```
/* bit_tools.cpp

Authors: Cyrus Haywoon, Julissa Hernandez, Nathaniel Levin, Sheel Shah,
         Joseph Sparks, Edward Welch, and Jeffrey Elkner


A collection of functions useful for writing and debugging functions using bitwise operators.
char* byte2bitstr(unsigned char byte) takes an unsigned char (byte) as an argument and returns
a string of 8 characters '0' or '1' representing the bits in the byte. unsigned char
invert(byte, p, n) takes a byte (unsigned char) as an argument and returns a new byte with the
n bits beginning at position p of the argument inverted (1s become 0s and 0s become 1s).
unsigned char rightrot(unsigned char byte, char n) rotates n bits of byte to the right.
Example: 42 (000101010) rotated by 3 bits will yield 010000101 (69). unsigned char
setbits(unsigned char byte, char p, char n, unsigned char byte2) sets the n bits of byte
beginning at position p to the lower order n bits of byte2.
*/

#include <stdio.h>
#include <stdlib.h>

char* byte2bitstr(unsigned char byte) {
  char* binstr = (char*)malloc(9);

  for (int i = 7; i >= 0; i--) {
    binstr[7 - i] = ((byte >> i) & 1) ? '1' : '0';
  }
  binstr[8] = '\0';

  return binstr;
}

unsigned char invert(unsigned char byte, char p, char n) {
  return byte ^ (~(0xFF << n) << (p + 1 - n));
}

unsigned char rightrot(unsigned char byte, char n)
{
  while (n-- > 0) {
    byte = (byte & 1) ? (byte >> 1) | 0x80 : byte >> 1;
  }
  return byte;
}

char bitcount(unsigned char byte) {
    char bcount;

    // count each 1 in the input (binary)
    for (bcount = 0; byte != 0; byte &= (byte-1))
        bcount++;

    return bcount;
}

unsigned char setbits(unsigned char byte, char p, char n, unsigned char byte2) {
  return byte & ~(~(0xFF << n) << (p + 1 - n)) |
         ((byte2 & ~(0xFF << n)) << (p + 1 - n));
}
```