```c
#include <stdio.h>
#include <string.h>

int main(){
    // This problem can be solved without pointers as well
    // However, careful and responsible use of pointers makes life easy
    char str[1000], mod[1000], substr[100], firstCharUpr, firstCharLwr;
    char *ptrCurr, *ptrNext;
    int i, idxCurr, idxNext, count = 0;

    gets(str);
    gets(substr);

    if('a' <= str[0] && str[0] <= 'z'){
        firstCharUpr = str[0] - 'a' + 'A';
        firstCharLwr = str[0];
    }else{
        firstCharUpr = str[0];
        firstCharLwr = str[0] - 'A' + 'a';
    }

    for(i = 0; i < strlen(str); i++)
        if(str[i] == firstCharUpr || str[i] == firstCharLwr)
            count++;

    printf("%d\n", count);

    ptrCurr = str;
    idxCurr = 0;
    i = 0; // Counter for modified string

    while(strlen(ptrCurr) > 0){
        ptrNext = strstr(ptrCurr,substr);
        // If no more occurrences of the sensitive string
        // simply print the original string characters
        if(!ptrNext){
            for(; idxCurr < strlen(str); idxCurr++)
                mod[i++] = str[idxCurr];
            break;
        }
        else{
            // Find index of the next occurrence of sensitive string
            idxNext = idxCurr + ptrNext - ptrCurr;
            // Print the original string till the sensitive string
            for(; idxCurr < idxNext; idxCurr++)
                mod[i++] = str[idxCurr];
            // Instead of the sensitive string, print XXX
            mod[i++] = 'X';
            mod[i++] = 'X';
            mod[i++] = 'X';
            ptrCurr = ptrNext + 2;
            // Jump over the sensitive string
            idxCurr = idxNext + strlen(substr);
        }
    }
    mod[i] = '\0'; // Dont forget the NULL character
    printf("%s", mod);
    return 0;
}
```