WARM UP LESSONS – MORE ON BITS OPERATION

CONTENTS

Big-endian vs Little-Endian 1
About Parity bit 2
About Floating point Representation 2
Exercises: 3
Hint for division: 4

BIG-ENDIAN VS LITTLE-ENDIAN

Take the following sample array:

Int32 X[2] ;

X[0] = 0x01020304

X[2] = 0x05060708

Big-endian: The following shows how a big-endian system would organize the data.

Value	01	02	03	04	05	06	07	08
Memory Addr:	F1	F2	F3	F4	F5	F6	F7	F8

Notice that each byte increases from most significant to least significant.

Little-endian: The following shows how a big-endian system would organize the data.

Value	04	03	02	01	08	07	06	05
Memory Addr:	F1	F2	F3	F4	F5	F6	F7	F8

Notice that each byte increases from most significant to least significant.

But no worry there if you were to do:

```
printf("%x %x", X[0] and X[2]) ;

You will see get value of  01020304  and  05060708.
```

Beware!

It will be interpreted incorrectly if the data is saved on a little-endian host and then being interpreted on a big-endian host, and vice versa.

About Parity bit

This is a commonly used in error detection and cryptography.

When data is transmitted from one device to another, you should always check the parity bit as part of the error checking.

Parity of a number can be odd or even. If the number contains odd number of 1-bits, it is a odd-parity; otherwise, "even parity".:

Bitwise Operation

Read the book about bit-wise operation.

About Floating point Representation

Coming Soon.

EXERCISES:

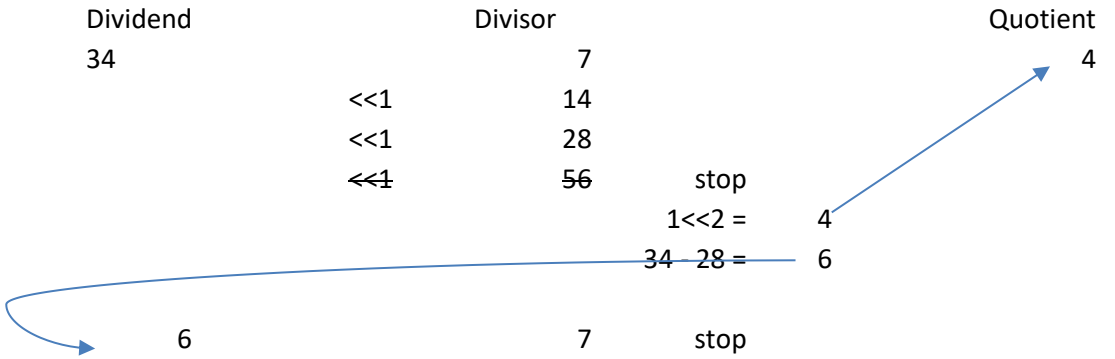
1) Write a program to ask user for a number, and you will display it's value in base-10, base-16, and base-2, and its parity bit.

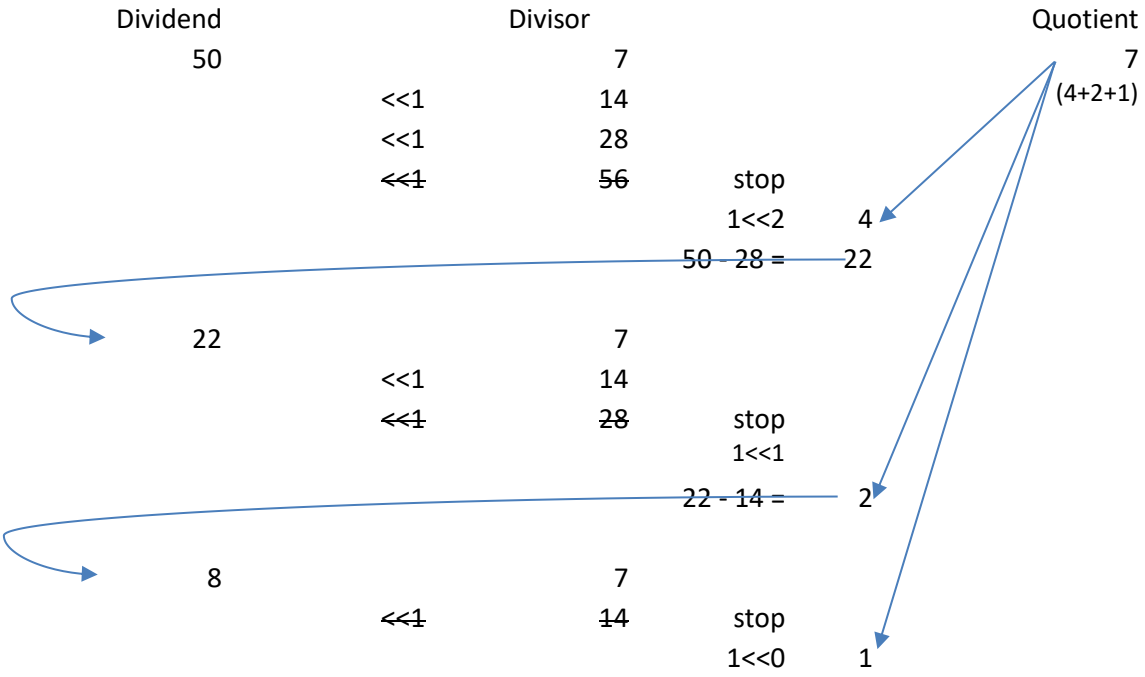
   Examples for your output:

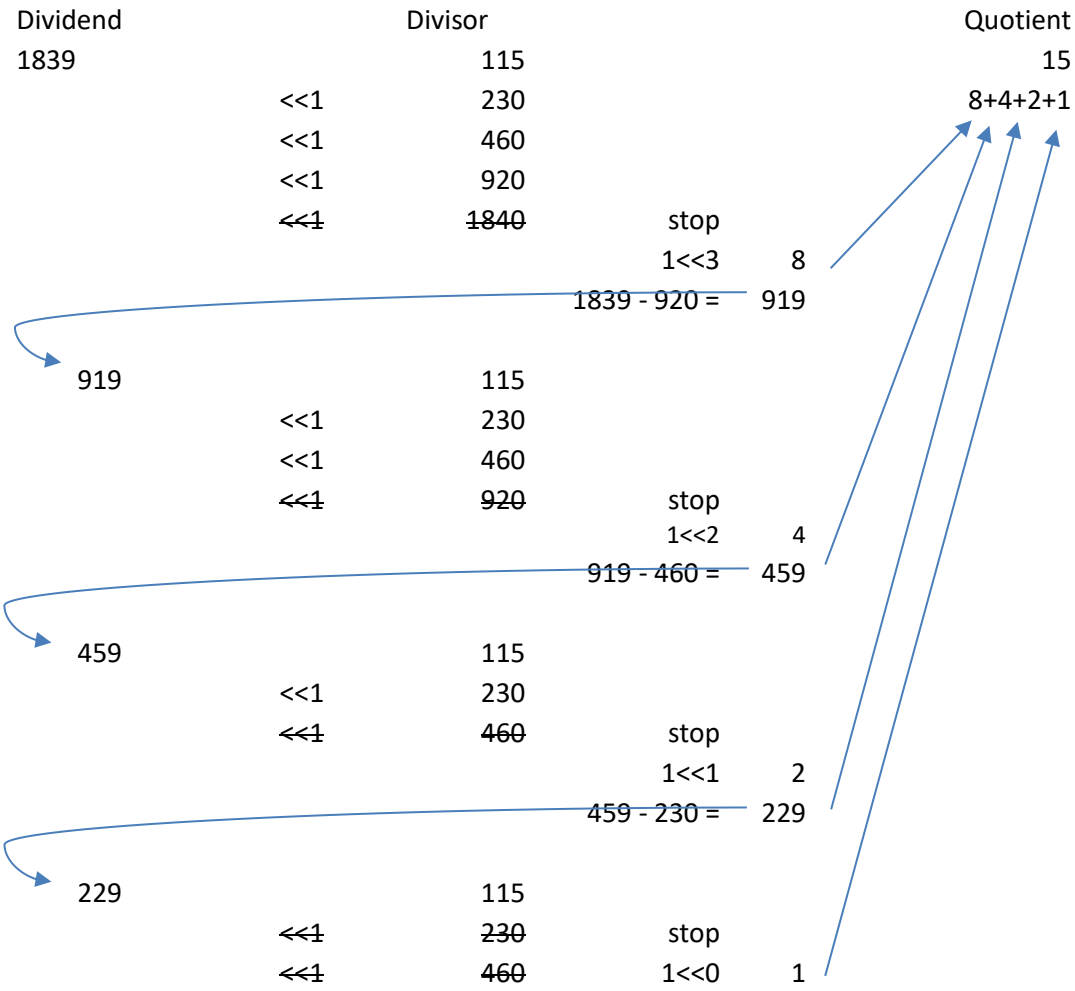
   Sample 1: Num = 13 = D (Hex) = 1101 (Binary). Odd Parity
   Sample 2: Num = 8 = 8 (Hex) = 1000 (Binary). Even Parity

2) Write 4 functions to perform +, -, x, /, % with using ONLY bitwise operations.
3) Write a swap function without creating a separate variables to hold the temp.
4) Write abs(int) without using +, -, x, /, %
5) Write abs(float) without using +, -, x, /, % (This needs you to understand the structure of a floating point number) . Skip this if you have yet covered this.

HINT FOR DIVISION:






Dividend
Divisor
Quotient
1839
115
15
<<1
230
8+4+2+1
<<1
460
<<1
920
~~<<1~~
~~1840~~
stop
1<<3
8
1839 - 920 =
919
919
115
<<1
230
<<1
460
~~<<1~~
~~920~~
stop
1<<2
4
919 - 460 =
459
459
115
<<1
230
~~<<1~~
~~460~~
stop
1<<1
2
459 - 230 =
229
229
115
~~<<1~~
~~230~~
stop
~~<<1~~
~~460~~
1<<0
1