Задача 1.

Задачу можно решить используя стандартные алгоритмы из стандартной библиотеки. Каждое слово необходимо привести к нижнему регистру, потом отсортировать (для следующей функции), а после посчитать симметрическую разность. Если получившаяся разность содержит не более 1 элемента, то слова похожи

```
#include <iostream>
#include <algorithm>
#include <cctype>

int main() {
    std::string s;
    std::cin >> s;
    std::for_each(s.begin(), s.end(), [](char& x){ x = tolower(x);});
    std::sort(s.begin(), s.end());
    int N;
    std::cin >> N;
    int result = 0;
    for(int i = 0; i < N; i++) {
        std::string curr;
        std::string delta;
        std::cin >> curr;

        std::for_each(curr.begin(), curr.end(), [](char& x){ x = tolower(x);});
        std::sort(curr.begin(), curr.end());
        std::set_symmetric_difference(curr.begin(), curr.end(), s.begin(),
s.end(), std::back_inserter(delta));

        if(delta.size() <= 1) {
            result += 1;
        }
    }
    std::cout << result << std::endl;
    return 0;
}
```

Задача 2.

Заметим, что в паре с минимальным элементом, должен стоять максимальный элемент. Продолжая рассуждать в таком же ключе, становится понятно, что если отсортировать числа по возрастанию, то парами будут первый и последний, второй и предпоследний и тд.

```
#include <iostream>
#include <algorithm>
#include <cctype>

int main() {
    int N;
    std::cin >> N;
    std::vector<int> nums(N);
    std::vector<std::pair<int, int> > result(N/2);
    for(int i = 0; i < N; i++) {
        std::cin >> nums[i];
    }

    std::sort(nums.begin(), nums.end());

    auto iter = nums.begin();
    std::advance(iter, N/2);
    std::rotate(nums.begin(), iter, nums.end());

    iter = nums.begin();
    std::advance(iter, N/2);
    std::reverse(nums.begin(), iter);

    iter = nums.begin();
    std::advance(iter, N/2);
    std::transform(nums.begin(), iter, iter, result.begin(), [](int a, int b)
{return std::make_pair(a, b);});

    for(auto it = result.begin(); it != result.end(); it++) {
        std::cout << (*it).first << ' ' << (*it).second << std::endl;
    }

    return 0;
}
```