The aim of the word finding game is to find and mark the hidden words in certain letters placed in a square or rectangle. Words can be arranged horizontally, vertically or diagonally. A simple game example is shown in the following figure

A	Y	D	G	A	R	T	U
R	E	I	A	A	M	N	K
Y	R	R	L	A	R	T	İ
A	S	D	S	M	R	T	G
A	M	A	A	F	A	E	V
R	R	E	E	S	D	Z	E
A	S	P	I	N	A	R	S
E	V	R	E	M	G	A	C

- ARAS
- ARYA
- MERVE
- PINAR
- SEVGİ
- YILMAZ

In this lab, you are expected to create a word finding game in C programming language. First, you must create a two-dimensional character string of 15 x 15 dimensions and get the values of this array from the user. This will be your square grid where you will search for words. You should then take a word from the user and search for that word in your two-dimensional array. If your program finds the word in the string, the word should be printed as it is, the other characters should be printed as "*". If your word is not in your two dimensional array, all characters will be printed with the "*" character.

input1

```
AYDGARTUAAHILAL
REIAAMNKLTEMHEM
YRRLARTBISDSGAS
ASDSMRTUYUASOAI
AMSAFAELTRCSZZG
RREESDZEAGIADIV
ASUUCANNFTNRERE
ARYEZGATATAPMUS
UTSARMESEARRAGA
AAAZAYHANASVUUY
SEASDAAPOLYYAMS
```

output1.txt

```
***************
***************
***************
***************
***************
***************
***************
***************
***************
***************
***************
```

****

F U R K A E V R E M K L A T O

****

E R T K F U R K A N N N U Y L

****

Z E H R A M M E R Y E M R F M

****

P I N A R N R S U L U T R U K

PINAR********

PINAR

Name your work as StudentID.c and upload it to the system. Make sure that your program is running in the Ubuntu environment. For the correct output format, carefully review the sample input and output files provided. You can perform the following operations to check the accuracy of your program.

1. Compile your program using the gcc command.
2. Using the ./a.out <input1> myOutput1.txt command, run your program with the input1.txt file and save your output to myOutput1.txt file.
3. Automatically compare the true output and your output using the diff myOutput1.txt output1.txt command. If there is no warning on the command prompt after entering this command, this means that your program is working correctly for these values. If you see a warning, this indicates a problem with your output.

4. Try commands in items 2 and 3 for other input files given to you.

5. Test your program for different inputs you will create yourself. Note that the input files given to you and the input files used during the evaluation may differ from each other.