BOGGLE

Slides adapted from Brendon Go and Rishi Bedi


pxleyes.com

What do you have to do?

- Set Up the Board
- Deal with the Human turn and Human word search function
- Deal with the computer turn and Computer word search function

Game Set Up

- First you'll have to draw the board. You first need to know if you will take in user input or draw a random board.
- If you draw a random board, you will have to "shake" the cubes and put them in random locations.

Useful code snippets

```
#include “shuffle.h”

shuffle(array, length);

#include “random.h”

randomInteger(0, 6);

#include <cctype>

 isalpha(ch);

#include “simpio.h”

getYesOrNo(“Do you want to eat cake? ”);
```

Human Turn

- Ask for the user input
- Check that the word is at least of length four
- Check that the word is in the dictionary

Human Word Search

Base Case:

- We have found what we are looking for

Otherwise enter the Recursive Case:

- For every possible option
  - “Choose” that option
  - Fully explore that option
  - “Unchoose” that option

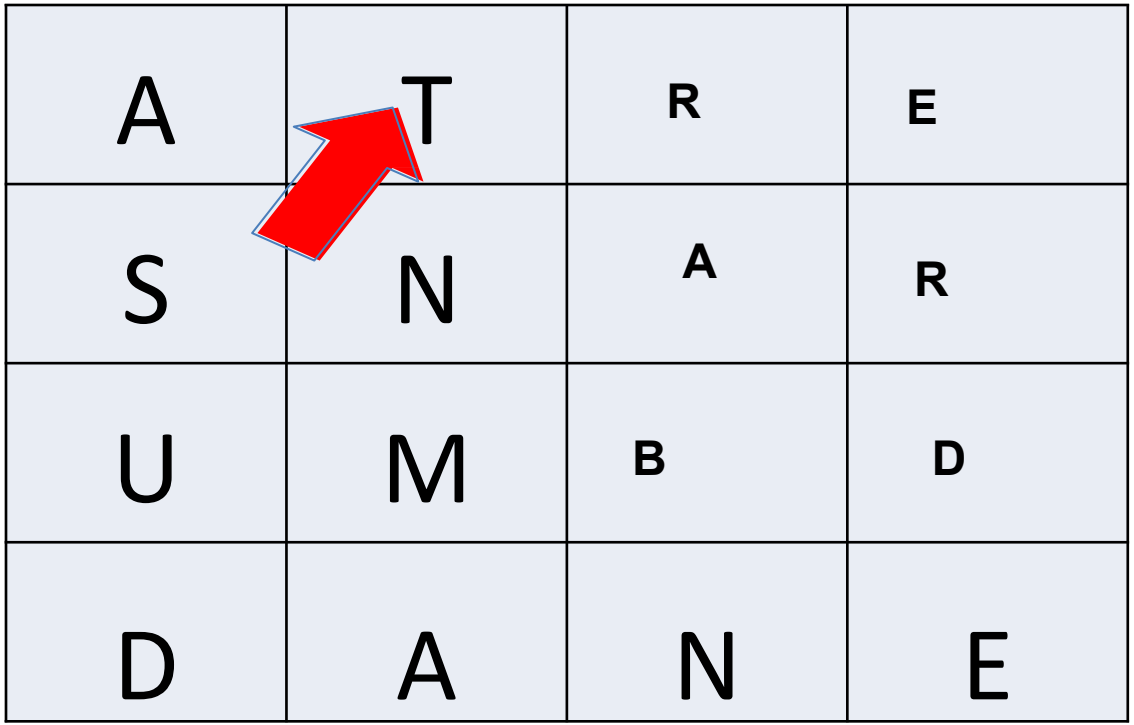
`humanWordSearch` Demo

word = “smart”

A	T	R	E
S	N	A	R
U	M	B	D
D	A	N	E

`humanWordSearch` Demo

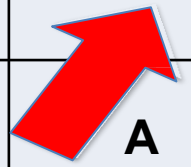
word = “smart”



`humanWordSearch` Demo

word = “smart”

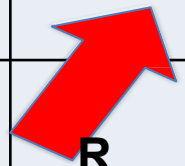
A	T	R	E
S	N	A	R
U	M	B	D
D	A	N	E



`humanWordSearch` Demo

word = “smart”

A	T	R	E
S	N	A	R
U	M	B	D
D	A	N	E



`humanWordSearch` Demo

word = “smart”

A	T	R E	
S	N	A.	R
U	M		
D	A	B. N	D E

- We found the first letter
  - Mark it as used
    - Why?

`humanWordSearch` Demo

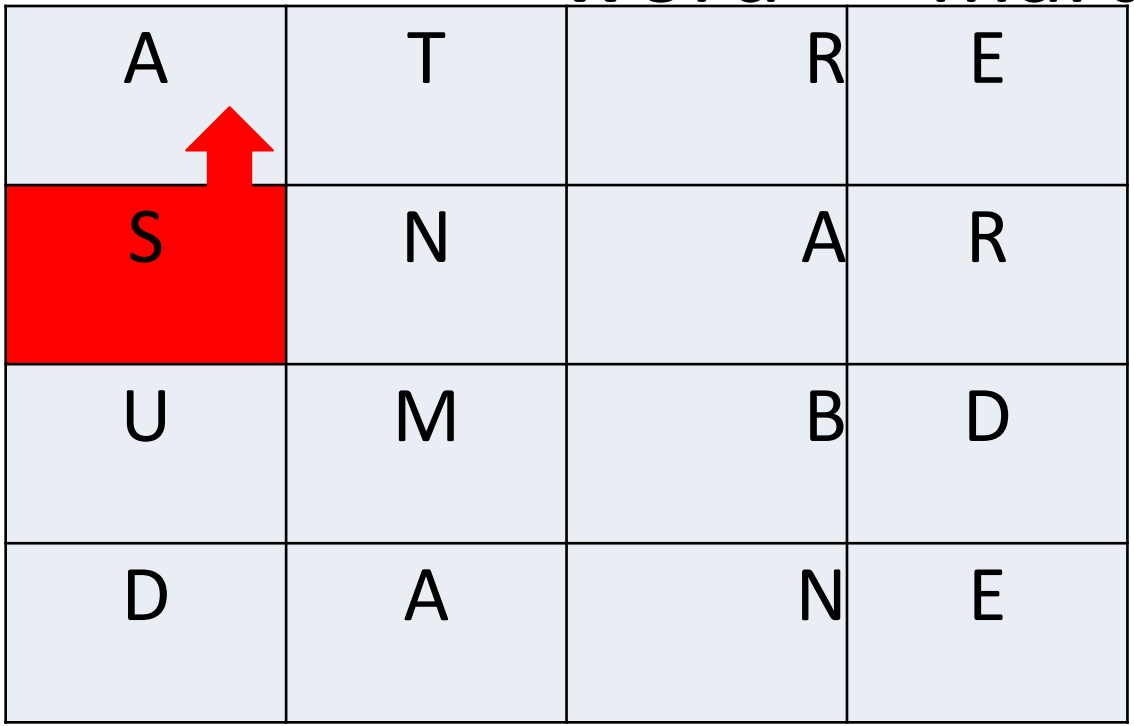
word = “mart”

A	T	R	E
S	N	A	R
U	M	B	D
D	A	N	E

- We found the first letter
  - Mark it as used
    - Why?
  - Highlight square
  - Look at its neighbors for the second letter.

`humanWordSearch` Demo

word = “mart”



- We found the first letter
  - Mark it as used
    - Why?
  - Highlight square
  - Look at its neighbors for the second letter.

`humanWordSearch` Demo

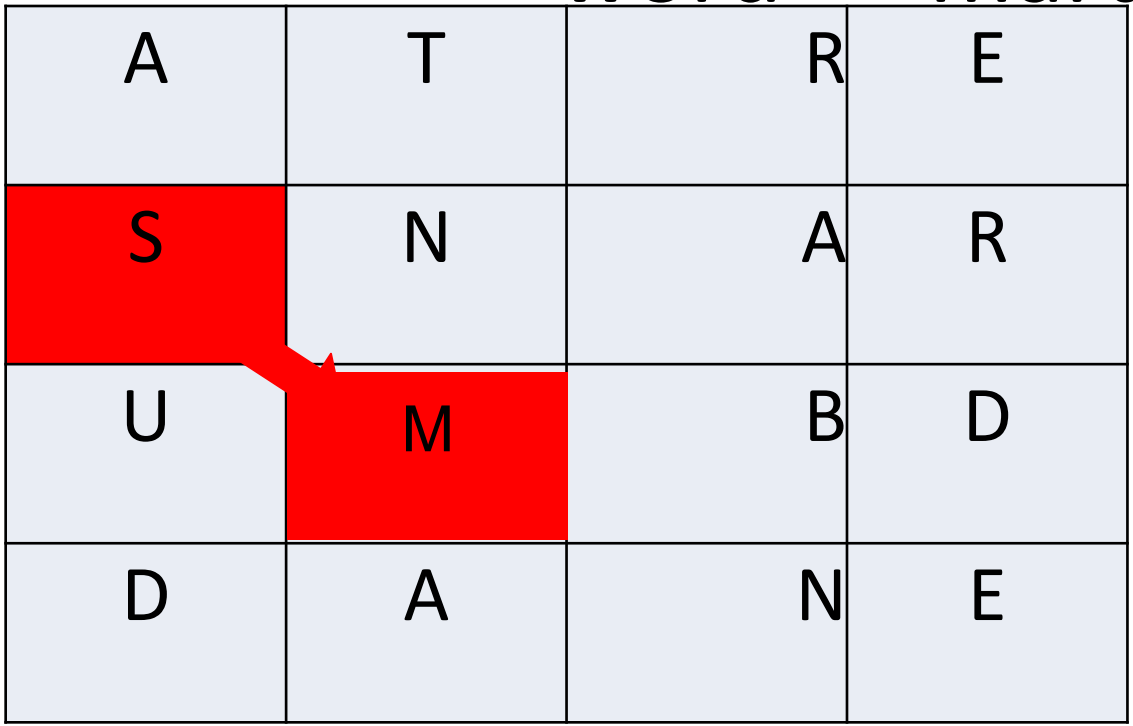
word = “mart”

A	T	R	E
S	N	A	R
U	M	B	D
D	A	N	E

- A few steps later...

`humanWordSearch` Demo

word = “mart”



- A few steps later...

`humanWordSearch` Demo

…a few steps

A	T	R	E
S	N	A	R
U	M	B	D
D	A	N	E

- How do we know when we are here?
  - That's our base case
- *What if that first "S" did not work out?*
  - Keep looking

A Quick Note:

You biggest ally here is a helper method!

The computer turn

Simply allow computer word search to do its thing!

Computer Word Search

- A similar, but very different recursive backtracking problem.
- You are not looking for ALL of the words on the board, not just a single word.
- This means your BASE CASE will be very different...

`computerWordSearch()` Demo

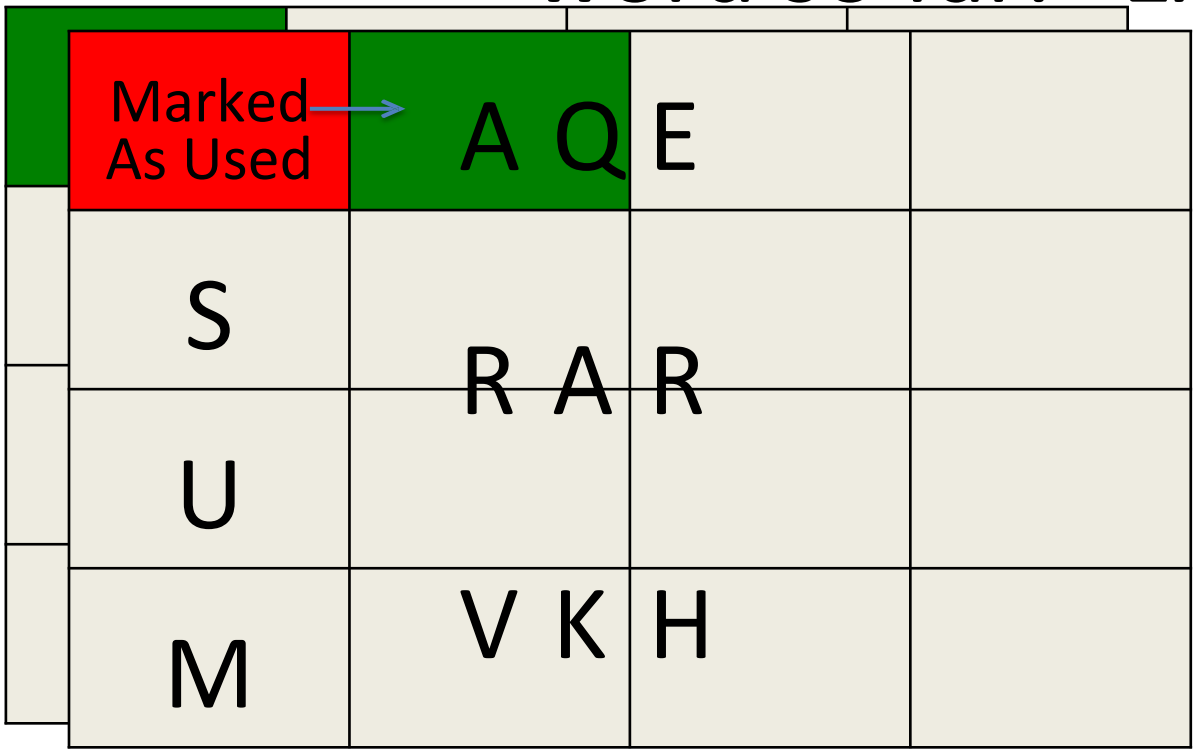
word so far: “E”

E	A	Q	E
S	R	A	R
U	V	K	H
M	E	J	O

Select each neighbor in turn and recurse down.

`computerWordSearch()` Demo

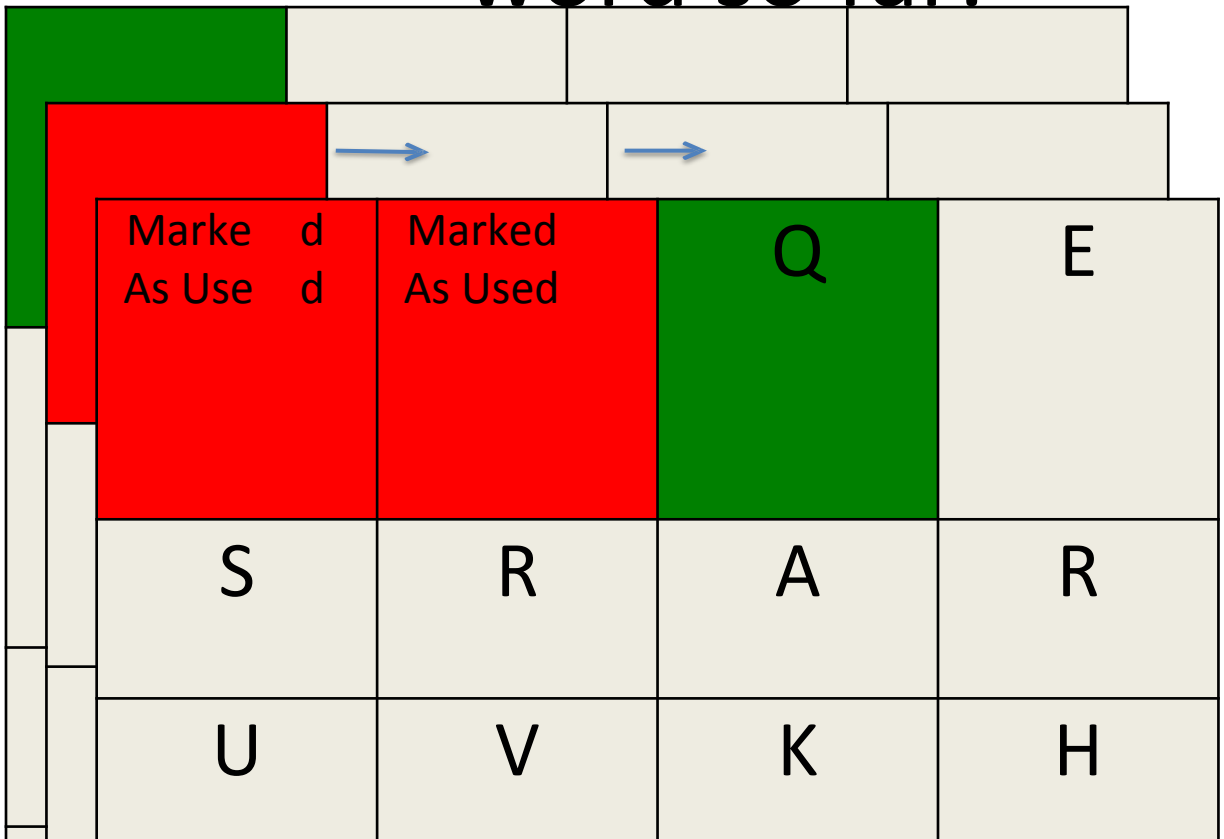
word so far: “EA”



Select each neighbor in turn and recurse down.

`computerWordSearch()` Demo

word so far:

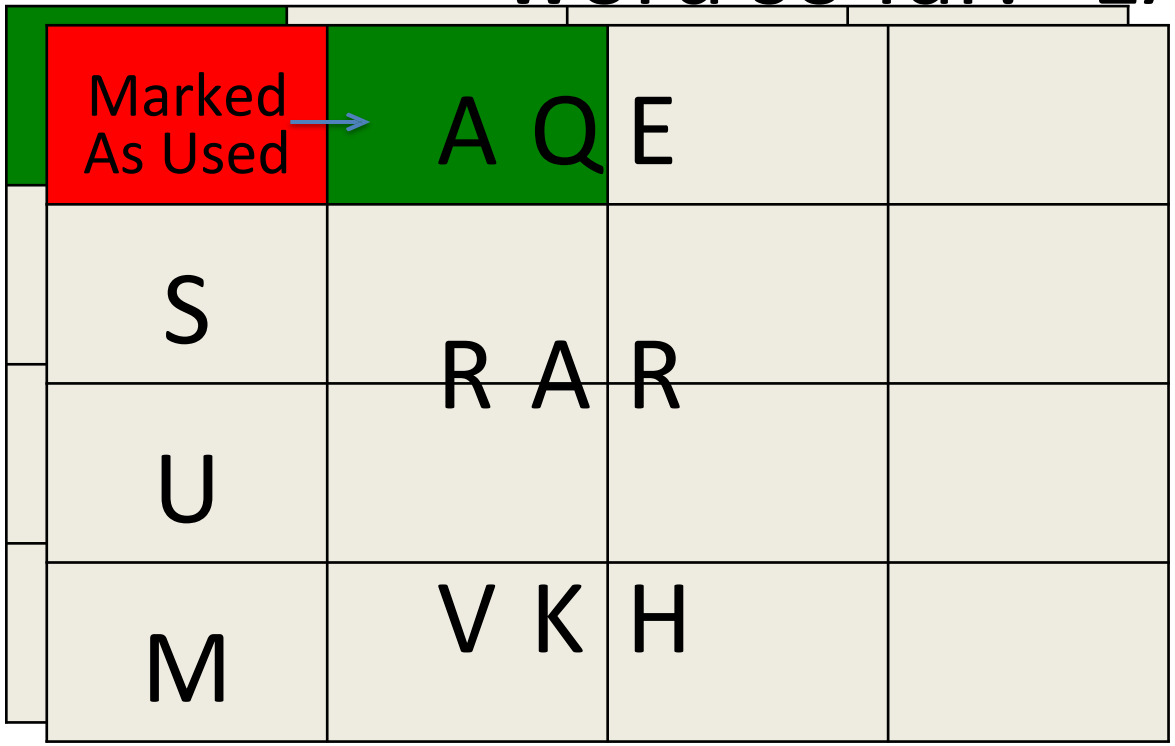


Select each neighbor in turn and recurse down.

BUT WAIT! EAQ is not the start of any english word! So should we continue??

`computerWordSearch()` Demo

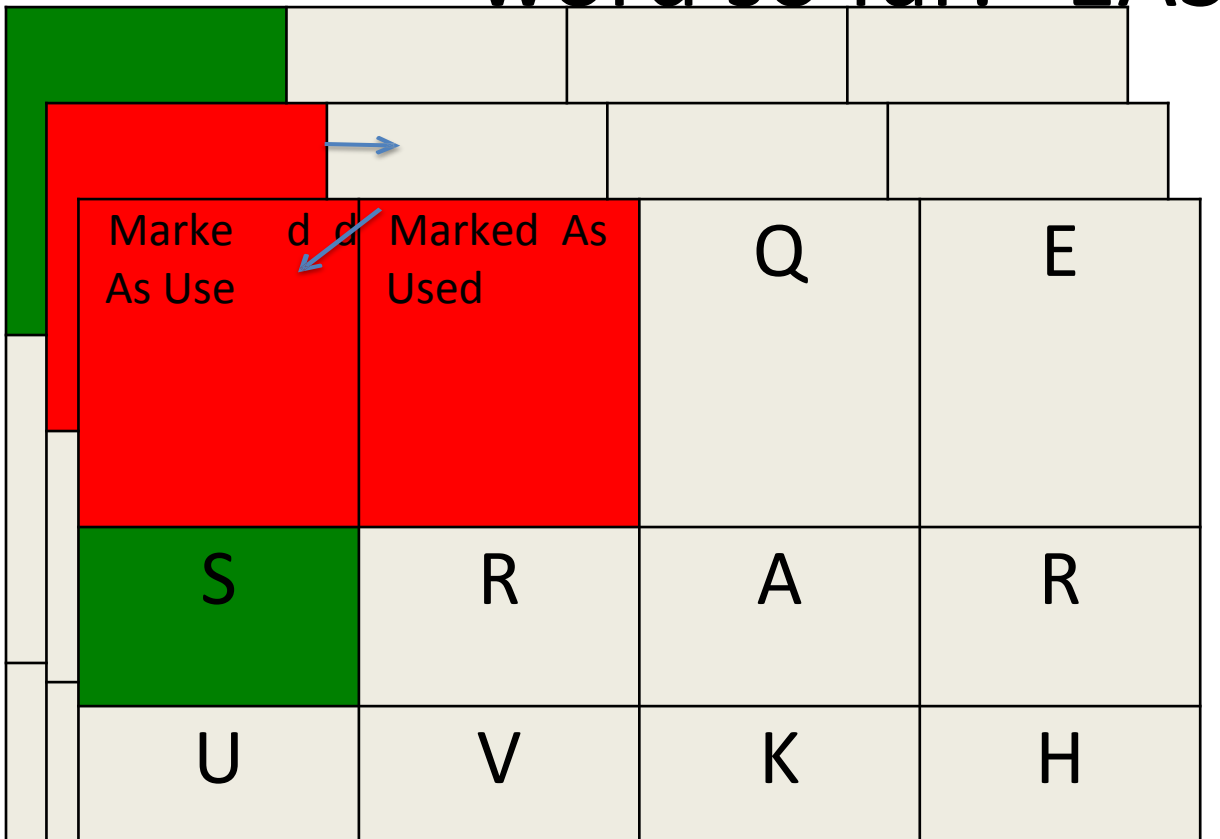
word so far: “EA”



Select each neighbor in turn and recurse down.

`computerWordSearch()` Demo

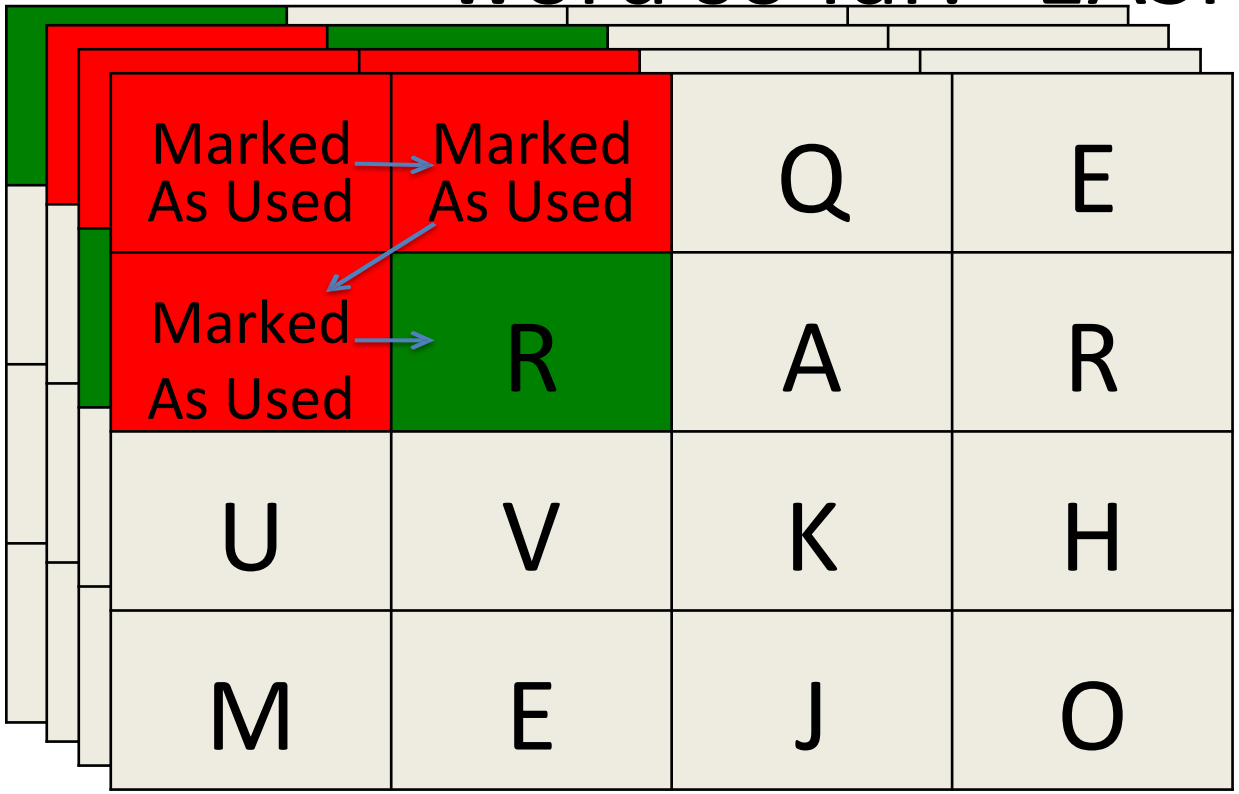
word so far: “EAS”



Select each neighbor in turn and recurse down.

`computerWordSearch()` Demo

word so far: “EASR”

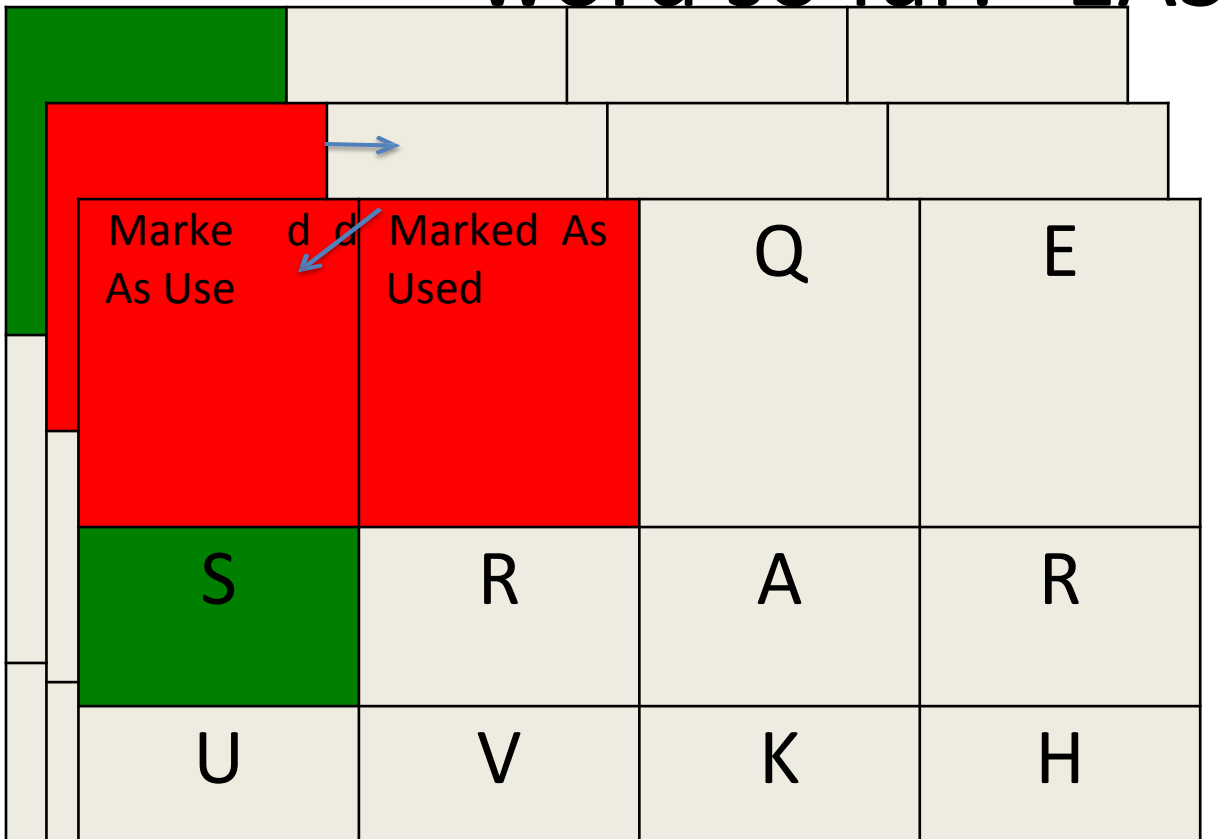


Select each neighbor in turn and recurse down.

But wait, no word begins with EASR!

`computerWordSearch()` Demo

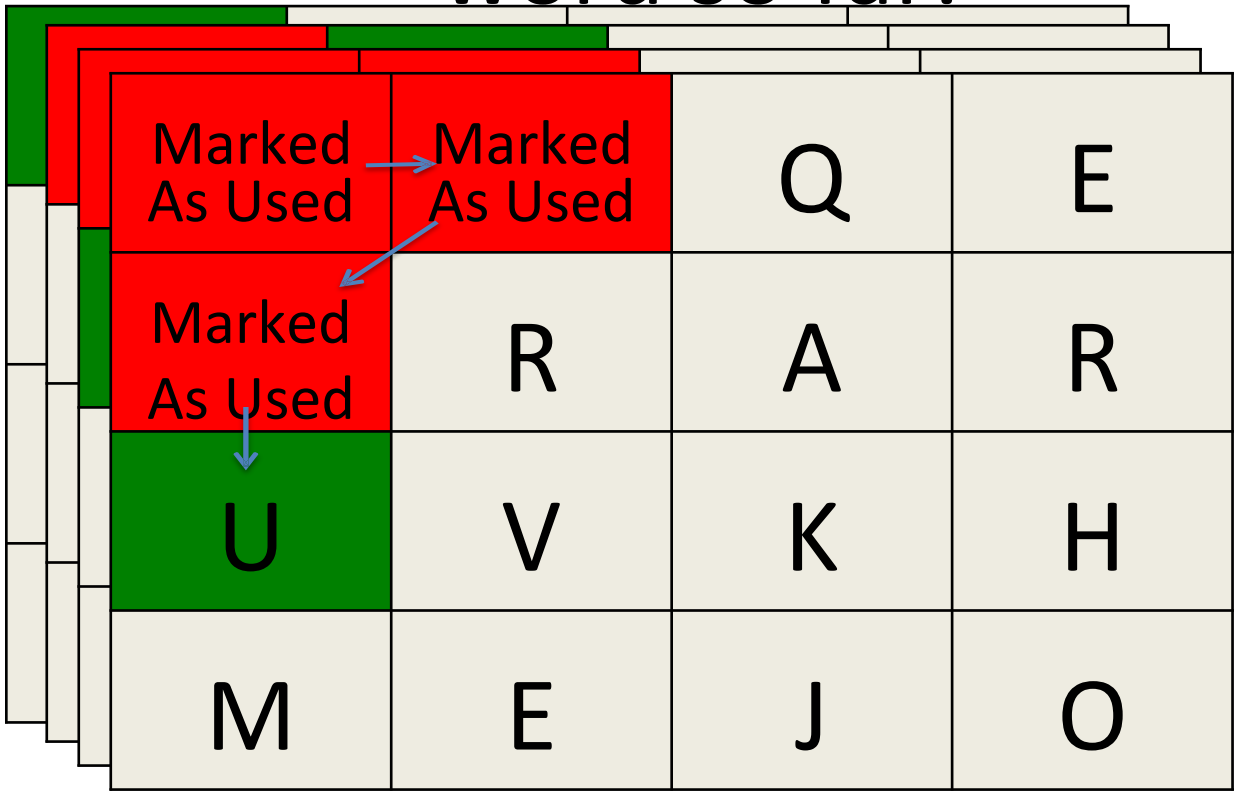
word so far: “EAS”



Select each neighbor in turn and recurse down.

`computerWordSearch()` Demo

word so far:

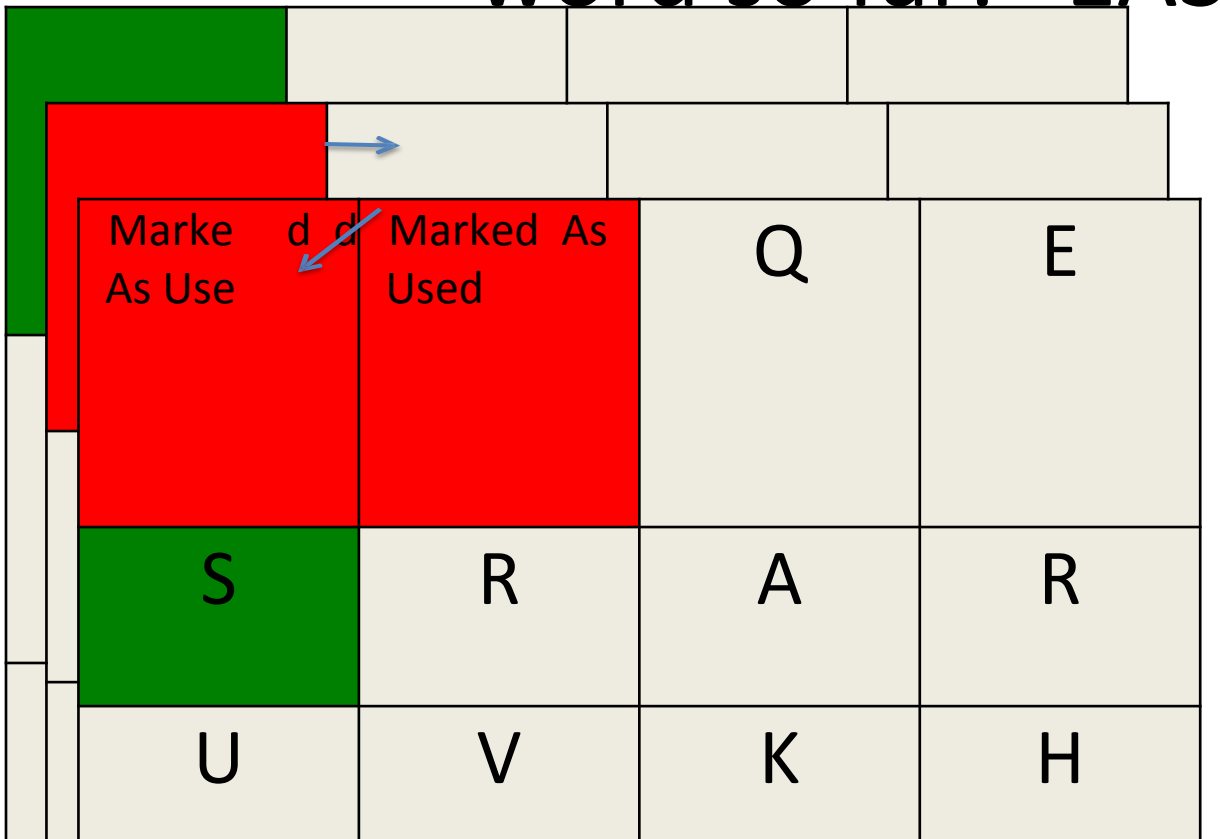


Select each neighbor in turn and recurse down.

But wait, no word begins with "EASU"!

`computerWordSearch()` Demo

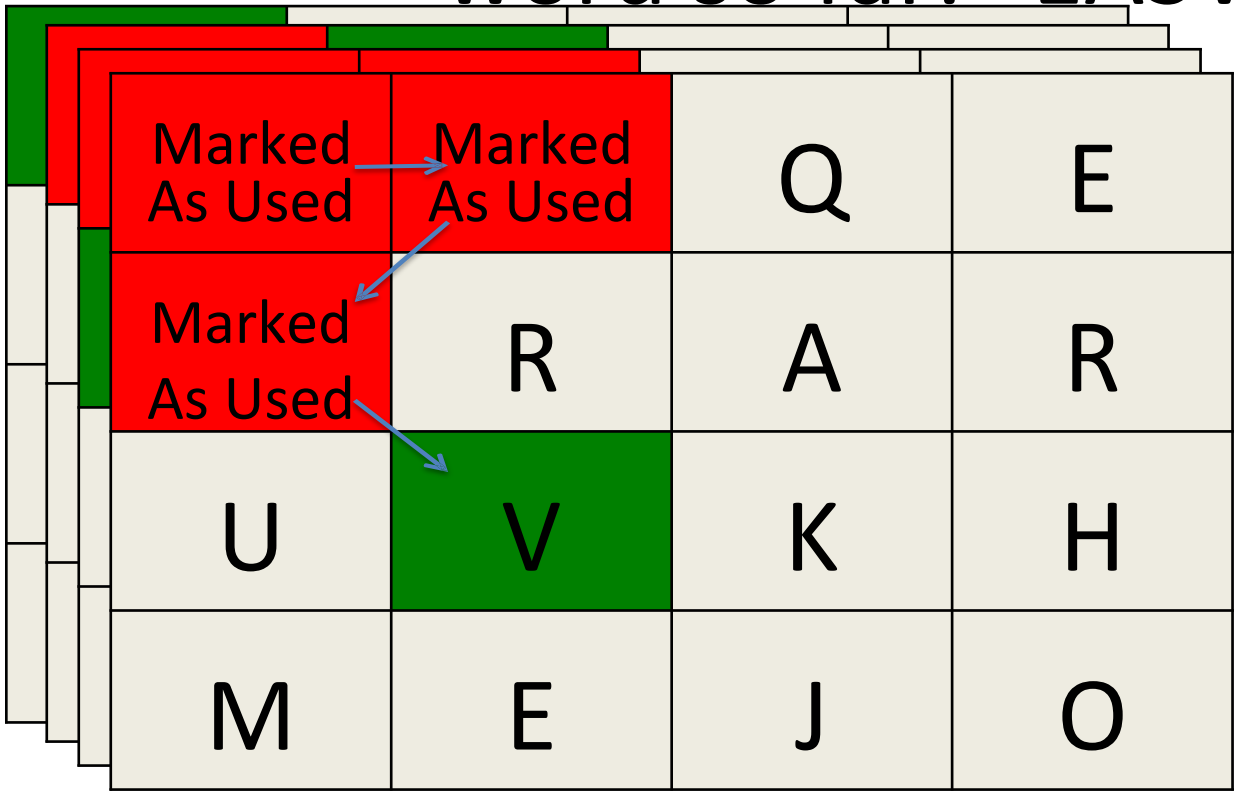
word so far: “EAS”



Select each neighbor in turn and recurse down.

`computerWordSearch()` Demo

word so far: “EASV”

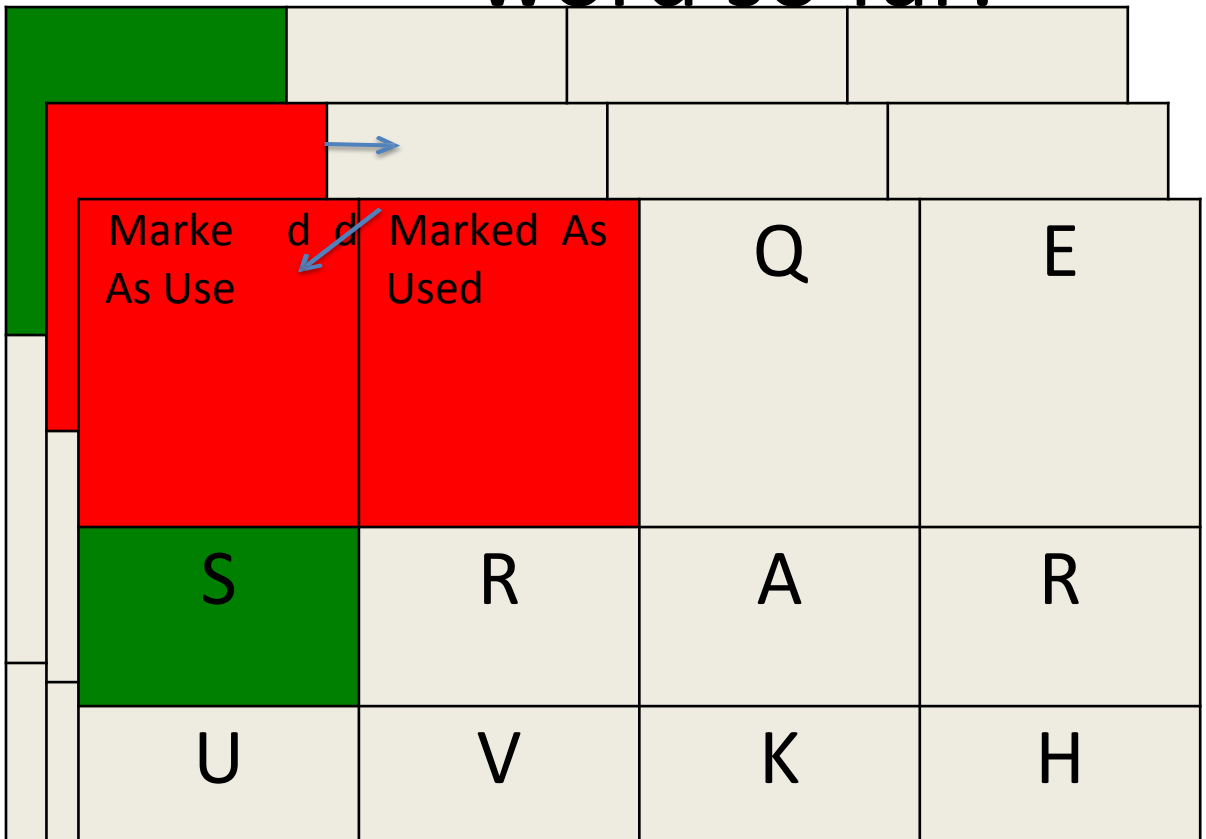


Select each neighbor in turn and recurse down.

STOP! No words start with “EASV”!

`computerWordSearch()` Demo

word so far:

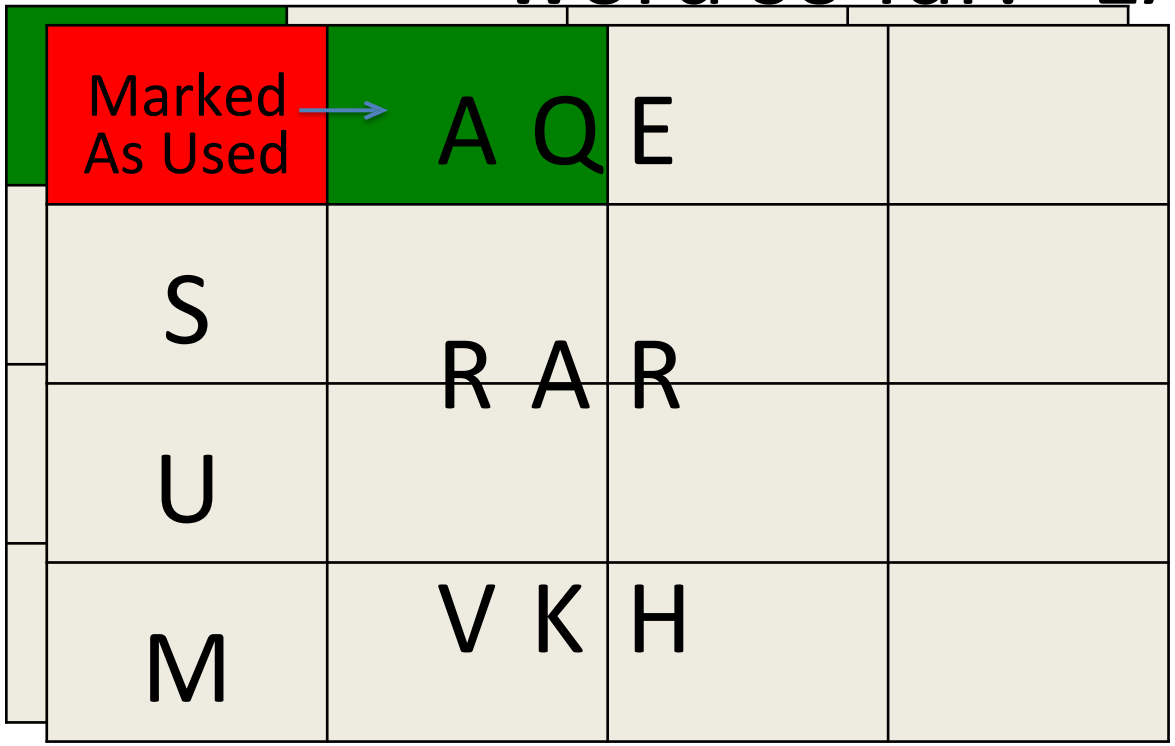


Select each neighbor in turn and recurse down.

We have looked at all of S's neighbors, so we will head back up.

`computerWordSearch()` Demo

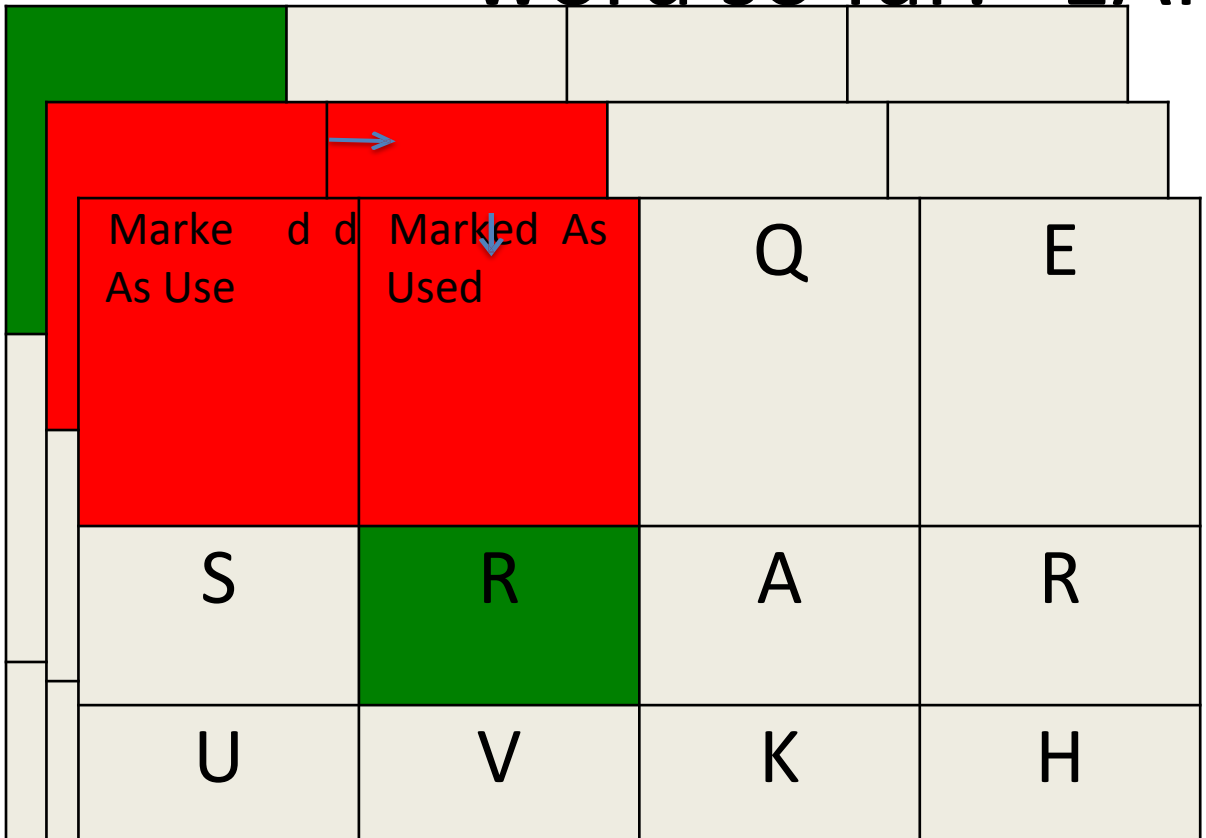
word so far: “EA”



Select each neighbor in turn and recurse down.

`computerWordSearch()` Demo

word so far: “EAR”

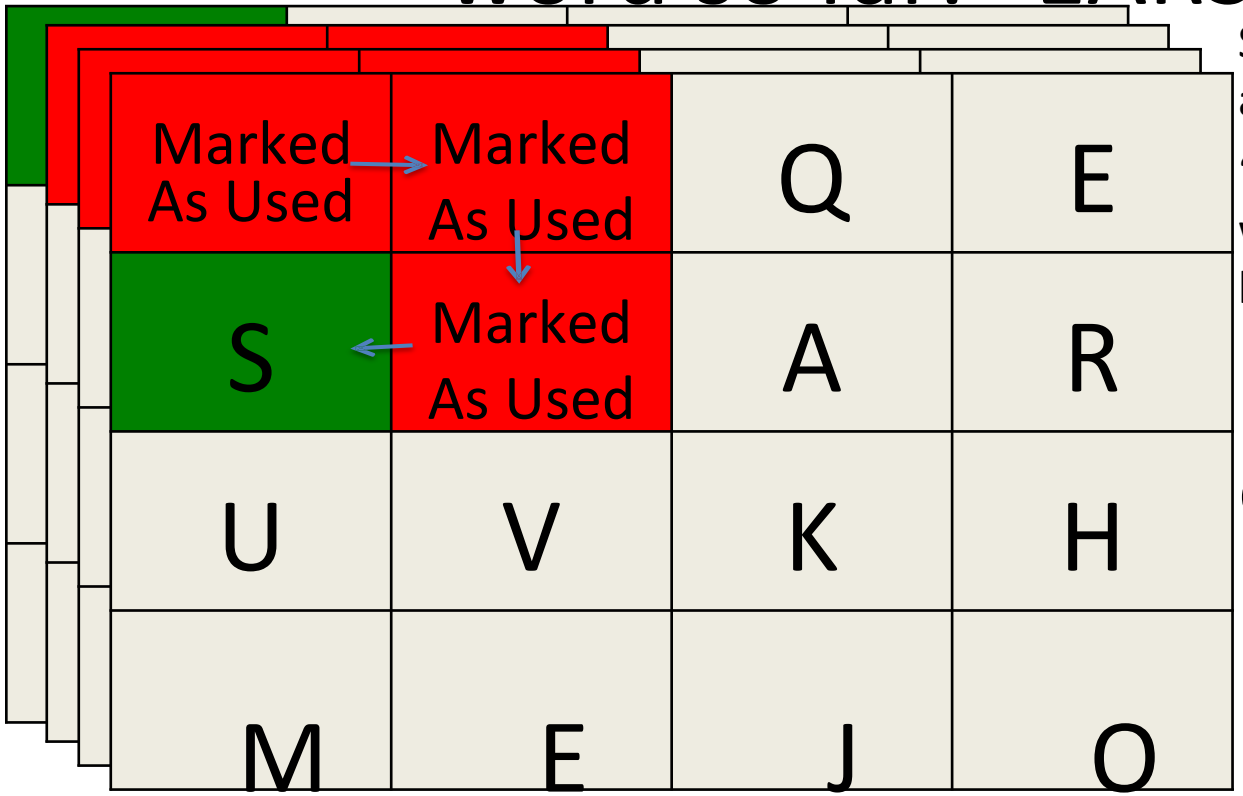


Select each neighbor in turn and recurse down.

“EAR” is a word, *but it is not 4 letters*.

computerWordSearch() Demo

word so far: “EARS”



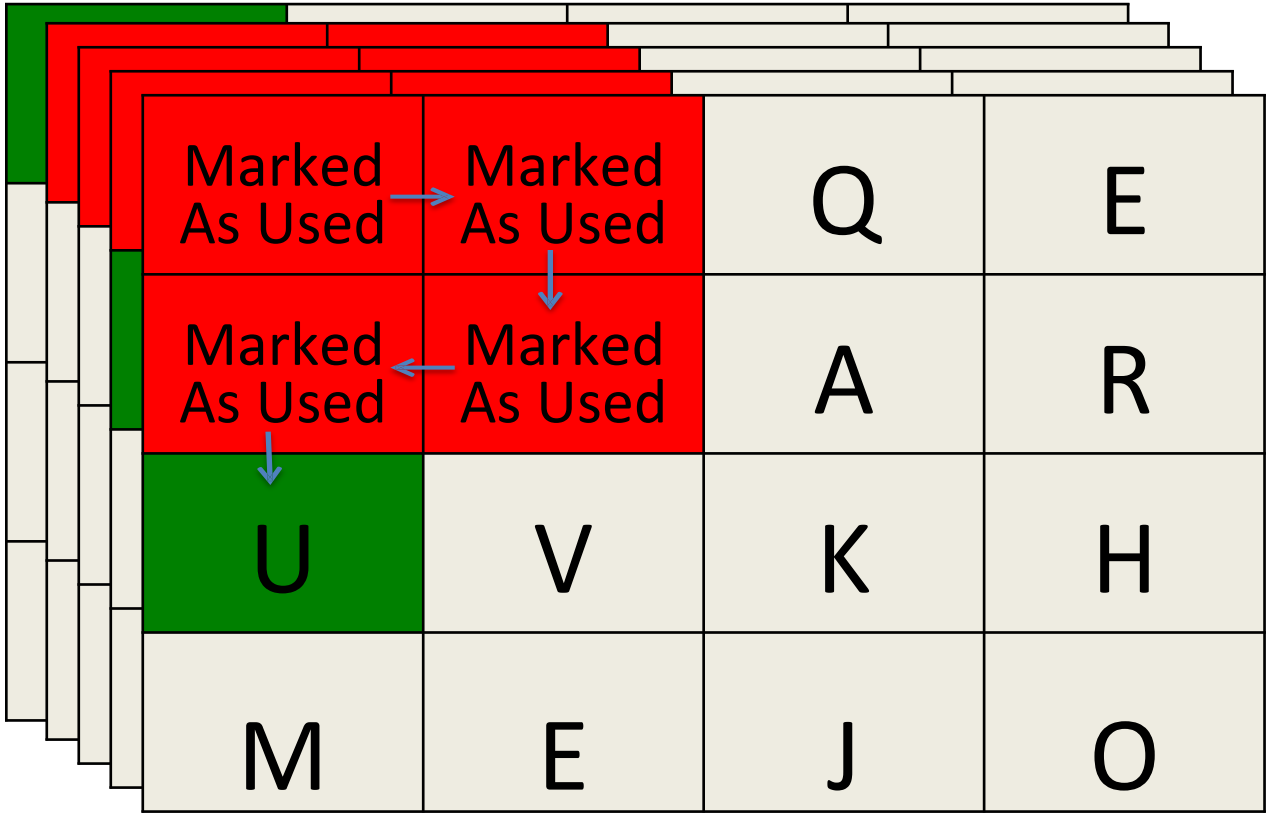
Select each neighbor in turn and recurse down.

“EARS”! Hey, that’s a word and it’s 4 letters at least.

Let’s add it to our set, and **keep looking**!

`computerWordSearch()` Demo

word so far:



Select each neighbor in turn and recurse down.

A helpful code snippet

```
Use lexicon.containsPrefix(wordSoFar);
```

A quick note on the GUI

- I highly recommend dealing with the coloring ONCE you've gotten boggle working
- Consider what the colors mean in regard to our choose, explore, unchoose framework!

GOOD LUCK!