```
ascii_to_hex:

'I' == 0x49
'5' == 0x35
'd' == 0x64

binary_to_hex1:

11111111 11111111 11111111 11111111    0xffffffff
00000010 00000000 10000000 00000000    0x02008000
00000000 00000000 00011111 11100000    0x00001fe0
11111000 01111111 00000000 00000000    0xf87f0000

bitmask0:

test if mine has lowest bit on: (mine & 1) != 0
set lowest bit in yours: yours = yours | 1
clear lowest bit in yours: yours = yours & ~1
toggle lowest bit in yours: yours = yours ^ 1
// The following are treating the bits as a set.
// If bit at index i is 1 then i is in the set.
union mine with yours: mine = mine | yours
intersect mine with yours: mine = mine & yours
remove yours from mine: mine = mine & ~yours
is yours a subset of mine?: (yours & mine) == yours

bitmask1:

test if mine has either of two lowest bits on: (mine & 0x3) != 0
test if mine has both of two lowest bits on: (mine & 0x3) == 0x3
set lowest 8 bits of mine: mine |= 0xff
clear every other bit in mine: mine &= 0x55555555

bitmask2:

all bits on: ~0
one bit on in position n, all others off: 1 << n
n least significant bits on, all others off: (1 << n) - 1
most significant bit on, all others off: (1 << 31)
k most significant bits on, all others off: (~0 << (32 - k))

bitmask3:

1 << x: 2 to the x power
~x + 1: -x, arithmetic negation
x >> 31: -1 if x was negative, 0 otherwise
x &= (x - 1): clears lowest "on" bit in x
```

```
bitset1:

bitset(22, 5)       54
bitset(15, 31)      -2147483633
bitset(12, 0)       13
bitclear(54, 5)     22
bitclear(15, 31)    15
bitclear(-12, 31)   2147483636

bitwise1:

this        == 11110000
that        == 01010101
this & that == 01010000
this | that == 11110101
this ^ that == 10100101
~this       == 00001111
this >> 2   == 00111100
that << 1   == 10101010

hex_to_binary1:

0x1       00000000 00000000 00000000 00000001
0x1ff     00000000 00000000 00000001 11111111
0x800000  00000000 10000000 00000000 00000000
0xa017    00000000 00000000 10100000 00010111
```