DALHOUSIE UNIVERSITY

CSCI 3901: Software Development Concepts

Assignment 4: Scrabble Board

Name:Abhishek Latawa **Date:12 November,2023**

Problem 1: Writing Test Cases

Method 1 : boolean loadboard(BufferedReader puzzlestream)

S No.	Description of test cases(Property)	Property Input of data	Expected and outcome and it's side effects	Type of Test Case
1	If board is loaded successfully	BufferedReader puzzleStream	Returns True	Input Validation
2	If the puzzlestream is empty	BufferedReader puzzleStream	Returns False	Boundary Case
3	If the puzzlestream contains invalid characters	BufferedReader puzzleStream	Returns False	Boundary Case
4	If the puzzlestream row is incomplete	BufferedReader puzzleStream	Returns False	Input Validation
5	If there are more then one starting cell in the puzzlestream	BufferedReader puzzleStream	Returns False	Boundary case
6	Word Multipliers should always be in the uppercase	BufferedReader puzzleStream	Returns True	Boundary case
7	No. of rows are more then the board size	BufferedReader puzzleStream	Returns False	Data flow

Method 2:boolean Dictionary(BufferedReader wordstream)

S No.	Description of test cases(Property)	Property Input of data	Expected and outcome and it's side effects	Type of Test Case
1	When wordstream contains valid words with one word per line	BufferedReader wordStream	Returns True	Control Flow
2	When wordstream contains words in Uppercase	BufferedReader wordStream	Returns False	Input Validation
3	When wordstream is empty	BufferedReader wordStream	Returns False	Input Validation
4	When wordstream contains invalid characters	BufferedReader wordStream	Returns False	Input validation
5	When wordstream contains repetitive words	BufferedReader wordStream	Returns False	Input Validation
6	When wordstream contains white spaces at the start and end	BufferedReader wordStream	Return true	Input validation
7	When wordstream contains numeric values	BufferedReader wordStream	Returns False	Input Validation

Method 3:boolean letterVlaue(BufferedReader valuestream)

S No.	Description of test cases(Property)	Property Input of data	Expected and outcome and it's side effects	Type of Test Case
1	When valuestream contains two tab separated words with string and integer	BufferedReader valueStream	Returns false	Input Validation
2	When valuestream do not contains two tab separated words with string and integer	BufferedReader valueStream	Returns false	Input Validation
3	When Valuestream is empty	BufferedReader	Returns true	Boundary Value

		valueStream		
4	When valuestream String value contains invalid characters	BufferedReader valueStream	Return true	Input Validation
5	When valuestream Integer value contains negative integer	BufferedReader valueStream	Return False	Boundary Value
6	When value stream contains empty String	BufferedReader valueStream	Return False	Input Validation
7	When valuestream Integer value contains null value	BufferedReader valueStream	Return False	Boundary Value

Method 4: Void Print(PrintWriter outstream)

S No.	Description of test cases(Property)	Property Input of data	Expected and outcome and it's side effects	Type of Test Case
1	When the puzzlestream is null	outstream	Returns false	Data flow
2	When puzzlestream contains small board	outstream	Return true	Input Validation
3	When puzzlestream contains very large board	outstream	Return true	Control Flow
4	When puzzle stream contains only one row	outstream	Return true	Data FLow
5	When puzzle stream contains only one cell	outstream	Return true	Dalat Flow

Method 5: int placeWords(List<String> words)

S No.	Description of test cases(Property)	Property Input of data	Expected and outcome and it's	Type of Test Case

			side effects	
1	When Board is empty	puzzlestream	Returns unsolved	Data flow
2	When board do not contains the Starting point	puzzlestream	returns unsolved	Data flow
3	When list of word is empty	String	returns unsolved	Input Validation
4	When List contains only one word	String	Should be placed on board	Input Validation
5	When list contains more then one valid words	String	Should be placed on board	Input Validation
6	When list contains word which is not present in the dictionary	String	Returns unsolved	Control Flow
7	When list of words contain Invalid letters	String	Returns unsolved	Input Validation
8	When list of word contains empty string	String	Returns unsolved	Input Validation
9	When more then one word is placed but the augmented word is not in dictionary	String	Returns unsolved	Control Flow
10	When list of words are in uppercase	String	Returns unsolved	Input Validation
11	When list of words are placed successfully on board	String	Returns true	Control flow
12	When word length is greater then the board length	String	Returns unsolved	Input Validation

Problem 2:

Overview:

In this I have Implemented the Java Class WordPlacement which is designed in such a ways that it creates a Scrabble board along with the dictionary of words and letter values. The Class prints the board with the help of the print method and places the word into the scrabbleboard with the Greedy approach (i.e. word which contains the max no. of points)

Files and external data

The Implementation contains the one curriculum class:

WordPlacement: In this class I have Implemented 5 methods stated below:

boolean loadboard: The boolean loadboard method accepts the board which checks all the input validations that can occur while accepting the board.It returns true if the board is loaded successfully and false otherwise.

boolean dictionary: This method accepts the lines of words with one word at a time. This method also checks all the input validations for the set of words and returns true if all words are valid and false otherwise.This method uses the getter get all dictionary words.

boolean letterValue(): This method accepts two tab separated words one is string and other is Integer value with all the validation checks for the proper formatting of the words that are accepted from the valuestream.

Void Print(): This method prints the board using a printwriter into the file and checks for the validation if the board is empty.

int placeWords():This method accepts the list of words that needs to be placed on the board. Each word placed should start with the starting cell which contains “*” symbol and all the other words placed in such a ways that it gets the maximum points

int calculateScore(): This method is used to calculate the score of the words placed on the scrabble board. It returns the score of each word placed on the board.

void placeWordAt(): This method is made to check if the words can be placed horizontally or vertically in the board.It places the word either horizontally or vertically on the board.

boolean canPlaceWordAt(): This method checks all the possible scenarios of the word to be placed on the board.It returns true if the word can be placed on the word either horizontally or either vertically.

Class BoardInput: this class contains two method one is boarddictionary and other one is charactervalues.

boolean boarddictonary(): This method returns the true if the dictionary is ready to be used false otherwise.

Boolean charactervalues(): This method accepts the character and their integer values and returns true if the stream data is read properly without any error and false otherwise.

Data structures and their relations to each other:

Maps:

newLetterValues: This Map stores the information of the lettervalues. For each letter there is one score point Integer. This Map is of String and Integer type when used as a key value pair and returns the information for each character that is stored in this map.

Placedwords : This Map is of String and a class type. This MAP stores the information of the placed word and keeps the track of the rows and columns of the placed word so that whenever the new word is added we should know from where we have to place the new word**.**

2D Arrays:

ScrabbleBoard:This data structure is used to load the board in the form of a matrix. It represents the rows and columns and stores the cell information like a matrix.If the Scrabble board is loaded successfully we can use this board to places the words in the board.

Sets:

Set<String> ValidWords: It is a Hashset which is used to store the information of the valid words from the dictionary.if the word is present in this set then we can use this dictionary to compare all the words that are being placed onto the board before placing.

These data structures work together to manage course prerequisites, capacities, and to detect cycles in prerequisites.

These data structures work together to get the scrabble board ready and to place the words onto the board. If the any of these data structures not ready to be used, we will not be able to place the word in the board or we will get exception and we cannot solve the puzzle.

Relations:

- **allCourses** represents the course prerequisites structure, linking courses to their prerequisites.
- **Coursewithdemand** relates prerequisite courses to their demand for other courses.
- **Courseswithcapacity** links course IDs to their seat capacities.
- Various sets like discovered, discovering, and addedCourses are used for course path and cycle detection, ensuring proper sequencing and avoiding infinite loops.

Assumptions:

- I have assumed that all the words that needs to be placed on to the board should be present in the dictionary i.e. Validwords set.
- Every word in the list is added on the board even if its substring is present already onto the board.

Choices:

- I have used the 2D arrays rather then the maps to store the information of the srabbleboard.

Key algorithms and design elements:

- The main key algorithm was the 2D arrays which is more time efficient than other datastructures.
- I have used Maps datastructures to store the information of the placed words as we will need to keep track of the already placed word.
- I have used one class named Placedwordinfo which keeps track of the rows and columns of the already placed words. It uses one constructor which initialises everytime when the word is stored in the Placed word Map.
- I have also taken a class which is intersection info which keeps track of the words crossing rows and columns.

Limitations:

- My Design will not work for the larger puzzle board.
- I have used a 2D array for my puzzle board Input which somewhere limits me to retrieve values easily.

References:

- ***[1]OpenAI, "ChatGPT," chat.openai.com, Nov 3, 2023. https://chat.openai.com***
- **[2]"GeeksforGeeks | A computer science portal for geeks,"** ***GeeksforGeeks*****, 2019. https://www.geeksforgeeks.org**
- ***Youtube***
- ***Stack Overflow***