There are 4 problems. You can assume that the input to your program conforms to the description given in each problem, so you do not need to check for valid input. The output of each program should be a single boolean value.

1. (Counting Votes) Suppose that there are two candidates running in an election, call them “a” and “b”. Write a program that takes the sequence of votes cast and determines if the winner always leads in the vote count. The input is a character string representing the votes. For example, the input

   babaabbababba

indicates that the first vote was for candidate “b”, the second for candidate “a”, and so on. In this case, candidate “b” eventually wins with 7 votes to 6, but candidate “a” leads after five votes. Your program should print out true if one candidate always leads (or ties) and false otherwise. Put another way: print true if one candidate never trails the other, and false otherwise.

2. (Same Letters) Write a program that reads in two strings and prints true if exactly the same set of letters appears in both strings, and false if one of the strings contains a letter that the other does not contain. You may assume that the input strings are made up only of lower case letters. For example, if the input is

   lion
   zebra

the output should be false, while for the input

   rats
   strata

the output should be true.
3. (Word Search) The input is a first line containing a positive integer \( n \), a second line containing a single word, and then \( n \) subsequent lines, each containing a string of \( n \) lower case letters. The goal is to determine if the input word appears in any of the rows or columns of the matrix of characters formed by the \( n \) input strings. For simplicity, consider only forward and down directions. For example, for the input

\[
\begin{array}{ccccccc}
 7 \\
cat \\
oefkewp \\
kldscne \\
rhdljcb \\
lyupcar \\
uwfhe \\
jryieun \\
etbmdsk \\
\end{array}
\]

the output should be true, since “cat” appears starting in the third row and sixth column and reading down.

4. (Divide Weight Evenly) Suppose that you have a collection of \( n \) items with varying weights, and you want to divide the items into two sets of equal weight. The input to the program is a first line containing a positive integer \( n \), and a second line containing \( n \) positive integers. The output should be true if the items can be split into two groups of equal weight and false otherwise. For example, if the input is

\[
\begin{array}{ccccccc}
 3 \\
10 & 2 & 6 \\
\end{array}
\]

then the output should be false since the total weight is 18, and no subset of the items has combined weight \( 18/2 = 9 \). If the input is

\[
\begin{array}{ccccccc}
 8 \\
6 & 4 & 9 & 1 & 7 & 3 & 12 & 2 \\
\end{array}
\]

then the output should be true, since the total weight is 44, and the combined weight of the third, fourth, and seventh items is \( 9 + 1 + 12 = 22 \).

You can assume that \( n < 20 \).