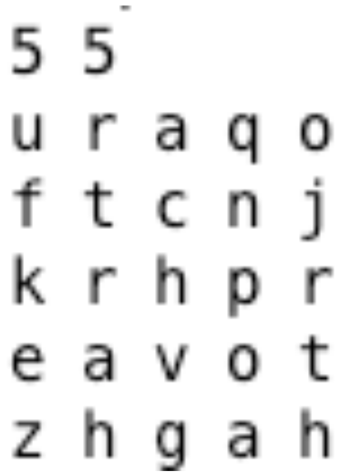
CSC 220 – Project #3 –Word Search

Due: Sunday, April 26th, before midnight

Due Date:
11:59pm, April 26th, 2015

Project Details: (Paired programming assignment, group max 3)

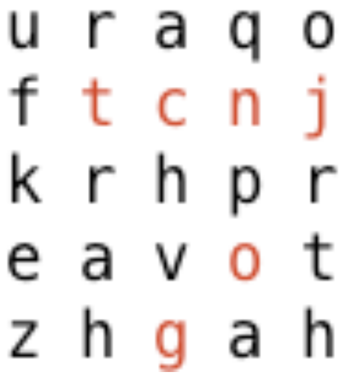
Write a program to search words in a matrix. This program will find the words listed in command line and highlight the words with color. On Canvas, there are several matrices for testing. For example, inside 0505matrix.txt file, you will read:



```
5 5
u r a q o
f t c n j
k r h p r
e a v o t
z h g a h
```

Here, the first 5 is the number of rows, the second 5 is the number of columns. If the JAVA program name is *findwords* and we want to search tcnj and go in this matrix. We can type the following command

```
%./findwords tcnj go < 0505matrix.txt
```

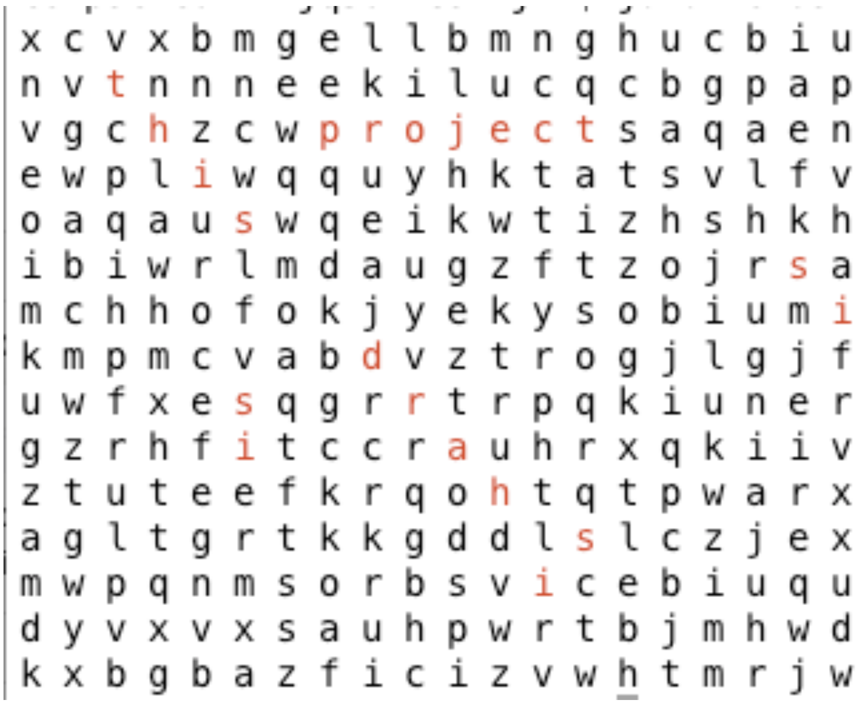


```
u r a q o
f t c n j
k r h p r
e a v o t
z h g a h
```

As you can see, tcnj and go are red colored.

If you want to search this project is hard inside file 1520matrix. You can type:

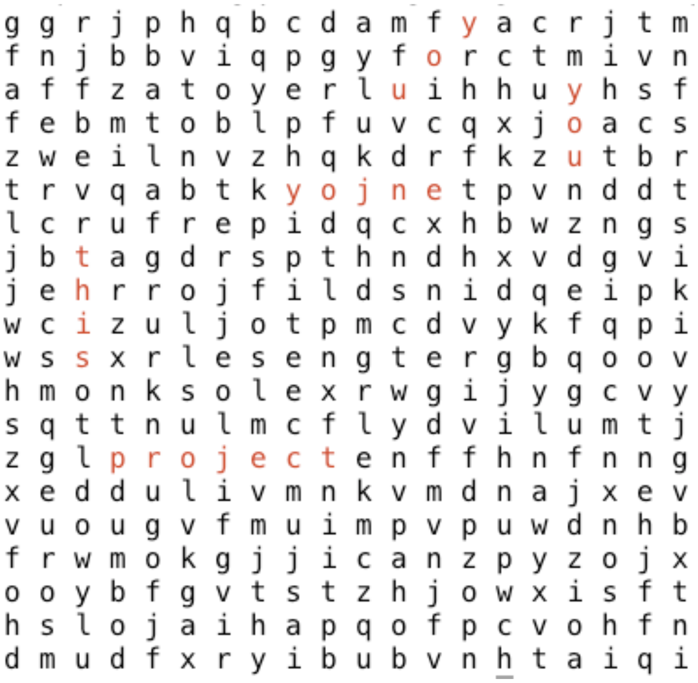
```
%./findwords this project is hard < 1520matrix.txt
```



```
x c v x b m g e l l b m n g h u c b i u
n v t n n n e e k i l u c q c b g p a p
v g c h z c w p r o j e c t s a q a e n
e w p l i w q q u y h k t a t s v l f v
o a q a u s w q e i k w t i z h s h k h
i b i w r l m d a u g z f t z o j r s a
m c h h o f o k j y e k y s o b i u m i
k m p m c v a b d v z t r o g j l g j f
u w f x e s q g r r t r p q k i u n e r
g z r h f i t c c r a u h r x q k i i v
z t u t e e f k r q o h t q t p w a r x
a g l t g r t k k g d d l s l c z j e x
m w p q n m s o r b s v i c e b i u q u
d y v x v x s a u h p w r t b j m h w d
k x b g b a z f i c i z v w h t m r j w
```

Similarly, to search you enjoy this project inside 2020matrix.txt, you can type:

```
%./findwords you enjoy this project < 2020matrix.txt
```



```
g g r j p h q b c d a m f y a c r j t m
f n j b b v i q p g y f o r c t m i v n
a f f z a t o y e r l u i h h u y h s f
f e b m t o b l p f u v c q x j o a c s
z w e i l n v z h q k d r f k z u t b r
t r v q a b t k y o j n e t p v n d d t
l c r u f r e p i d q c x h b w z n g s
j b t a g d r s p t h n d h x v d g v i
j e h r r o j f i l d s n i d q e i p k
w c i z u l j o t p m c d v y k f q p i
w s s x r l e s e n g t e r g b q o o v
h m o n k s o l e x r w g i j y g c v y
s q t t n u l m c f l y d v i l u m t j
z g l p r o j e c t e n f f h n f n n g
x e d d u l i v m n k v m d n a j x e v
v u o u g v f m u i m p v p u w d n h b
f r w m o k g j j i c a n z p y z o j x
o o y b f g v t s t z h j o w x i s f t
h s l o j a i h a p q o f p c v o h f n
d m u d f x r y i b u b v n h t a i q i
```

How to print a letter with color: All that you have to do is output certain escape sequences to the terminal and it will change either background or foreground, note that this stays in that color until you **reset** it or change the colors again. All that you have to do is put the sequences in the string that you are outputting and it will change the colors. (B is background and F is foreground):

REDB	"\033[1;41m"	**REDF**	"\033[31m"
GREENB	"\033[1;42m"	**GREENF**	"\033[1;32m"
YELLOWB	"\033[1;43m"	**YELLOWF**	"\033[1;33m"
BLUEB	"\033[1;44m"	**BLUEF**	"\033[1;34m"
MAGENTAB	"\033[1;45m"	**MAGENTAF**	"\033[1;35m"
CYANB	"\033[1;46m"	**CYANF**	"\033[1;36m"
WHITEB	"\033[1;47m"	**WHITEF**	"\033[1;37m"
RESET	"\033[0m"		

An example java program to output Hello World in Red would be:
Code:

```
#include <iostream>
using namespace std;
int main(int argc, const char* argv[])
{
        const string REDF = "\033[31m";
        const string RESET  = "\033[0m";
        // REDF is what turns this red
        cout << REDF << "Hello, World!" << endl;
        // And then just clear the colors so
        // that your terminal doesn't still have the red
        cout << RESET;
        return 0;
}
```

How to search ONE word in the matrix: Your program should read the whole matrix. If the program is visiting [i][j] element now, it will check whether the word matches the strings on [i][j] element’s left side, right side, upward, downward, and four diagonal directions. If the program finds a match, it will mark a separating matrix to keep the record. This separating matrix can be an int array or char array. Please keep in mind that the input matrix may have multiple matches for one word. Your result must show all the matches. Each word is a char array that terminates with the null character '\0'. So you can loop through your word until you reach '\0'.

How to search multiple words in the matrix: Once your program can search one word, you can use a for loop to go over each word typed at the command line. Note all the words you type will be saved to argv[]. You can use **argc** to check how many words typed. To access the ith word, just use **argv[i]**.

How to read the matrix into memory: You can use cin just like we did in the lectures and labs. You can create a `char temp;` and then `cin >> temp;` to read one value from the matrix. Since the initialization of the matrix is tricky, you may use parseFile from fileParser.h and fileParser.cpp in your program. An example of how to use them is in readFile.cpp

How many functions do I need? It is up to you. It depends on how you will divide work among functions. But that does not mean you can/should just define one function, that is main(). If you do so, believe me, it is hard to do debugging. I prefer you define one function that will check whether the word match the string starting from [i][j] at **one** direction (up, down, left, right, diagonals). If there is match, another function will record this information into a separating matrix.

Do NOT make one function do too many things!!!!!

Anything else I should know about the function that checks match? If you are checking the string starting at [i][j] on right direction, your program will check letter by letter to the right side. Make sure that your program will NOT move over the right edge of the matrix. In other words, the match checking must be always inside the matrix.

Another thing you should be careful of is the arguments/parameters. Remember that most data types are passed by value by default, except for arrays which are passed by reference. Make sure that if you plan to modify the value that is passed in, then pass it as a reference.

Challenge for the bored. There are many opportunities for improvement. Try adding other features, such as searching for all words in a dictionary file. Or, print a different color for each word or based on the number of times the word is found. Or, make an alternative graphical output using SDL. In a graphical version, make a place to specify a word to search for.

Readable code:
Make sure that your code is readable: use descriptive variable names. Indent your code. Comment your code according to the commenting guidelines.

What to turn in:

1. JAR **every file** under work directory into a file called `Project2.jar`. Submit the jar file to Canvas under assignment `Project2`.

2. Due date: 11:59pm, April 26th, 2015. Canvas time rules!