```c
/*
 * João Paulo Batista Ferreira
 * 2009113274
 * Algoritmos e Estruturas de Dados - TP3 exA
*/

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

typedef struct item{
    char word[256];
    unsigned int counter;
}item;

typedef struct node {
    item info;
    struct node *right;
    struct node *left;
} node;

typedef node *nodePtr;

void eraseTree(nodePtr leaf)
{
    if (leaf == NULL)
        return;
    eraseTree(leaf->left);
    eraseTree(leaf->right);
    free(leaf);
}

void visit (nodePtr leaf)
{
    printf("%s %d\n", leaf->info.word, leaf->info.counter);
}

void emOrdem (nodePtr leaf)
{
    if (leaf != NULL)
    {
        emOrdem(leaf->left);
        visit (leaf);
        emOrdem(leaf->right);
    }
}

nodePtr insert (nodePtr leaf, char* word)
{

    if (leaf == NULL)
    {
        leaf = (nodePtr) malloc (sizeof(node));

        strcpy(leaf->info.word, word);
        leaf->info.counter = 1;
        leaf->left = NULL;
        leaf->right = NULL;

        return leaf;
    }

    if (strcmp(word,leaf->info.word ) < 0)
    {
        t++;
        leaf->left = insert (leaf->left, word);
    }

    else if (strcmp(word,leaf->info.word) > 0)
    {
        t++;
        leaf->right = insert (leaf->right, word);
    }

    else
        leaf->info.counter++;

    return leaf;
}

nodePtr worker(char *word, nodePtr leaf)
{
    int i, size;

    size = strlen(word);

    for (i=0 ; i<size ; i++)
        word[i]=tolower(word[i]);

    leaf = insert(leaf, word);

    return leaf;
}

int main(int argc, char **argv)
{
    char word[256];
    nodePtr tree=NULL;

    while( (scanf("%s", word)) != EOF )
        tree = worker (word, tree);

    emOrdem(tree);

    eraseTree(tree);

    return 0;
}
```