is %d",decimal_val);
   printf("\nDecimal value of quotient is %d",q);
            getch();
}
```

```
"C:\Users\krithikha\Desktop\svnit\sem3\comp org\assgn6.exe"
NON-RESTORING DIVISION ALGORITHM
Enter two numbers to multiply:
Both must be less than 16
Enter A: 5
Enter B: 3


Unsigned Binary Equivalents are:
A = 00101
B = 00011
B'+ 1 = 11101

-->
Remainder is = 00010
Quotient is = 00001
Decimal value of reminder is 2
Decimal value of quotient is 1
```

**SIGNED NUMBERS**

**RESTORING DIVISION**

```
#include <stdio.h>

#include <conio.h>

#include <math.h>


int a=0,b=0,c=0,com[5]={1,0,0,0,0},s=0;

int anum[5]={0},anumcp[5] ={0},bnum[5]={0};
```

```
int acomp[5]={0},bcomp[5]={0},rem[5]={0},quo[5]={0},res[5]={0};


void binary(){
    int r, r2, i, temp;
    for(i = 0; i < 5; i++){
        r = a % 2;
        a = a / 2;
        r2 = b % 2;
        b = b / 2;
        anum[i] = r;
        anumcp[i] = r;
        bnum[i] = r2;
        if(r2 == 0){                    //finding one s complement
            bcomp[i] = 1;
        }
        if(r == 0){
            acomp[i] =1;
        }
    }
  //part for two's complementing
  c = 0;
  for( i = 0; i < 5; i++){
        res[i] = com[i]+ bcomp[i] + c;
        if(res[i]>=2){
            c = 1;
```

```
        }
        else
            c = 0;
        res[i] = res[i]%2;
    }
  for(i = 4; i>= 0; i--){
    bcomp[i] = res[i];
  }
}
void add(int num[]){
    int i;
    c = 0;
    for( i = 0; i < 5; i++){
        res[i] = rem[i]+ num[i] + c;
        if(res[i]>=2){
            c = 1;
        }
        else
            c = 0;
        res[i] = res[i]%2;
    }
    for(i = 4; i>= 0; i--){
        rem[i] = res[i];
                    }
}
```

```
void shl(){//for shift left
    int i;
    for(i = 4; i > 0  ; i--){//shift the remainder
        rem[i] = rem[i-1];
    }
    rem[0] = anumcp[4];
    for(i = 4; i > 0  ; i--){//shift the remtient
        anumcp[i] = anumcp[i-1];
    }
    anumcp[0] = 0;

}

void main(){

    int i;
    printf("RESTORING DIVISION ALGORITHM");
    printf("\nEnter two numbers to divide: ");
    printf("\nBoth must be less than 16");
    // for two numbers each below 16
    do{
        printf("\nEnter A: ");
        scanf("%d",&a);
        printf("Enter B: ");
        scanf("%d",&b);
```

```
 }while(a>=16 || b>=16);


int a1=a;

int b1=b;

        a=abs(a);

        b=abs(b);


 binary();

 printf("\n\nBinary Equivalents(IGNORING SIGN)are: ");

 printf("\nA = ");

 for(i = 4; i>= 0; i--){

     printf("%d",anum[i]);

 }

 printf("\nB = ");

 for(i = 4; i>= 0; i--){

     printf("%d",bnum[i]);

 }

 printf("\nB'+ 1 = ");

 for(i = 4; i>= 0; i--){

     printf("%d",bcomp[i]);

 }

 printf("\n\n-->");

 //division part

 shl();

 for(i=0;i<5;i++){
```

```
        add(bcomp);      //to subtract B
        if(rem[4]==1){//simply add for restoring
            anumcp[0] = 0;
            add(bnum);
        }
        else{
            anumcp[0] = 1;
        }
        if(i<4)
            shl();

    }
   if(a1>0)                                //for positive divident
{
    printf("\nRemainder is = ");int decimal_val=0;
    for(i = 4; i>= 0; i--){
        printf("%d",rem[i]);
        decimal_val = decimal_val + rem[i] *pow(2,i);
    }
      printf("\nDecimal value of reminder is %d",decimal_val);

    }
```

```
        else if(a1<0)
        {
                printf("\nRemainder is = ");int decimal_val=0;
for(i = 4; i>= 0; i--){
    decimal_val = decimal_val + rem[i] *pow(2,i);
}
for(i = 4; i>= 0; i--){                //ones compliment
        if(rem[i]==1)
                rem[i]=0;
        else if(rem[i]==0)
                rem[i]=1;
                }
         c = 0;
        for( i = 0; i < 5; i++){
     rem[i] = com[i]+ rem[i] + c;
    if(rem[i]>=2){
         c = 1;
    }
    else
         c = 0;
    rem[i] = rem[i]%2;
        }
        for(i = 4; i>= 0; i--){
    printf("%d",rem[i]);
         }
```

```
        printf("\nDecimal value of reminder is %d",(decimal_val*-1));
            }
if((a1>0&&b1>0)|| (a1<0&&b1<0))
   {
            printf("\nQuotient is = ");int q=0;
    for(i = 4; i>= 0; i--){
         printf("%d",anumcp[i]);
          q = q + anumcp[i] *pow(2,i);
   }
   printf("\nDecimal value of quotient is %d",q);
   }
   else
   {
            printf("\nQuotient is = ");int q=0;
    for(i = 4; i>= 0; i--){
          q = q + anumcp[i] *pow(2,i);
   }
    for(i = 4; i>= 0; i--){                //ones compliment
                                    if(anumcp[i]==1)
                                              anumcp[i]=0;
                                    else if(anumcp[i]==0)
                                    anumcp[i]=1;
                                    }
                                     c = 0;
            for( i = 0; i < 5; i++){
```

```
 anumcp[i] = com[i]+ anumcp[i] + c;
if(anumcp[i]>=2){
    c = 1;
}
else
    c = 0;
anumcp[i] = anumcp[i]%2;
  }
  for(i = 4; i>= 0; i--){
printf("%d",anumcp[i]);
                }
printf("\nDecimal value of quotient is %d",q*-1);
}
  getch();
}
```

```
"C:\Users\krithikha\Desktop\svnit\sem2\fcp\assgn6 signed restoring.exe"
RESTORING DIVISION ALGORITHM
Enter two numbers to divide:
Both must be less than 16
Enter A: -12
Enter B: 5


Binary Equivalents(IGNORING SIGN)are:
A = 01100
B = 00101
B'+ 1 = 11011

-->
Remainder is = 11110
Decimal value of reminder is -2
Quotient is = 11110
Decimal value of quotient is -2
```

**USING NON-RESTORING DIVISION**

```
#include <stdio.h>

#include <conio.h>

#include <math.h>


int a=0,b=0,c=0,com[5]={1,0,0,0,0},s=0;

int anum[5]={0},anumcp[5] ={0},bnum[5]={0};

int acomp[5]={0},bcomp[5]={0},rem[5]={0},quo[5]={0},res[5]={0};
```

```
void binary(){


    int r, r2, i, temp;
    for(i = 0; i < 5; i++){
        r = a % 2;
        a = a / 2;
        r2 = b % 2;
        b = b / 2;
        anum[i] = r;
        anumcp[i] = r;
        bnum[i] = r2;
        if(r2 == 0){                    //finding one s complement
            bcomp[i] = 1;
        }
        if(r == 0){
            acomp[i] =1;
        }
    }
  //part for two's complementing
  c = 0;
  for( i = 0; i < 5; i++){
        res[i] = com[i]+ bcomp[i] + c;
        if(res[i]>=2){
            c = 1;
        }
```

```
        else
            c = 0;
        res[i] = res[i]%2;
    }
  for(i = 4; i>= 0; i--){
    bcomp[i] = res[i];
  }
}
void add(int num[]){
    int i;
    c = 0;
    for( i = 0; i < 5; i++){
        res[i] = rem[i]+ num[i] + c;
        if(res[i]>=2){
            c = 1;
        }
        else
            c = 0;
        res[i] = res[i]%2;
    }
    for(i = 4; i>= 0; i--){
        rem[i] = res[i];
                    }
}
void shl(){//for shift left
```

```
    int i;
    for(i = 4; i > 0  ; i--){//shift the remainder
        rem[i] = rem[i-1];
    }
    rem[0] = anumcp[4];
    for(i = 4; i > 0  ; i--){//shift the remtient
        anumcp[i] = anumcp[i-1];
    }
    anumcp[0] = 0;

}

void main(){
    int i;
    printf("SIGNED NON-RESTORING DIVISION ALGORITHM");
    printf("\nEnter two numbers to multiply: ");
    printf("\nBoth must be less than 16");
    // for two numbers each below 16
    do{
        printf("\nEnter A: ");
        scanf("%d",&a);
        printf("Enter B: ");
        scanf("%d",&b);
    }while(a>=16 || b>=16);
```

```
int a1=a;
int b1=b;
        a=abs(a);
        b=abs(b);
 binary();


 printf("\n\nUnsigned Binary Equivalents(ignoring signgs) are: ");
 printf("\nA = ");
 for(i = 4; i>= 0; i--){
     printf("%d",anum[i]);
 }
 printf("\nB = ");
 for(i = 4; i>= 0; i--){
     printf("%d",bnum[i]);
 }
 printf("\nB'+ 1 = ");
 for(i = 4; i>= 0; i--){
     printf("%d",bcomp[i]);
 }
 printf("\n\n-->");
 //division part
 shl();
 for(i=0;i<4;i++){
        shl();
     if(rem[4]==1){
```

```
                    //simply add for restoring
        add(bnum);
        if(rem[4]==1)
        {
        anumcp[0] = 0;
       }
       else
       anumcp[0]=1;
    }
    else{
       add(bcomp);//subtract b
        if(rem[4]==1)
        {
                anumcp[0] = 0;
       }
       else
                anumcp[0]=1;
    }

}
if(rem[4]==1)
{
       add(bnum);
        }
       if(a1>0)
```

```
        {

printf("\nRemainder is = ");int decimal_val=0;
for(i = 4; i>= 0; i--){
     printf("%d",rem[i]);
     decimal_val = decimal_val + rem[i] *pow(2,i);
}
   printf("\nDecimal value of reminder is %d",decimal_val);

}
        else if(a1<0)
        {
                    printf("\nRemainder is = ");int decimal_val=0;
for(i = 4; i>= 0; i--){
     decimal_val = decimal_val + rem[i] *pow(2,i);
}
for(i = 4; i>= 0; i--){                //ones compliment
        if(rem[i]==1)
                    rem[i]=0;
        else if(rem[i]==0)
                    rem[i]=1;
        }
         c = 0;
        for( i = 0; i < 5; i++){
      rem[i] = com[i]+ rem[i] + c;
```

```
        if(rem[i]>=2){
            c = 1;
        }
        else
            c = 0;
        rem[i] = rem[i]%2;
          }
          for(i = 4; i>= 0; i--){
        printf("%d",rem[i]);
                        }
     printf("\nDecimal value of reminder is %d",(decimal_val*-1));
          }
          if((a1>0&&b1>0)|| (a1<0&&b1<0))
 {
          printf("\nQuotient is = ");int q=0;
  for(i = 4; i>= 0; i--){
        printf("%d",anumcp[i]);
         q = q + anumcp[i] *pow(2,i);
}
 printf("\nDecimal value of quotient is %d",q);
 }
 else
 {
          printf("\nQuotient is = ");int q=0;
  for(i = 4; i>= 0; i--){
```

```
        q = q + anumcp[i] *pow(2,i);
    }
    for(i = 4; i>= 0; i--){                //ones compliment
            if(anumcp[i]==1)
                    anumcp[i]=0;
            else if(anumcp[i]==0)
                    anumcp[i]=1;
                    }
            c = 0;
   for( i = 0; i < 5; i++){
        anumcp[i] = com[i]+ anumcp[i] + c;
        if(anumcp[i]>=2){
            c = 1;
        }
        else
            c = 0;
        anumcp[i] = anumcp[i]%2;
           }
           for(i = 4; i>= 0; i--){
        printf("%d",anumcp[i]);
                    }
    printf("\nDecimal value of quotient is %d",q*-1);
    }
            getch();
}
```

"C:\Users\krithikha\Desktop\svnit\sem3\comp org\assgn6 signed.exe"

```
SIGNED NON-RESTORING DIVISION ALGORITHM
Enter two numbers to multiply:
Both must be less than 16
Enter A: 4
Enter B: -3


Unsigned Binary Equivalents(ignoring signgs) are:
A = 00100
B = 00011
B'+ 1 = 11101

-->
Remainder is = 00001
Decimal value of reminder is 1
Quotient is = 11111
Decimal value of quotient is -1
```