```
.global uint32ToBinary
.global bro32
.global countOnes32

.text

// void uint32ToBinary(char str[], uint32_t x)
// address of string in R0, value in R1
uint32ToBinary:
    MOV R2, #0x80000000 // mask with bit 31 set
utb_loop:
    TST R1, R2 // return 0 if bit not set, non-zero if bit set
    MOVNE R3, #'1' // bit set
    MOVEQ R3, #'0' // bit clear
                   // if bitset char c='1', if bitclear c='0'
/*
    STRB R3, [R0]  // store ascii character in string
                   // *str = c
    ADD R0, R0, #1 // increment pointer by sizeof(char)
                   // str++
*/
    STRB R3, [R0], #1 // store ascii character in string,
                      // inc R0 by sizeof(char)=1
                      // *(str++) = c
    MOVS R2, R2, LSR #1 // move mask bit to the right
    BNE utb_loop // loop back for remaining 31 bits
    MOV R3, #0 // add null terminator
    STRB R3, [R0]
    BX LR

// uint32_t bro32(uint32_t x)
// value in R0, return bro(value) in R0
bro32:
    MOV R1, R0 // move original value to R1
    MOV R0, #0 // zero result
    MOV R2, #0x80000000 // test mask with bit 31 set
    MOV R3, #0x00000001 // apply mask with bit 0 set
bro_loop:
    TST R1, R2 // return 0 if bit not set, non-zero if bit set
    ORRNE R0, R0, R3 // if bit in R1 set, set bit in R0
    MOVS R2, R2, LSR #1 // move mask bit to the right
    MOV R3, R3, LSL #1  // move mask bit to the left
    BNE bro_loop // loop back for remaining 31 bits
    BX LR

// uint32_t countOnes32(uint32_t x)
// value in R0, number of 1's on value returned in R0
countOnes32:
    MOV R1, R0 // store number in R1
    MOV R0, #0 // zero count
    MOV R2, #0x80000000 // mask with bit 31 set
co_loop:
    TST R1, R2 // return 0 if bit not set, non-zero if bit set
    ADDNE R0, R0, #1 // increment count if bit is not zero
    MOVS R2, R2, LSR #1 // move mask bit to the right
    BNE co_loop // loop back for remaining 31 bits
    BX LR
```