```c
#include <stdio.h>

int main(){
    // Convert a decimal number to binary format
    int i, j, a = 101;

    // We will solve this problem two ways
    // Make sure we require less than 100 binary digits to store our numbers
    int digitsLSB[100];
    int digitsMSB[100];
    int aCopy = a; // We will destroy a so make a copy

    // The case of a = 0 can be handled as a special case
    // but for a < 0 we need to learn more things
    if(a <= 0){
        printf("Enter a stricly positive number");
        return 0;
    }

    // The least significant bit LSB is the rightmost bit
    // MSB is the most significant bit, the left most bit

    /**** METHOD 1 ****/
    // Calculate the bits from LSB to MSB
    i = 0;
    while(a){
        // The LSB of a is simply given by whether a is even or not
        digitsLSB[i++] = a % 2;
        a /= 2; // Remove the LSB and move on, repeating the process
    }

    // Print the binary representation of a MSB to LSB
    // Remember, a has been destroyed - use aCopy
    printf("The binary representation of %d is ", aCopy);
    for(j = i-1; j >= 0; j--)
        printf("%d",digitsLSB[j]);
    printf("\n");

    /**** METHOD 2 ****/
    // Calculate the bits from MSB to LSB
    //The MSB of a is found out from the largest power of 2 less than a
    i = 0;
    int pow2 = 1;
    // Remember, a has been destroyed - use aCopy
    while(pow2 <= aCopy){
        pow2 *= 2;
        i++;
    }
    i--; // We went one too far
    pow2 /= 2;

    printf("The binary representation of %d is ", aCopy);
    // Print the digits MSB to LSB
    // Remember, need to go to zero power for LSB
    while(i >= 0){
        if(pow2 <= aCopy){
            printf("1");
            aCopy -= pow2;
        }else
            printf("0");
        pow2 /= 2;
        i--;
    }

    return 0;
}
```