**Siena College’s 32nd Annual** **High School Programming Contest**

**Sponsored by Transfinder**

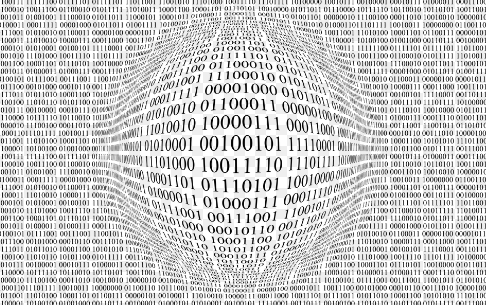
##### **March 29, 2019**

###### Green Problem #3:  A Quick Change (of Base)

Background Information:  All computer scientists know how crucial binary (base 2) numbers and positional notation in general are for designing and implementing computer architectures.

For positional notation in base 10 the integer 234 = 2\*100 + 3\*10 + 4\*1 = 2\*102 + 3\*101 + 4\*100. The position of the “2” means that it represents 200.

For positional notation in base 2 the integer 11001 = 1\*24 + 1\*23 + 0\*22 + 0\*21 +1\*20. This is equal to 25 in base 10

There is a nice algorithm for converting a base 10 number to its equivalent base 2 representation. The algorithm uses repeated division by 2 until reaching a quotient of 0. The remainders from the division are used to build the equivalent binary representation of the number. Here is the sequence of divisions for converting 77 base 10 to 1001101 base 2.

77 / 2 = 38 with a remainder of **1**

38 / 2 = 19 with a remainder of **0**

19 / 2 = 9 with a remainder of **1**

9 / 2 = 4 with a remainder of **1**

4 / 2 = 2 with a remainder of **0**

2 / 2 = 1 with a remainder of **0** and

1 / 2 = 0 with a remainder of **1**.

The remainders can be used (from bottom up) to build the binary equivalent of the starting base 10 value.

###### Programming Problem:

Input:   N, a positive integer with N < 1,000

Output:  The list of quotients and remainders from the number base conversion algorithm described above with one pair per line followed by the final binary value for N.

###### Example 1: Input 77 Example 2: Input:  29

###### Output: 38 1 Output:  14 1

19 0 7 0

9 1 3 1

4 1 1 1

2 0 0 1

1 0 11101

0 1

1001101

V1