Boggle

To run

Just make sure that sowpods.txt is in the bin directory and open a command line there. Type “Java boggle” to run.

Task 1

The following lines from BoggleBoard.java will generate a 5x5 boggle board.

```
BoggleBoard.makeDice();
BoggleBoard.makeBoard();
```

And then to display the board.

```
BoggleBoard.printBoard();
```

Task 2

```
For word in dictionary
        For character in word
                If first character in word
                        For row X on board
                                For column Y on board
                                        If board(x,y)  equals character
                                                Initialise path(x,y)
                                                Queue path in pathlist
                                        Endif
                                Endfor
                        endfor
                else
                        for every path in pathlist dequeue
                                for every neighbour of last coord of path
                                        if neighbour not in path
                                                if neighbour equals character
                                                        initialise path p as new path(path)
+neighbour
                                                        enqueue p
                                                endif
                                        endif
                                endfor
                        endfor
                endif
        endfor
        if pathlist is not empty
                add word to foundwords
        endif
endfor
for word in foundwords
        print word
```

endfor

Worst case is every word is on the boggle board so every neighbour of every character of every word is searched. However each word is still only processed once making it O(n) where n is the number of words.

Task 3a

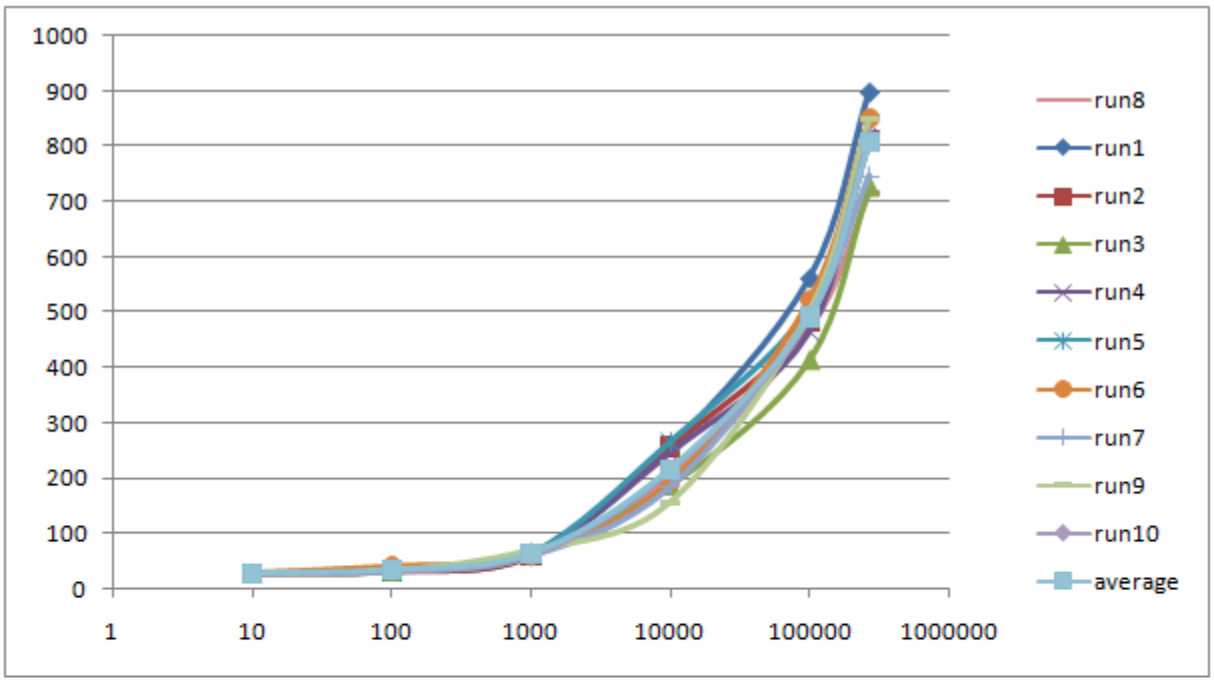
A much better way to run boggle would be to look at every possible path on the boggle board and trim a trie data structure. Worst possible case is there is a 25 letter word for every path, making complexity of the datastructure (m*T(m)*C), where n is the number of dice and where T(m) is the m^{th} triangular number and C is the size of each node in the trie. This would be O(m^2) space taken.

The time to initialise would be O(n) where n is the number of words as for each word, you depth first traverse the trie until you are at the correct letter and add a marker, or adding leaves for new characters and a marker.

Task 4: Running Time

	words	words	words	words	words	words
	10	100	1000	10000	100000	267751
run1	28	32	59	256	562	899
run2	29	32	60	257	484	811
run3	28	32	66	187	413	726
run4	28	33	61	246	465	841
run5	28	33	63	265	508	806
run6	28	40	61	193	523	851
run7	27	33	61	185	495	747
run8	27	34	64	201	471	717
run9	27	34	71	158	504	851
run10	28	32	61	213	490	816
average	27.8	33.5	62.7	216.1	491.5	806.5

Logarithmic scale



Non-logarithmic scale