

# 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

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

# 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:  $Bag \rightarrow Bag + a * Tiles(A) + b * Tiles(B) + \dots + z * Tiles(Z)$   
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	<i>increase</i>		
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