Activision CTN Programming Test

# Objective

Implement a program to find all possible words of two letters or more created by a series of N lettered wheels with M letters per wheel, like a combination lock.



* Consider how your program will perform for large values of N and M.
* Words must be formed from a continuous forward run of letters, can start at any wheel, but cannot “wrap”. The maximum word length will therefore be M.
* In the above example ‘AT’, ‘MAT’ and ‘MATCH’ are all valid words. ‘CAT’ is not valid, as it is not in order.
* If we incremented each wheel by one letter, we can see ‘NO’ would be a valid word. ‘ON’ would not be valid, as it is backwards.
* There is no need to remove duplicate words and the order of the output word list does not matter.

# Data Format

The lettered wheels are represented in a plain text file wheels.txt. The first two lines define the number of wheels and the number of letters per wheel. Each following line describes the letters in one wheel. E.g.

5

4

ETMJ

NTAO

RNTC

TOCS

AEHS

The dictionary file is a plain text file dictionary.txt which contains an unsorted list of valid words with one word per line:

AARDVARK

AARDVARKS

AT

MATCH

CAT

DOG

JARS

JAR

ON

NO

ZEBRA

# Submission

The program should be written in C++ to run on Windows on the command line, taking *wheels.txt* and *dictionary.txt* as arguments.

The program should output the found words to stdout, followed by the total number of words found.

e.g.

C:\Test>find\_words.exe wheels.txt dictionary.txt

C:\Test>MAT

C:\Test>MATCH

C:\Test>AT

C:\Test>NO

C:\Test>JAR

C:\Test>JARS

C:\Test>Found 6 words

Please supply source code and a compiled executable. The code should be of quality you would expect to submit as a professional, with appropriate error handling and comments.

Enclosed are a provided *wheels.txt, dictionary.txt* *and output.txt*. Your program should work with arbitrary data, but please check that your program produces the same results as output.txt when using wheels.txt and dictionary.txt as inputs.