```
                ******SIGNED MAGNITUDE MULTIPLICATION ALGORITHM******

   Enter two numbers to multiply:
   Both must be less than 16
   Enter b: -2
   Enter Q: 3

    Expected product = -6
   S.C. = 4

    Signed Binary Equivalents are:
    b = 0010
    q = 0011


   S.C. = 4
   -->
    ADD B: 0010 : 0011
    R-SHIFT: 0001:0001
   S.C. = 3
   -->
    ADD B: 0011 : 0001
    R-SHIFT: 0001:1000
   S.C. = 2
   -->
    R-SHIFT: 0000:1100
   S.C. = 1
   -->
    R-SHIFT: 0000:0110
    product is = 100000110
   PS C:\Users\Roshan\Desktop\CA>
```

```
                         ******BOOTH'S MULTIPLICATION

   Enter two numbers to multiply:
   Both must be less than 16
   Enter A: 2
   Enter B: 3

   Expected product = 6

   Binary Equivalents are:
   A = 00010
   B = 00011
   B'+ 1 = 11101


   -->
   AR-SHIFT: 00000 : 00001
   -->
   SUB B: 11101 : 00001
   AR-SHIFT: 11110 : 10000
   -->
   ADD B: 00001 : 10000
   AR-SHIFT: 00000 : 11000
   -->
   AR-SHIFT: 00000 : 01100
   -->
   AR-SHIFT: 00000 : 00110
   Product: 0000000110
   PS C:\Users\Roshan\Desktop\CA>
```

```
             ***** Non-Restoring Division Algorithm *****
   Enter dividend (0-31): 14
   Enter divisor (1-15): 3

   Binary Representation:
   Dividend: 01110
   Divisor: 00011
   2's Complement of Divisor: 11101

   Stepwise Execution:
   Shift Left: 00000 | 11100
   Subtract B: 11101
   Quotient Updated: 11101 | 11100
   Shift Left: 11011 | 11000
   Add B: 11110
   Quotient Updated: 11110 | 11000
   Shift Left: 11101 | 10000
   Add B: 00000
   Quotient Updated: 00000 | 10001
   Shift Left: 00001 | 00010
   Subtract B: 11110
   Quotient Updated: 11110 | 00010
   Shift Left: 11100 | 00100
   Add B: 11111
   Quotient Updated: 11111 | 00100
   Final Adjustment (Add B): 00010

   Final Results:
   Quotient: 00100
   Remainder: 00010

   Expected Quotient: 4
   Expected Remainder: 2
   PS C:\Users\Roshan\Desktop\CA>
```

```
PS C:\Users\Roshan\Desktop\CA> cd "c:\Users\Roshan\Desktop\CA\" ; if ($?) {
-lole32 } ; if ($?) { .\LAB1 }
                *****1's and 2's COMPLEMENT******

Enter a valid binary number: 100
The 1s complement of the entered number is: 011
The 2s complement of the entered number is: 100
PS C:\Users\Roshan\Desktop\CA>
```

```
              ***** Restoring Division Algorithm *****
   Enter dividend (0-31): 14
   Enter divisor (1-15): 3

   Binary Representation:
   Dividend: 01110
   Divisor: 00011
   2's Complement of Divisor: 11101

   Stepwise Execution:
   Shift Left: 00000 | 11100
   Subtract B: 11101
   Restore Remainder: 00000 | 11100
   Shift Left: 00001 | 11000
   Subtract B: 11110
   Restore Remainder: 00001 | 11000
   Shift Left: 00011 | 10000
   Subtract B: 00000
   Quotient Updated: 00000 | 10001
   Shift Left: 00001 | 00010
   Subtract B: 11110
   Restore Remainder: 00001 | 00010
   Shift Left: 00010 | 00100
   Subtract B: 11111
   Restore Remainder: 00010 | 00100

   Final Results:
   Quotient: 00100
   Remainder: 00010

   Expected Quotient: 4
   Expected Remainder: 2
   PS C:\Users\Roshan\Desktop\CA>
```