**Shortest direction**

[string](http://www.practice.geeksforgeeks.org/tag-page.php?tag=string&isCmp=0)

[Flipkart](http://www.practice.geeksforgeeks.org/tag-page.php?tag=Flipkart&isCmp=1)

A person wants to go from origin to a particular location, he can move in only 4 directions(i.e East, West, North, South) but his friend gave him a long route, help a person to find minimum Moves so that he can reach to the destination.

**Note:** You need to print the lexicographically sorted string. Assume the string will have only ‘E’ ‘N’ ‘S’ ‘W’ characters.

**Input:**

The first line of input contains an integer T denoting the number of test cases.  
The first line of each test case is S, S is the long route (string).

**Output:**

Print minimum Moves so that he can reach to the destination.

**Constraints:**

1 ≤ T ≤ 100  
1 ≤ S length ≤ 1000

**Example:**

**Input**  
2  
SSSNEEEW  
NESNWES

**Output**  
EESS  
E

\*\*For More Examples Use Expected Output\*\*

<http://www.practice.geeksforgeeks.org/problem-page.php?pid=821>

#include <iostream>

#include <stdio.h>

#include <math.h>

#include <vector>

#include <map>

#define ll long long int

using namespace std;

int main() {

int t;

scanf("%d", &t);

while(t--) {

std::string s;

cin >> s;

std::map<char, int> m;

for(int i =0; i<s.size(); i++) {

m[s[i]]++;

}

string ans = "";

if(m['N'] > m['S']) {

for(int i =0; i < m['N'] - m['S']; i++) {

ans += 'N';

}

}else if(m['N'] < m['S']) {

for(int i =0; i < m['S'] - m['N']; i++) {

ans += 'S';

}

}

if(m['E'] > m['W']) {

for(int i =0; i < m['E'] - m['W']; i++) {

ans += 'E';

}

}else if(m['E'] < m['W']) {

for(int i =0; i < m['W'] - m['E']; i++) {

ans += 'W';

}

}

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

cout << ans << endl;

}

return 0;

}

-------------------------EN C#------------------------------------------------

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication1

{

class Program

{

static void Main(string[] args)

{

string s = "SSSNEEEW";

string ans = "";

int N = s.Count(e => e == 'N');

int S = s.Count(e => e == 'S');

int E = s.Count(e => e == 'E');

int W = s.Count(e => e == 'W');

if (N > S)

{

ans += new string('N', N - S);

}

else if (S > N)

{

ans += new string('S', S - N);

}

if (E > W)

{

ans += new string('E', E - W);

}

else if (W > E)

{

ans += new string('W', W - E);

}

char[] sort = ans.ToCharArray();

Array.Sort(sort);

ans = new string(sort);

Console.WriteLine(ans);

// }

Console.ReadLine();

}

}

}