CO ASSIGNMENT-8

-RUSHIL JARIWALA (U21CS001)

Restoring Division Algorithm Implementation

```
#include <stdio.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()
{
   a = fabs(a);
   b = fabs(b);
   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)
       {
           bcomp[i] = 1;
       }
       if (r == 0)
       {
           acomp[i] = 1;
       }
   }
   // part for two's complementing
```

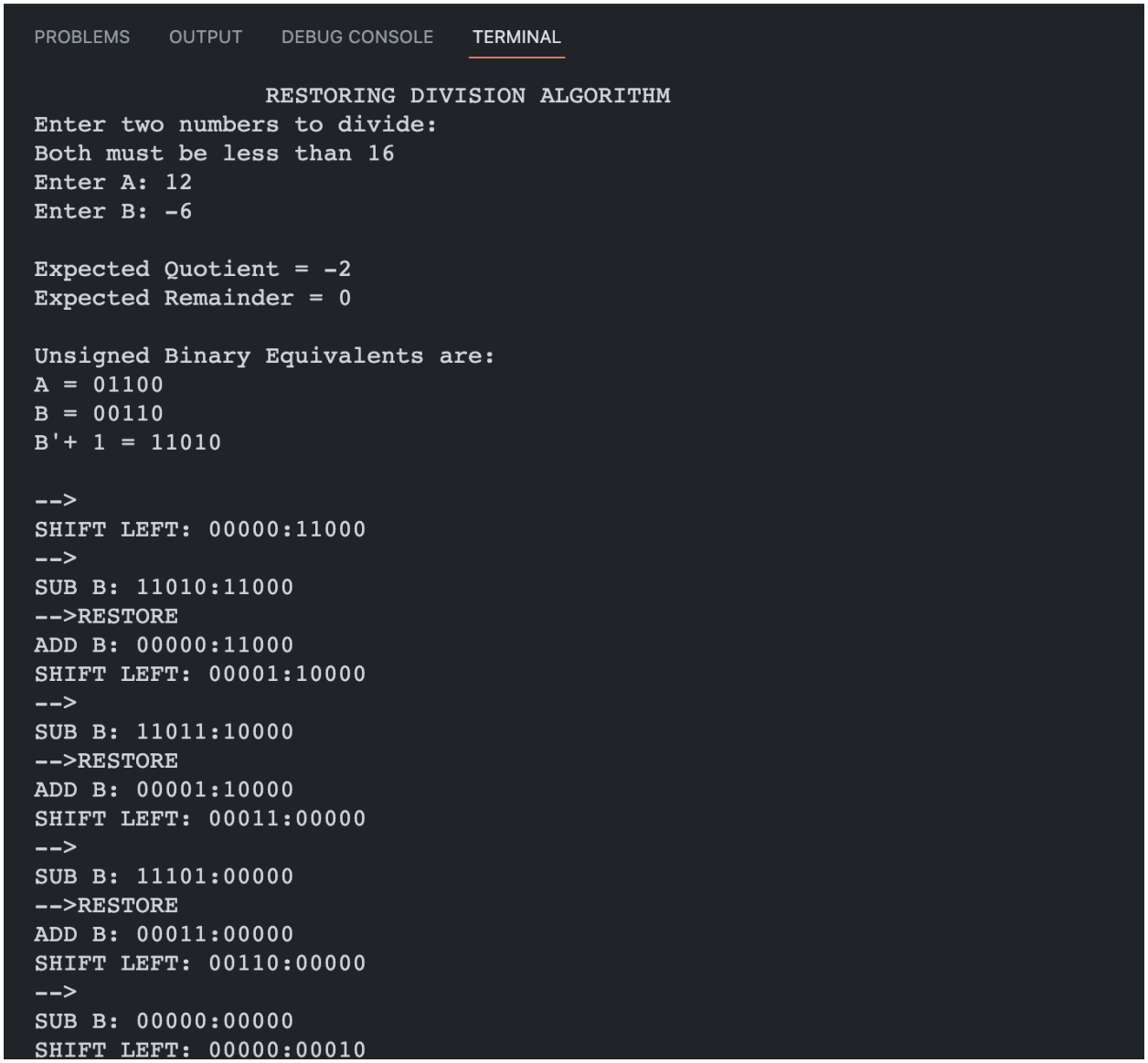
```
    c = 0;
    for (i = 0; i < 5; i++)
    {
        res[i] = com[i] + bcomp[i] + c;
        if (res[i] >= 2)
        {
            c = 1;
        }
        else
            c = 0;
        res[i] = res[i] % 2;
    }
    for (i = 4; i >= 0; i--)
    {
        bcomp[i] = res[i];
    }
}
void add(int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

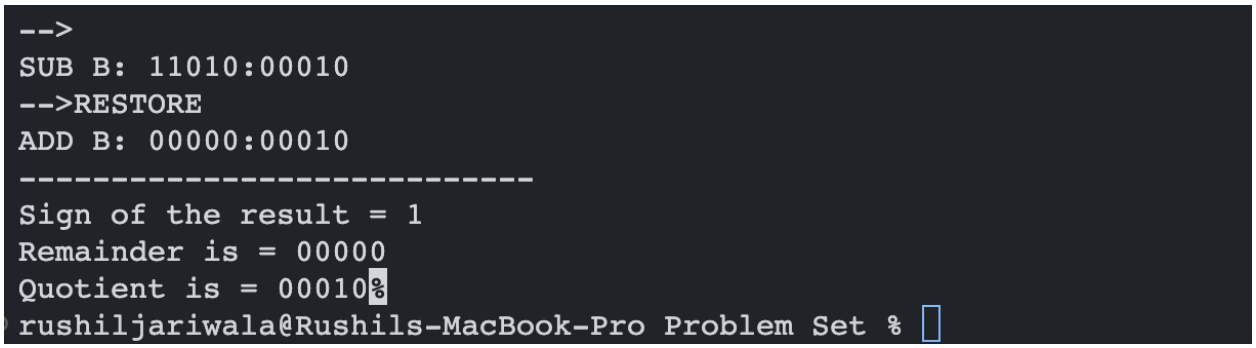
```
    }
    printf(":");
    for (i = 4; i >= 0; i--)
    {
        printf("%d", anumcp[i]);
    }
}
void shl()
{ // for shift left
    int i;
    for (i = 4; i > 0; i--)
    { // shift the remainder
        rem[i] = rem[i - 1];
    }
    rem[0] = anumcp[4];
    for (i = 4; i > 0; i--)
    { // shift the remtient
        anumcp[i] = anumcp[i - 1];
    }
    anumcp[0] = 0;
    printf("\nSHIFT LEFT: "); // display together
    for (i = 4; i >= 0; i--)
    {
        printf("%d", rem[i]);
    }
    printf(":");
    for (i = 4; i >= 0; i--)
    {
        printf("%d", anumcp[i]);
    }
}

void main()
{
    int i;
    printf("\t\tRESTORING DIVISION ALGORITHM");
```

```
printf("\nEnter two numbers to divide: ");
printf("\nBoth must be less than 16");
do
{
    printf("\nEnter A: ");
    scanf("%d", &a);
    printf("Enter B: ");
    scanf("%d", &b);
} while (a >= 16 || b >= 16);

printf("\nExpected Quotient = %d", a / b);
printf("\nExpected Remainder = %d", a % b);
if (a * b < 0)
{
    s = 1;
}

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 < 5; i++)
    {
        printf("\n-->"); // start with subtraction
        printf("\nSUB B: ");
        add(bcomp);
        if (rem[4] == 1)
        { // simply add for restoring
            printf("\n-->RESTORE");
            printf("\nADD B: ");
            anumcp[0] = 0;
            add(bnum);
        }
        else
        {
            anumcp[0] = 1;
        }
        if (i < 4)
            shl();
    }
    printf("\n----------------------------");
    printf("\nSign of the result = %d", s);
    printf("\nRemainder is = ");
    for (i = 4; i >= 0; i--)
    {
        printf("%d", rem[i]);
    }
    printf("\nQuotient is = ");
    for (i = 4; i >= 0; i--)
    {
        printf("%d", anumcp[i]);
    }
    // getch();
}
```

Output


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
RESTORING DIVISION ALGORITHM
Enter two numbers to divide:
Both must be less than 16
Enter A: 12
Enter B: -6
Expected Quotient = -2
Expected Remainder = 0
Unsigned Binary Equivalents are:
A = 01100
B = 00110
B'+ 1 = 11010
-->
SHIFT LEFT: 00000:11000
-->
SUB B: 11010:11000
-->RESTORE
ADD B: 00000:11000
SHIFT LEFT: 00001:10000
-->
SUB B: 11011:10000
-->RESTORE
ADD B: 00001:10000
SHIFT LEFT: 00011:00000
-->
SUB B: 11101:00000
-->RESTORE
ADD B: 00011:00000
SHIFT LEFT: 00110:00000
-->
SUB B: 00000:00000
SHIFT LEFT: 00000:00010


-->
SUB B: 11010:00010
-->RESTORE
ADD B: 00000:00010

Sign of the result = 1
Remainder is = 00000
Quotient is = 00010%
rushiljariwala@Rushils-MacBook-Pro Problem Set %