SE 3XA3: Module Interface Specification Scrabble Project

Team #214, The Trifecta
Kanakabha Choudhri, Choudhrk
Lucia Cristiano, Cristial
Raymond Tu, Tur1

March 13, 2020

This document is the Module Interface Specification of the Scrabble Project being done by Team Trifecta.

Table 1: **Revision History**

Date	**Version**	**Notes**
Date 1	1.0	Notes
Date 2	1.1	Notes

Tile Module

Module

Tile Type

Uses

N/A

Syntax

Exported Constants

N/A

Exported Types

Tile = tuple of (letter: str, score: $\mathbb{N}$)

Exported Access Programs

Routine name	In	Out	Exceptions
init	str	Tile	invalid_size
getLetter		str	
getScore		$\mathbb{N}$	

Semantics

State Variables

letter
score

Environment Variables

None

State Invariant

$0 < score \leq 10$

Assumptions

N/A

Access Routine Semantics

init($letter$):

- transition: $score := LETTER\_VALUES[letter]$
- output: None
- exception: None

getLetter():

- transition: None
- output: letter
- exception: None

getScore():

- transition: None
- output: score
- exception: None

Local Constants

$LETTER\_VALUES$ = tuple of ($"A" : \mathbb{N}, "B" : \mathbb{N}, "C" : \mathbb{N}, "D" : \mathbb{N}, "E" : \mathbb{N}, "F" : \mathbb{N}, "G" : \mathbb{N}, "H" : \mathbb{N}, "I" : \mathbb{N}, "J" : \mathbb{N}, "K" : \mathbb{N}, "L" : \mathbb{N}."M" : \mathbb{N}, "N" : \mathbb{N}, "O" : \mathbb{N}, "P" : \mathbb{N}, "Q" : \mathbb{N}, "R" : \mathbb{N}, "S" : \mathbb{N}, "T" : \mathbb{N}, "U" : \mathbb{N}, "V" : \mathbb{N}, "W" : \mathbb{N}, "X" : \mathbb{N}, "Y" : \mathbb{N}, "Z" : \mathbb{N}$)

Bag Module

Module

Bag Type

Uses

Tile

Syntax

Exported Constants

N/A

Exported Types

Bag = list of Tiles

Exported Access Programs

Routine name	**In**	**Out**	**Exceptions**
init		Bag	
addToBag	Tile, N	Bag	
initBag			
takeFromBag		Tile	
getRemainingTiles		N	

Semantics

State Variables

Bag

Environment Variables

None

State Invariant

$0 \leq |Bag| \leq 100$

Assumptions

N/A

Access Routine Semantics

init():

- transition: $Bag \rightarrow Bag$
- output: None
- exception: None

addToBag(Tile, n):

- transition: $Bag \rightarrow Bag + n * Tiles$
- output: None
- exception: None

initBag():

- transition: $Bag \rightarrow Bag + a * Tiles(A) + b * Tiles(B) + ... + z * Tiles(Z)$ where a, b,..., z are the number of that lettered tile to be in the bag. Additionally shuffles the order of the letters.
- output: None
- exception: None

takeFromBag():

- transition: $|Bag| \rightarrow |Bag| - 1$
- output: $Bag(|Bag| - 1)$
- exception: None

getRemainingTiles():

- transition: None
- output: $|Bag|$
- exception: None

Rack Module

Module

Rack Type

Uses

Bag

Syntax

Exported Constants

N/A

Exported Types

Rack = set of Tiles where $t : Tile \in Bag$

Exported Access Programs

Routine name	In	Out	Exceptions
init	Bag	Rack	
addToRack			
initialize			
getRackStr		String	
getRackArr		Rack	
removeFromRack	Tile		
getRackLength		N	
replenishRack			

Semantics

State Variables

rack
bag

Environment Variables

None

State Invariant

$0 < |rack| \leq 7$

Assumptions

N/A

Access Routine Semantics

init(Bag):

- transition: $rack := \emptyset$
  $bag = Bag$
- output: None
- exception: None

addToRack():

- transition: $rack \rightarrow rack + t$
  where $t : Tile \in bag$
- output: None
- exception: None

initialize():

- transition: $rack \rightarrow rack + 7 * t$
  where $t : Tile \in bag$
- output: None
- exception: None

getRackStr():

- transition: None
- output: $r : Rack \rightarrow s : String$
  where r and s represent same set of characters.
- exception: None

getRackArr():

- transition: None
- output: rack
- exception: None

removeFromRack(tile):

- transition: $rack \rightarrow rack \setminus tile$
  where tile : Tile
- output: None
- exception: None

getRackLength():

- transition: None
- output: —rack—
- exception: None

replenishRack():

- transition: $rack \rightarrow rack + n * t$
  where $n : 7 - |rack|$
- output: None
- exception: None

Player Module

Module

Player Type

Uses

Bag, Rack

Syntax

Exported Constants

N/A

Exported Types

Player = tuple of $(rack : Rack, score : \mathbb{N})$

Exported Access Programs

Routine name	**In**	**Out**	**Exceptions**
init	Bag	Player	
getRackStr		String	
getRackArr		Rack	
increaseScore	$\mathbb{N}$		
getScore		$\mathbb{N}$	

Semantics

State Variables

Score
Rack

Environment Variables

None

State Invariant

N/A

Assumptions

N/A

Access Routine Semantics

init(Bag):

- transition: $Rack = t : Tile \in Bag$
  $score = 0$
- output: None
- exception: None

getRackStr():

- transition: None
- output: $r : Rack \rightarrow s : String$
  where r and s represent same set of characters.
- exception: None

getRackArr():

- transition: None
- output: Rack
- exception: None

increaseScore($increase$):

- transition: $score \rightarrow score + increase$
- output: None
- exception: None

getScore():

- transition: None
- output: score
- exception: None

Board Module

Module

Board Type

Uses

N/A

Syntax

Exported Constants

N/A

Exported Types

Board = 16×16 matrix of Tiles

Exported Access Programs

Routine name	**In**	**Out**	**Exceptions**
init		Board	
getBoard		Board	
updateBackBoard	N, N, String, String		

Semantics

State Variables

backBoard

Environment Variables

None

State Invariant

$|Board| = 256$

Assumptions

N/A

Access Routine Semantics

init():

- transition: $Board \rightarrow Board$
- output: None
- exception: None

getLetter():

- transition: None
- output: backBoard
- exception: None

updateBackBoard($row, column, direction, word$):

- transition: $Board \rightarrow Board + word$
  where first letter of word is added from Board[row][column] and the rest are added to row(right) or column(down) depending on direction.
- output: None
- exception: None

EndTurn Module

Module

Uses

Tiles, Bag, Rack

Syntax

Exported Constants

N/A

Exported Types

N/A

Exported Access Programs

Routine name	**In**	**Out**	**Exceptions**
updateFrontBoard	N, N, String, String	List	
removeTile	String, Rack		
exchangeTile	String, Rack		
calculateScore	$\mathbb{N}, \mathbb{N}$, String, String	N	
checkWinState	Rack, Rack, Bag	$\mathbb{B}$	

Semantics

State Variables

word_score

Environment Variables

None

State Invariant

N/A

Assumptions

N/A

Access Routine Semantics

updateFrontBoard($row, column, direction, word$):

- transition: Empty $List \rightarrow List$ of Tuples
- output: List of Tuples
- exception: None

removeTile($word, rack$):

- transition: $Rack \rightarrow Rack \setminus lettersinword$
  $Rack \setminus lettersinword \rightarrow (Rack \setminus lettersinword) + n$
  where n = letters in word.
- output: None
- exception: None

exchangeTile($word, rack$):

- transition: $Rack \rightarrow Rack$
- output: None
- exception: None

calculateScore($row, column, direction, word$):

- transition: $word\_score \rightarrow +(\forall letters \in word \cdot score)$
- output: word_score
- exception: None

checkWinState($rack1, rack2, bag$):

- transition: None
- output: $\mathbb{B}$
- exception: None

WordChecks Module

Module

N/A

Uses

N/A

Syntax

Exported Constants

N/A

Exported Types

N/A

Exported Access Programs

Routine name	**In**	**Out**	**Exceptions**
checkRack	String, Rack	$\mathbb{B}$	
checkInDict	String	$\mathbb{B}$	

Semantics

State Variables

N/A

Environment Variables

N/A

State Invariant

N/A

Assumptions

N/A

Access Routine Semantics

checkRack($word, rack$):

- transition: None
- output: $\mathbb{B}$
- exception: None

checkInDict($word$):

- transition: None
- output: $\mathbb{B}$
- exception: None