Bits and Bytes

Pre-requisites:

Knowledge of

- bitwise operators &, |, >>, <<, ^, ~
- Data representation of signed and unsigned integers

Q1. BitAnd

```
/*
/*  bitAnd - x & y using only ~ and |
/* Example bitAnd(6, 5) = 4
*/ Legal ops:  ~ |
/*Max ops: 8
*/
int bitAnd (int x, int y) {
     return 0;
}
```

Q2. BitXor

```
/*
/*  bitXor - x ^ y using only ~ and &
/* Example bitXor(4, 5) = 1
*/ Legal ops:  ~ &
/* Max ops: 14
*/
int bitXor (int x, int y) {
     return 0;
}
```

Q3. Sign

```
/*
* sign - return 1 if positive, 0 if zero, and -1 if negative
* Examples: sign(130) = 1, sign(-23) = -1
* Legal ops: ! ~ & ^ | + << >>
* Max ops : 10
*/
int sign(int x) {
   //TODO
    return 0;
}
```

Q4. GetByte

```
/*
* getByte - extract byte n from word x
* Examples: getByte(0x12345678, 1)  = 0x56
*Legal ops: ! ~ & ^| + << >>
Max ops: 6
*/
int getByte(int x, int n) {
    return 0;
}
```

Q5. LogicalShift

```
/* logicalShift - shift x to the right by n, using a logical shift
*  can assume that 0 <=n<=31
* examples: logicalShift(0x87654321, 4) = 0x8765432
*  Legal ops:  ~ & ^ | + << >>
*Max ops: 20
*/
int logicalShift(int x, int n) {
    return 0;
}
```

Q6. Conditional

```
/*
* conditional - same as x ? y : z
* example conditional (2, 4, 5) = 4
* ! ~ & ^ | + << >>
* Max ops: 16
*/
int conditional(int x, int y, int z) {
    return 0;
}
```

Q7. Bang

```c
/* bang - Compute !x without using !
*  Examples: bang(3)=0, bang(0)=1
*  Legal ops: ~ & ^ | + << >>
*  Max ops: 12
*/
int bang(int x) {
    return 0;
}
```

Q8. Invert

```c
/*
/* Return x with the n bits that begin at position p inverted (i.e. turn 0
/* into 1 and vice versa)
/* and the rest left unchanged. Consider the indices of x to begin with the
/* lower -order bit   numbered
/* Legal ops: ~ & ^ | << >>
/* as zero
*/
int invert (int x, int p, int n) {
    //TODO
    return 0;
}
```

Q9. Base64 Encoding

Please solve this hackerrank problem

Submission

This assignment has to be strictly submitted in C Language. Problems 1-8 can be implemented in one file and problem 9 in a separate file. As always, please share github links to your solutions.