SE 3XA3: MIS
Space Invaders

April 12, 2022

Team Information:

Team Number	Name	MACID
L03 G07	Qianlin Chen	chenq84
	Jiacheng Wu	wuj187
	Tingyu Shi	shit19

Table 1: Revision History

Date	**Developer(s)**	**Change**
January 26, 2022	All team members	Initial Document
March 18, 2022	Qianlin Chen	Display Modules
March 10, 2022	Jiacheng Wu	Control Modules
March 10, 2022	Tingyu Shi	Model modules
April 11, 2022	All team members	Revised document

Contents

List of Tables

List of Figures

1 MonsterColor Module

Module

MonsterColor

Uses

None

Syntax

Exported Constants

None

Exported Types

MonsterType = $\{RED, BLUE, PINK\}$ # Represent Three colors of monsters

Exported Access Programs

None (This is an Enum class in python)

Semantics

State Variables

None

State Invariant

None

Assumptions

None

Access Routine Semantics

None

Consideration

When implementing in Python, use Enum class.

2 BulletState Module

Module

BulletState

Uses

None

Syntax

Exported Constants

None

Exported Types

BulletState = $\{FIRE, READY\}$ # Represent two states of bullets

Exported Access Programs

None (This is an Enum class in python)

Semantics

State Variables

None

State Invariant

None

Assumptions

None

Access Routine Semantics

None

Consideration

When implementing in Python, use Enum class.

3 Monster Module

Template Module

Monster

Uses

MonsterColor, Bullet, pygame.sprite.Sprite

Syntax

Exported Constants

None

Exported Types

Monster = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new Monster	$\mathbb{R}$, $\mathbb{R}$, $\mathbb{R}$, $MonsterColor$, $\mathbb{R}$	Monster	IllegalArgumentException
~~setX~~ Method removed	$\mathbb{R}$		IllegalArgumentException
~~setY~~ Method removed	$\mathbb{R}$		IllegalArgumentException
~~getX~~ Method removed		$\mathbb{R}$	
~~getY~~ Method removed		$\mathbb{R}$	
~~getColor~~ Method removed		$MonsterColor$	
reduceLife			
isDead		$\mathbb{B}$	
update	$\mathbb{Z}$		
~~shoot~~ Method removed		*Bullet*	
getItemType		$\mathbb{Z}$	

Semantics

State Variables

speed: $\mathbb{R}$
~~*X*: $\mathbb{R}$~~
~~*Y*: $\mathbb{R}$~~
~~*monster_color*: *MonsterColor*~~
~~*X_change*: $\mathbb{R}$~~
~~*Y_change*: $\mathbb{R}$~~
life: $\mathbb{Z}$
itemType: $\mathbb{Z}$
image: *.png file*
rect: *image.get_rect (This is the API of pygame library)*

State Invariant

~~$0 \leq X \leq 736$~~

Assumptions

None

Access Routine Semantics

new Monster($x, y, color, s,$):

- transition:
  $speed, itemType := s, 1$
  $color = MonsterColor.GREEN \Rightarrow life := 1$
  $color = MonsterColor.BLUE \Rightarrow life := 2$
  $color = MonsterColor.PINK \Rightarrow life := 3$
  $image := corresponding\ image$
  $rect := image.get\_rect(topleft\ =\ (x,\ y))$
- output: $out := self$
- exception: exc $:= ((s < 0) \vee (x < 0) \vee (y < 0) \Rightarrow$ IllegalArgumentException)

~~setX(x)~~ Method removed:

- transition: $X := x$
- output: none
- exception: $((x < 0) \Rightarrow$ IllegalArgumentException)

~~setY(y)~~ Method removed:

- transition: $Y := y$
- output: none
- exception: $((y < 0) \Rightarrow$ IllegalArgumentException)

~~getX()~~ Method removed:

- transition: none
- output: $out := X$
- exception: none

~~getY()~~ Method removed:

- transition: none
- output: $out := Y$
- exception: none

~~getColor()~~ Method removed:

- transition: none
- output: $out := monster\_color$
- exception: none

reduceLife():

- transition: $life := life - 1$
- output: none
- exception: none

isDead():

- transition: none
- output: $life = 0$
- exception: none

update($direction$):

- transition:
  $rect.x := rect.x \ + \ (direction \times speed)$
- output: none
- exception: none

~~shoot()~~ Method removed:

- transition: none
- output: $new\ Bullet(20,\ X,\ Y)$
- exception: none

getItemType():

- transition: none
- output: $itemType$
- exception: none

4 SpaceShip Module

Template Module

SpaceShip

Uses

Bullet, pygame.sprite.Sprite

Syntax

Exported Constants

None

Exported Types

SpaceShip = ?

Exported Access Programs

Routine name	**In**	**Out**	**Exceptions**
new Monster	$\mathbb{R}$, $\mathbb{R}$, $(\mathbb{R}, \mathbb{R})$, $\mathbb{R}$, $\mathbb{Z}$	SpaceShip	IllegalArgumentException
~~setX~~ Method removed	$\mathbb{R}$		IllegalArgumentException
~~setY~~ Method removed	$\mathbb{R}$		IllegalArgumentException
~~getX~~ Method removed		$\mathbb{R}$	
~~getY~~ Method removed		$\mathbb{R}$	
~~moveLeft~~ Method removed			
~~moveRight~~ Method removed			
~~stopMove~~ Method removed			
reduceLife			
~~isDead~~ Method removed		$\mathbb{B}$	
boundaryDetection			
shoot		*Bullet*	
update			
move			
prepare_bullet			
getBulletsGroup		pygame sprite group	
increaseLife			
getLife		$\mathbb{Z}$	
setLife	$\mathbb{Z}$		
increaseBullet			

Semantics

State Variables

screen_size_info: $(\mathbb{R}, \mathbb{R})$
space_ship_number: $\mathbb{Z}$
speed: $\mathbb{R}$
~~*X*: $\mathbb{R}$~~
~~*Y*: $\mathbb{R}$~~
~~*X_change*: $\mathbb{R}$~~
life: $\mathbb{Z}$
image: *.png file*
rect: *image.get_rect (This is the API of pygame library)*
bullets_group: pygame sprite group
state: BulletState
shoot_time: $\mathbb{Z}$
bullet_number: $\mathbb{Z}$

State Invariant

~~$0 \leq X \leq 736$~~
$0 \leq life \leq 5$

Assumptions

None

Access Routine Semantics

new SpaceShip(x, y, size, s, number):

- transition: screen_size_info := size
  space_ship_number := number
  image := *corresponding image*
  rect := *image.get_rect*$(midbottom\ =\ (x,\ y))$
  speed := s
  life := 5
  bullets_group := new Pygame sprite group
  state := BulletState.READY
  shoot_time := 0
  bullet_number := 0

- output: *out* := *self*

- exception: exc := $((s < 0) \vee (x < 0) \vee (y < 0) \Rightarrow$ IllegalArgumentException)

~~setX(x)~~:

- transition: $X := x$

- output: none

- exception: $((x < 0) \Rightarrow \text{IllegalArgumentException})$

~~setY(y)~~:

- transition: $Y := y$
- output: none
- exception: $((y < 0) \Rightarrow \text{IllegalArgumentException})$

~~getX()~~:

- transition: none
- output: $out := X$
- exception: none

~~getY()~~:

- transition: none
- output: $out := Y$
- exception: none

~~moveLeft()~~:

- transition: $X\_change := -1 * speed$
- output: none
- exception: none

~~moveRight()~~:

- transition: $X\_change := speed$
- output: none
- exception: none

~~stopMove()~~:

- transition: $X\_change := 0$
- output: none
- exception: none

reduceLife():

- transition: $life := life - 1$
- output: none
- exception: none

~~isDead()~~:

- transition: none
- output: $life = 0$
- exception: none

boundaryDetection():

- transition:
  $rect.left \leq 0 \Rightarrow (rect.left := 0)$
  $rect.right \geq screen\_size\_info[0] \Rightarrow (rect.right := screen\_size\_info[0])$
- output: none
- exception: none

shoot():

- transition: none
- output: add bullet(s) to bullet_group according to user keyboard input
- exception: none

update():

- transition:
  move()
  boundaryDetection()
  shoot()
  prepare_bullet()
  bullets_group.update()
- output: none
- exception: none

move():

- transition:
  rect.x += speed if user wants to move right
  rect.x -= speed if user wants to move left
- output: none
- exception: none

prepare_bullet():

- transition: set up the gap time between two shootings
- output: none

- exception: none

getBulletsGroup():

- transition: none
- output: out := bullets_group
- exception: none

increaseLife():

- transition: life := life + 1
- output: none
- exception: none

getLife():

- transition: none
- output: out := life
- exception: none

setLife(newLife):

- transition: life := newLife
- output: none
- exception: none

increaseBullet():

- transition: none
- output: bullet_number := bullet_number + 1
- exception: none

5 Bullet Module

Template Module

Bullet

Uses

pygame.sprite.Sprite

Syntax

Exported Constants

None

Exported Types

Bullet = ?

Exported Access Programs

Routine name	**In**	**Out**	**Exceptions**
new Bullet	$\mathbb{R}$, $\mathbb{R}$, ($\mathbb{R}$, $\mathbb{R}$), $\mathbb{R}$, $\mathbb{Z}$	Bullet	IllegalArgumentException
~~setX~~Method removed	$\mathbb{R}$		IllegalArgumentException
~~setY~~Method removed	$\mathbb{R}$		IllegalArgumentException
~~getX~~Method removed		$\mathbb{R}$	
~~getY~~Method removed		$\mathbb{R}$	
~~setState~~Method removed	*BulletState*		
~~getState~~Method removed		*BulletState*	
move			
boundaryDetection			
update			

Semantics

State Variables

screen_size_info : $(\mathbb{R}, \mathbb{R})$
speed: $\mathbb{R}$
~~X: $\mathbb{R}$~~
~~Y: $\mathbb{R}$~~
~~*Y_change*: $\mathbb{R}$~~
~~*state: BulletState*~~
image: *.png file*
rect: *image.get_rect (This is the API of pygame library)*

State Invariant

none

Assumptions

None

Access Routine Semantics

new Bullet(x, y, $size$, s, direction):

- transition:
  screen_size_info := size
  speed := s
  $image$:= $corresponding\ image(direction\ is\ the\ condition)$
  $rect$:= $image.get\_rect(center\ =\ (x,\ y))$
- output: $out := self$
- exception: exc := $((s < 0) \vee (x < 0) \vee (y < 0) \Rightarrow$ IllegalArgumentException)

~~setX(x)~~:

- transition: $X := x$
- output: none
- exception: $((x < 0) \Rightarrow$ IllegalArgumentException)

~~setY(y)~~:

- transition: $Y := y$
- output: none
- exception: $((y < 0) \Rightarrow$ IllegalArgumentException)

~~getX()~~:

- transition: none
- output: $out := X$
- exception: none

~~getY()~~:

- transition: none
- output: $out := Y$
- exception: none

~~setState($newState$)~~:

- transition: $state := newState$
- output: none

- exception: none

~~getState()~~:

- transition: none
- output: *out* := *state*
- exception: none

move():

- transition: rect.y := rect.y + speed
- output: none
- exception: none

boundaryDetection():

- transition: rect.y $\leq$ 30 OR rect.y $\geq$ screen_size_info[1] $\Rightarrow$ kill self object
- output: none
- exception: none

update():

- transition:
  move()
  boundaryDetection()
- output: none
- exception: none

6 Score Module

Template Module

Score

Uses

none

Syntax

Exported Constants

None

Exported Types

Score = ?

Exported Access Programs

Routine name	**In**	**Out**	**Exceptions**
new score		Score	
getScore		$\mathbb{N}$	
~~increaseAmount~~ increase	$\mathbb{N}$		

Semantics

State Variables

$score$: $\mathbb{N}$

State Invariant

$score \geq 0$

Assumptions

None

Access Routine Semantics

new Score():

- transition: $score := 0$
- output: $out := self$
- exception: exc := none

getScore():

- transition: none
- output: $out := score$
- exception: none

~~increaseAmount~~ increase($amount$):

- transition: $score := score + amount$
- output: none
- exception: none

7 Block Module(newly added module)

Template Module

Block

Uses

pygame.sprite.Sprite

Syntax

Exported Constants

None

Exported Types

Obstacle = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new Block	$\mathbb{R}$, $\mathbb{R}$	Obstacle	

Semantics

State Variables

image: *.png file*
rect: *image.get_rect (This is the API of pygame library)*

State Invariant

None

Assumptions

None

Access Routine Semantics

new Block(x, y):

- transition:
  $image := corresponding\ image$
  $rect := image.get\_rect(topleft\ =\ (x,\ y))$
- output: $out := self$
- exception: exc := none

8 Obstacle Module

Template Module

Obstacle

Uses

Block, pygame.sprite.Sprite

Syntax

Exported Constants

None

Exported Types

Obstacles = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new Obstacle		Obstacle	
creat_one_obstacle	$\mathbb{Z}, \mathbb{Z}$		
getBlocksGroup		pygame sprite group	

Semantics

State Variables

~~X: $\mathbb{R}$~~
~~Y: $\mathbb{R}$~~
~~$Width$: $\mathbb{R}$~~
~~$Height$: $\mathbb{R}$~~
~~$Area$: $\mathbb{R}$~~
blocks_group := pygame sprite group

State Invariant

None

Assumptions

None

Access Routine Semantics

new Obstacle():

- transition:
  create_one_obstacle(50, 400)
  create_one_obstacle(250, 400)
  create_one_obstacle(450, 400)
  create_one_obstacle(650, 400)
- output: $out := self$
- exception: exc := none

create_one_obstacle(xStart, yStart):

- transition: create blocks starting at (xStart, yStart) and add all the blocks to blocks_group
- output: $out := self$
- exception: exc := none

getBlocksGroup():

- transition: none
- output: $out := blocks\_group$
- exception: exc := none

~~getX()~~:

- transition: none
- output: $out := X$
- exception: none

~~getY()~~:

- transition: none
- output: $out := Y$
- exception: none

~~getArea()~~:

- transition: none
- output: $out := Area$
- exception: none

~~reduce_area(*amount*)~~:

- transition: $Area := Area - amount$
- output: none
- exception: none

9 Ammo Module

Template Module

Ammo

Uses

pygame.sprite.Sprite

Syntax

Exported Constants

None

Exported Types

Ammo = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new Ammo	$\mathbb{R}$, $\mathbb{R}$, $\mathbb{R}$	Ammo	IllegalArgumentException
~~setX~~	$\mathbb{R}$		IllegalArgumentException
~~setY~~	$\mathbb{R}$		IllegalArgumentException
~~getX~~		$\mathbb{R}$	
~~getY~~		$\mathbb{R}$	
~~reduce_life~~			
~~isDead~~		$\mathbb{B}$	
~~move~~			
update	$\mathbb{Z}$		
getItemType		$\mathbb{Z}$	

Semantics

State Variables

speed: $\mathbb{R}$
~~*X*: $\mathbb{R}$~~
~~*Y*: $\mathbb{R}$~~
~~*X_change*: $\mathbb{R}$~~
~~*Y_change*: $\mathbb{R}$~~
~~*life*: $\mathbb{N}$~~
itemType: $\mathbb{Z}$
image: *.png file*
rect: *image.get_rect (This is the API of pygame library)*

State Invariant

None

Assumptions

None

Access Routine Semantics

new Ammo(x, y, s):

- transition:
  speed := s
  itemType := 3
  image := *corresponding image*
  $rect := image.get\_rect(topleft = (x, y))$
- output: $out := self$
- exception: exc := $((s < 0) \vee (x < 0) \vee (y < 0) \Rightarrow$ IllegalArgumentException)

~~setX(x)~~:

- transition: $X := x$
- output: none
- exception: $((x < 0) \Rightarrow$ IllegalArgumentException)

~~setY(y)~~:

- transition: $Y := y$
- output: none
- exception: $((y < 0) \Rightarrow$ IllegalArgumentException)

~~getX()~~:

- transition: none
- output: $out := X$
- exception: none

~~getY()~~:

- transition: none
- output: $out := Y$
- exception: none

~~reduce_life()~~:

- transition: $life := life - 1$
- output: none
- exception: none

~~isDead()~~:

- transition: none
- output: $life = 0$
- exception: none

~~move()~~:

- transition:
  $X := X + X\_change$
  $X \leq 0 \Rightarrow (X\_change,\ Y := speed,\ Y + Y\_change)$
  $X \geq 736 \Rightarrow (X\_change,\ Y := -1 * speed,\ Y + Y\_change)$
- output: none
- exception: none

update(direction):

- transition:
  rect.x := rect.x + (direction $\times$ speed)
- output: none
- exception: none

getItemType():

- transition: None
- output: out := itemType
- exception: none

10 Bomb Module

Template Module

Bomb

Uses

pygame.sprite.Sprite

Syntax

Exported Constants

None

Exported Types

Bomb = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new Bomb	$\mathbb{R}$, $\mathbb{R}$, $\mathbb{R}$	Bomb	IllegalArgumentException
~~setX~~	$\mathbb{R}$		IllegalArgumentException
~~setY~~	$\mathbb{R}$		IllegalArgumentException
~~getX~~		$\mathbb{R}$	
~~getY~~		$\mathbb{R}$	
~~reduce_life~~			
~~isDead~~		$\mathbb{B}$	
~~move~~			
update	$\mathbb{Z}$		
getItemType		$\mathbb{Z}$	

Semantics

State Variables

speed: $\mathbb{R}$
~~*X*: $\mathbb{R}$~~
~~*Y*: $\mathbb{R}$~~
~~*X_change*: $\mathbb{R}$~~
~~*Y_change*: $\mathbb{R}$~~
~~*life*: $\mathbb{N}$~~
itemType: $\mathbb{Z}$
image: *.png file*
rect: *image.get_rect (This is the API of pygame library)*

State Invariant

None

Assumptions

None

Access Routine Semantics

new Bomb(x, y, s):

- transition:
  speed := s
  itemType := 4
  image := *corresponding image*
  $rect := image.get\_rect(topleft\ =\ (x,\ y))$
- output: $out := self$
- exception: exc := $((s < 0) \vee (x < 0) \vee (y < 0) \Rightarrow$ IllegalArgumentException)

~~setX(x)~~:

- transition: $X := x$
- output: none
- exception: $((x < 0) \Rightarrow$ IllegalArgumentException)

~~setY(y)~~:

- transition: $Y := y$
- output: none
- exception: $((y < 0) \Rightarrow$ IllegalArgumentException)

~~getX()~~:

- transition: none
- output: $out := X$
- exception: none

~~getY()~~:

- transition: none
- output: $out := Y$
- exception: none

~~reduce_life()~~:

- transition: $life := life - 1$
- output: none
- exception: none

~~isDead()~~:

- transition: none
- output: $life = 0$
- exception: none

~~move()~~:

- transition:
  $X := X + X\_change$
  $X \leq 0 \Rightarrow (X\_change,\ Y := speed,\ Y + Y\_change)$
  $X \geq 736 \Rightarrow (X\_change,\ Y := -1 * speed,\ Y + Y\_change)$
- output: none
- exception: none

update(direction):

- transition:
  rect.x := rect.x + (direction $\times$ speed)
- output: none
- exception: none

getItemType():

- transition: None
- output: out := itemType
- exception: none

11 Heart Module

Template Module

Heart

Uses

pygame.sprite.Sprite

Syntax

Exported Constants

None

Exported Types

Heart = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new Heart	$\mathbb{R}$, $\mathbb{R}$, $\mathbb{R}$	Heart	IllegalArgumentException
~~setX~~	$\mathbb{R}$		IllegalArgumentException
~~setY~~	$\mathbb{R}$		IllegalArgumentException
~~getX~~		$\mathbb{R}$	
~~getY~~		$\mathbb{R}$	
~~reduce_life~~			
~~isDead~~		$\mathbb{B}$	
~~move~~			
update	$\mathbb{Z}$		
getItemType		$\mathbb{Z}$	

Semantics

State Variables

speed: $\mathbb{R}$
~~*X*: $\mathbb{R}$~~
~~*Y*: $\mathbb{R}$~~
~~*X_change*: $\mathbb{R}$~~
~~*Y_change*: $\mathbb{R}$~~
~~*life*: $\mathbb{N}$~~
itemType: $\mathbb{Z}$
image: *.png file*
rect: *image.get_rect (This is the API of pygame library)*

State Invariant

None

Assumptions

None

Access Routine Semantics

new Heart(x, y, s):

- transition:
  speed := s
  itemType := 2
  image := *corresponding image*
  $rect := image.get\_rect(topleft\ =\ (x,\ y))$
- output: $out := self$
- exception: exc := $((s < 0) \vee (x < 0) \vee (y < 0) \Rightarrow$ IllegalArgumentException)

~~setX(x)~~:

- transition: $X := x$
- output: none
- exception: $((x < 0) \Rightarrow$ IllegalArgumentException)

~~setY(y)~~:

- transition: $Y := y$
- output: none
- exception: $((y < 0) \Rightarrow$ IllegalArgumentException)

~~getX()~~:

- transition: none
- output: $out := X$
- exception: none

~~getY()~~:

- transition: none
- output: $out := Y$
- exception: none

~~reduce_life()~~:

- transition: $life := life - 1$
- output: none
- exception: none

~~isDead()~~:

- transition: none
- output: $life = 0$
- exception: none

~~move()~~:

- transition:
  $X := X + X\_change$
  $X \leq 0 \Rightarrow (X\_change,\ Y := speed,\ Y + Y\_change)$
  $X \geq 736 \Rightarrow (X\_change,\ Y := -1 * speed,\ Y + Y\_change)$
- output: none
- exception: none

update(direction):

- transition:
  rect.x := rect.x + (direction × speed)
- output: none
- exception: none

getItemType():

- transition: None
- output: out := itemType
- exception: none

12 CollisionDectection Module This module has been deleted

Service Module

Service

Uses

None

Syntax

Exported Constants

None

Exported Types

None

Exported Access Programs

Routine name	In	Out	Exceptions
isCollided	$\mathbb{R}$, $\mathbb{R}$, $\mathbb{R}$, $\mathbb{R}$	$\mathbb{B}$	

Semantics

State Variables

None

State Invariant

None

Assumptions

None

Access Routine Semantics

isCollided (x_1, x_2, y_1, y_2):

- transition: none
- output: $distance(x_1, x_2, y_1, y_2) \leq 27$
- exception: exc := none

Local Function

distance: [$\mathbb{R}$, $\mathbb{R}$, $\mathbb{R}$, $\mathbb{R}$] $\Rightarrow \mathbb{R}$
distance$(x_1, x_2, y_1, y_2) \equiv \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$

13 MonsterMatrix Module

Template Module

MonsterMatrix

Uses

Monster, Ammo, Heart, Bomb, pygame.sprite.Sprite

Syntax

Exported Constants

None

Exported Types

MonsterMatrix = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new MonsterMatrix	$\mathbb{Z}$, $\mathbb{R}$, $(\mathbb{N}, \mathbb{N})$	MonsterMatrix	IllegalArgumentException
~~getMatrix~~		seq of seq [Monster, Ammo, Heart, Bomb]	
~~move~~			
shoot			
round1			
round2			
round3			
round4			
round5			
getMonstersGroup		pygame sprite group	
boundaryDetection			
move_down			
getBulletsGroup		pygame sprite group	
update			

Semantics

State Variables

speed: $\mathbb{R}$
~~*M: seq of seq[Ammo Monster Heart Bomb]*~~
monsters_group : pygame sprite group
direction : $\mathbb{Z}$
screen_size_info: $(\mathbb{R}, \mathbb{R})$
bullets_group: pygame sprite group

State Invariant

None

Assumptions

None

Access Routine Semantics

new MonsterMatrix($round, s$):

- transition:
  $speed := s$
  $round = 1 \Rightarrow M := m1(with\ monsters\ randomly\ replaced\ by\ Ammo,\ Bomb,\ Heart)$
  $round = 2 \Rightarrow M := m2(with\ monsters\ randomly\ replaced\ by\ Ammo,\ Bomb,\ Heart)$
  $round = 3 \Rightarrow M := m3(with\ monsters\ randomly\ replaced\ by\ Ammo,\ Bomb,\ Heart)$
  $round = 4 \Rightarrow M := m4(with\ monsters\ randomly\ replaced\ by\ Ammo,\ Bomb,\ Heart)$
  $round = 5 \Rightarrow M := m5(with\ monsters\ randomly\ replaced\ by\ Ammo,\ Bomb,\ Heart)$
- output: $out := self$
- exception: exc := $(\neg(0 \leq round \leq 5) \vee (s < 0)) \Rightarrow$ IllegalArgumentException)

round1():

- transition: add m1 to monsters_group with game items randomly replaced
- output: none
- exception: none

round2():

- transition: add m2 to monsters_group with game items randomly replaced
- output: none
- exception: none

round3():

- transition: add m3 to monsters_group with game items randomly replaced
- output: none
- exception: none

round4():

- transition: add m4 to monsters_group with game items randomly replaced
- output: none
- exception: none

round5():

- transition: add m5 to monsters_group with game items randomly replaced
- output: none
- exception: none

getMonstersGroup():

- transition: none
- output: out := monsters_group
- exception: none

boundaryDetection():

- transition: if any of the monster in monsters_group touches the edge of the screen, move the whole monster matrix down.
- output: none
- exception: none

move_down():

- transition: $\forall\ gameItem \in monsters\_group : gameItem.rect.y += \ 2$
- output: none
- exception: none

~~getMatrix()~~ Method removed:

- transition: none
- output: $out := M$
- exception: none

~~move()~~ Method removed:

- transition: Monster Matrix moves in direction east $\rightarrow$ south $\rightarrow$ west
- output: none
- exception: none

shoot():

- transition: Monsters M shoot bullets randomly and add bullets object to bullets_group.
- output: none
- exception: none

getBulletsGroup():

- transition: none
- output: out := bullets_group
- exception: none

update():

- transition:
  $\forall\ gameItem \in monsters\_group : gameItem.update()$
  boundaryDetection()
  bullets_group.update()
- output: none
- exception: none

Local Types

$$m1 \equiv \begin{bmatrix} GM & GM & GM & GM & GM & GM & GM & GM & GM & GM \\ GM & GM & GM & GM & GM & GM & GM & GM & GM & GM \\ GM & GM & GM & GM & GM & GM & GM & GM & GM & GM \\ GM & GM & GM & GM & GM & GM & GM & GM & GM & GM \\ GM & GM & GM & GM & GM & GM & GM & GM & GM & GM \end{bmatrix}$$

$$m2 \equiv \begin{bmatrix} BM & BM & BM & BM & BM & BM & BM & BM & BM & BM \\ BM & BM & BM & BM & BM & BM & BM & BM & BM & BM \\ GM & GM & GM & GM & GM & GM & GM & GM & GM & GM \\ GM & GM & GM & GM & GM & GM & GM & GM & GM & GM \\ GM & GM & GM & GM & GM & GM & GM & GM & GM & GM \end{bmatrix}$$

$$m3 \equiv \begin{bmatrix} BM & BM & BM & BM & BM & BM & BM & BM & BM & BM \\ BM & BM & BM & BM & BM & BM & BM & BM & BM & BM \\ BM & BM & BM & BM & BM & BM & BM & BM & BM & BM \\ BM & BM & BM & BM & BM & BM & BM & BM & BM & BM \\ BM & BM & BM & BM & BM & BM & BM & BM & BM & BM \end{bmatrix}$$

$$m4 \equiv \begin{bmatrix} PM & PM & PM & PM & PM & PM & PM & PM & PM & PM \\ BM & BM & BM & BM & BM & BM & BM & BM & BM & BM \\ BM & BM & BM & BM & BM & BM & BM & BM & BM & BM \\ BM & BM & BM & BM & BM & BM & BM & BM & BM & BM \\ GM & GM & GM & GM & GM & GM & GM & GM & GM & GM \end{bmatrix}$$

$$\text{m5} \equiv \begin{bmatrix} PM & PM & PM & PM & PM & PM & PM & PM & PM & PM \\ PM & PM & PM & PM & PM & PM & PM & PM & PM & PM \\ PM & PM & PM & PM & PM & PM & PM & PM & PM & PM \\ PM & PM & PM & PM & PM & PM & PM & PM & PM & PM \\ PM & PM & PM & PM & PM & PM & PM & PM & PM & PM \end{bmatrix}$$

GM means green monster.
BM means blue monster.
PM means pink monster.

14 MonsterDisplay Module This module has been deleted

UserInterface Module

MonsterDisplay

Uses

Monster, MonsterColor

Syntax

Exported Constants

N/A

Exported Types

MonsterDisplay = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new MonsterDisplay	pygame window object, MonsterColor	MonsterDisplay	
show	$\mathbb{N}$, $\mathbb{N}$, Boolean		

Semantics

Environment Variables

screen: It is the game screen that is manipulated by the following functions to alter display. This is update by a precise frame rate to depict various game objects on the game screen.

State Variables

$SCREEN$: *pygame window object*
img : Monster Picture

State Invariant

N/A

Assumptions

N/A

Access Routine Semantics

new MonsterDisplay($screen, monster\_color$)

- transition:
  $SCREEN := screen$
  $monster\_color = MonsterColor.GREEN \Rightarrow img :=$ green monster picture
  $monster\_color = MonsterColor.BLUE \Rightarrow img :=$ blue monster picture
  $monster\_color = MonsterColor.PINK \Rightarrow img :=$ pink monster picture

- output: $out := self$

- exception: None

show($x, y, isDead$):

- transition: $isDead \Rightarrow Display\ img\ at\ (x, y)$

- output: none

- input definitions: x and y represent the monster display coordinate. $isDead$ is used to clarify whether the monster is killed.

- exception: None

15 SpaceShipDisplay Module

UserInterface Module

SpaceShipDisplay

Uses

SpaceShip, pygame.sprite.Sprite

Syntax

Exported Constants

N/A

Exported Types

SpaceShipDisplay = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new SpaceShipDisplay	pygame window object, ~~$\mathbb{N}$~~, pygame sprite group	SpaceShipDisplay	
show	~~$\mathbb{N}$, $\mathbb{N}$~~		

Semantics

Environment Variables

screen: It is the game screen that is manipulated by the following functions to alter display. This is update by a precise frame rate to depict various game objects on the game screen.

State Variables

N/A

State Invariant

$SCREEN$: *pygame window object*
~~img : SpaceShip Picture~~
space_ship_group: pygame sprite group

Assumptions

N/A

Access Routine Semantics

new SpaceShipDisplay($screen$, ~~$spaceship\_number$~~, group)

- transition:
  $SCREEN := screen$
  ~~$space\_number = 1 \Rightarrow img :=$ spaceship1 picture~~
  ~~$space\_number = 2 \Rightarrow img :=$ spaceship2 picture~~
  space_ship_group := group
- output: $out := self$
- exception: None

show(~~x, y~~):

- transition: space_ship_group.draw(SCREEN) # This is pygame API
- output: none
- exception: None

16 BulletDisplay Module

UserInterface Module

BulletDisplay

Uses

BulletState, Bullet, pygame.sprite.Sprite

Syntax

Exported Constants

N/A

Exported Types

BulletDisplay = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new BulletDisplay	pygame window object, pygame sprite group	BulletDisplay	
show	~~$\mathbb{N}$, $\mathbb{N}$, *BulletState*~~		

Semantics

Environment Variables

screen: It is the game screen that is manipulated by the following functions to alter display. This is update by a precise frame rate to depict various game objects on the game screen.

State Variables

$SCREEN$: *pygame window object*
~~*img* : Bullet Picture~~
bullets_group: pygame sprite group

State Invariant

N/A

Assumptions

N/A

Access Routine Semantics

new BulletDisplay($screen$, bullets)

- transition:
  $SCREEN := screen$
  ~~img := bullet picture~~
  bullets_group := bullets

- output: $out := self$

- exception: None

show(~~x, y, $state$~~):

- transition: ~~$state = BulletState.FIRE \Rightarrow$ Display img at (x, y)~~
  bullets_group.draw(SCREEN) # This is pygame API

- output: none

- exception: None

17 ScoreDisplay Module

UserInterface Module

ScoreDisplay

Uses

Score

Syntax

Exported Constants

N/A

Exported Types

ScoreDisplay = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new ScoreDisplay	pygame window object	ScoreDisplay	
show	$\mathbb{N}$, $\mathbb{N}$, $\mathbb{N}$		

Semantics

Environment Variables

screen: It is the game screen that is manipulated by the following functions to alter display. This is update by a precise frame rate to depict various game objects on the game screen.

State Variables

$SCREEN : pygame\ window\ object$

State Invariant

N/A

Assumptions

N/A

Access Routine Semantics

new ScoreDisplay($screen$)

- transition: $SCREEN := screen$
- output: $out := self$
- exception: None

show($x, y, score$):

- transition: Display $score$ at (x, y).
- input definitions: x and y represent the score's coordinate. $score$ represent the total scores of player(s).
- output: none
- exception: None

18 ObstaclesDisplay Module

UserInterface Module

ObstaclesDisplay

Uses

Obstacle pygame.sprite.Sprite

Syntax

Exported Constants

N/A

Exported Types

ObstaclesDisplay = ?

Exported Access Programs

Routine name	**In**	**Out**	**Exceptions**
new ObstaclesDisplay	pygame window object, pygame sprite group	ObstaclesDisplay	
show	~~$\mathbb{N}$, $\mathbb{N}$, $\mathbb{B}$~~		

Semantics

Environment Variables

screen: It is the game screen that is manipulated by the following functions to alter display. This is update by a precise frame rate to depict various game objects on the game screen.

State Variables

$SCREEN$: *pygame window object*
~~img : Obstacle Picture~~
blocks_group: pygame sprite group

State Invariant

N/A

Assumptions

N/A

Access Routine Semantics

new ObstaclesDisplay($screen$, blocks)

- transition:
  $SCREEN := screen$
  ~~img := Obstacle Picture~~
  blocks_group := blocks
- output: $out := self$
- exception: None

show(~~$x, y, isDestroy$~~):

- transition: ~~$\neg isDestroy \Rightarrow Display\ img\ at\ (x, y)$~~
  blocks_group.draw(SCREEN) # This is pygame API
- output: none
- exception: None

19 AmmoDisplay Module This module has been deleted

UserInterface Module

AmmoDisplay

Uses

Ammo

Syntax

Exported Constants

N/A

Exported Types

AmmoDisplay = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new AmmoDisplay	pygame window object	AmmoDisplay	
show	$\mathbb{N}$, $\mathbb{N}$, $\mathbb{B}$		

Semantics

Environment Variables

screen: It is the game screen that is manipulated by the following functions to alter display. This is update by a precise frame rate to depict various game objects on the game screen.

State Variables

$SCREEN$: pygame window object
img : Picture of Ammo

State Invariant

N/A

Assumptions

N/A

Access Routine Semantics

new AmmoDisplay($screen$)

- transition:
  $SCRREN := screen$
  $img := Ammo\ Picture$
- output: $out := self$
- exception: None

show(x, y, isShot):

- transition: $\neg isShot \Rightarrow Display\ img\ at\ (x, y)$
- input definitions: x, y represent the coordinate of ammo picture to be displayed. *isShot* represents the state of ammo, it is True if the ammo is shoot.
- output: none
- exception: None

20 HeartDisplay Module This module has been deleted

UserInterface Module

HeartDisplay

Uses

Heart

Syntax

Exported Constants

N/A

Exported Types

HeartDisplay = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new HeartDisplay	pygame window object	HeartDisplay	
show	$\mathbb{N}$, $\mathbb{N}$, $\mathbb{B}$		

Semantics

Environment Variables

screen: It is the game screen that is manipulated by the following functions to alter display. This is update by a precise frame rate to depict various game objects on the game screen.

State Variables

$SCREEN$: pygame window object
img : Picture of Heart

State Invariant

N/A

Assumptions

N/A

Access Routine Semantics

new HeartDisplay($screen$)

- transition:
  $SCRREN := screen$
  $img := Heart\ Picture$
- output: $out := self$
- exception: None

show(x, y, isShot):

- transition: $\neg isShot \Rightarrow Display\ img\ at\ (x, y)$
- input definitions: x, y represent the coordinate of Heart picture to be displayed. *isShot* represents the state of Heart, it is True if the Heart is shoot.
- output: none
- exception: None

21 BombDisplay Module This module has been deleted

UserInterface Module

BombDisplay

Uses

Bomb

Syntax

Exported Constants

N/A

Exported Types

BombDisplay = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new BombDisplay	pygame window object	BombDisplay	
show	$\mathbb{N}$, $\mathbb{N}$, $\mathbb{B}$		

Semantics

Environment Variables

screen: It is the game screen that is manipulated by the following functions to alter display. This is update by a precise frame rate to depict various game objects on the game screen.

State Variables

$SCREEN$: pygame window object
img : Picture of Bomb

State Invariant

N/A

Assumptions

N/A

Access Routine Semantics

new BombDisplay(*screen*)

- transition:
  $SCRREN := screen$
  $img := Bomb\ Picture$
- output: $out := self$
- exception: None

show(x, y, isShot):

- transition: $\neg isShot \Rightarrow Display\ img\ at\ (x, y)$
- input definitions: x, y represent the coordinate of Bomb picture to be displayed. *isShot* represents the state of Bomb, it is True if the Bomb is shoot.
- output: none
- exception: None

22 MonsterMatrixDisplay Module

UserInterface Module

MonsterMatrixDisplay

Uses

~~BombDisplay, MonsterDisplay, AmmoDisplay, HeartDisplay,~~ MonsterMatrix, pygame.sprite.Sprite

Syntax

Exported Constants

N/A

Exported Types

MonsterMatrixDisplay = ?

Exported Access Programs

Routine name	In	Out	Excep
new MonsterMatrixDisplay	pygame window object, pygame sprite group	MonsterMatrixDisplay	
show	~~*MonsterMatrix*~~		

Semantics

Environment Variables

screen: It is the game screen that is manipulated by the following functions to alter display. This is update by a precise frame rate to depict various game objects on the game screen.

State Variables

$SCREEN$: pygame window object
monsters_group : pygame sprite group

State Invariant

N/A

Assumptions

N/A

Access Routine Semantics

new MonsterMatrixDisplay($screen$, monsters)

- transition:
  $SCREEN := screen$
  monsters_group := monsters
- output: $out := self$
- exception: None

show(~~M~~):

- transition: ~~$\forall object \in M \mid object.show(x, y, isDead/isShot)$~~
  ~~M here could be $Monster$ $Ammo$ $Heart$ $Bomb$~~
  monsters_group.drawSCREEN # This is pygame API
- output: none
- exception: None

23 WindowSetUp Module

UserInterface Module

~~SetUpDisplay~~ WindowSetUp

Uses

None

Syntax

Exported Constants

N/A

Exported Types

~~SetUpDisplay~~ WindowSetUp $= ?$

Exported Access Programs

Routine name	**In**	**Out**	**Exceptions**
new ~~SetUpDisplay~~ WindowSetUp	$\mathbb{Z}, \mathbb{Z}$	~~SetUpDisplay~~ WindowSetUp	
~~show~~ Method removed			
getScreen		pygame window object	
setTitle			
setIcon			
setBackground			

Semantics

Environment Variables

screen: It is the game screen that is manipulated by the following functions to alter display. This is update by a precise frame rate to depict various game objects on the game screen.

State Variables

$SCREEN$: pygame window object
w: $\mathbb{Z}$
h: $\mathbb{Z}$

State Invariant

N/A

Assumptions

N/A

Access Routine Semantics

new ~~SetUpDisplay~~ WindowSetUp (width, height)

- transition:
  $SCREEN := new\ pygame\ window\ object$
  w := width
  h := height
- output: $out := self$
- exception: None

~~show()~~:

- transition: Display The following contents
  - Welcoming message
  - Display game mode selection
  - Game instruction
  - Prevent game addiction message
- output: none
- exception: None

getScreen()

- transition: none
- output: $out := SCREEN$
- exception: none

setTitle()

- transition: set Title of the game window
- output: none
- exception: none

setIcon()

- transition: set Icon of the game window
- output: none
- exception: none

setBackgroup()

- transition: set background of the game window
- output: none
- exception: none

24 SingleController Module

Template Module

SingleController

Uses

BulletDisplay, MonsterMatrixDisplay, SpaceShipDisplay, ScoreDisplay, ObstaclesDisplay, pygame.sprite.Spite

Syntax

Exported Constants

None

Exported Types

SingleController = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new SingleController	pygame window, pygame screen	SingleController	
run	Keyboard Inputs		
getScore		$\mathbb{Z}$	
doesWin		$\mathbb{B}$	

Semantics

State Variables

All the model objects and corresponding display objects.

State Invariant

None

Assumptions

None

Access Routine Semantics

new SingleController(window, screen):

- transition: Create the model objects and corresponding display objects.
- output: $out := self$
- exception: None

run()

- transition: The controller should do the following things:
  - Let player move space by pressing ← and →.
  - Let player shoot bullet by pressing `SPACE`.
  - If any monster is dead or any game item is shot, let them disappear from the game window.
  - If any monsters are shot, increase the score.
  - If any game items are shot, do corresponding operations.
  - If a round is finished, go to the next round.
  - If the spaceship is shot, decrease spaceship lives.
  - If the obstacle is shot, decrease obstacle areas.
- output: none
- exception: none

getScore()

- transition : none
- output : out := score of this game
- exception: none

doesWin()

- transition : none
- output : out := if the player wins
- exception: none

25 DoubleController Module

Template Module

DoubleController

Uses

BulletDisplay, MonsterMatrixDisplay, SpaceShipDisplay, ScoreDisplay, ObstaclesDisplay, pygame.sprite.Sprite

Syntax

Exported Constants

None

Exported Types

DoubleController = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new DoubleController	pygame window, pygame screen	DoubleController	
run	Keyboard Inputs		
getScore		$\mathbb{Z}$	
doesWin		$\mathbb{B}$	

Semantics

State Variables

All the model objects and corresponding display objects.

State Invariant

None

Assumptions

None

Access Routine Semantics

new DoubleController(window, screen):

- transition: Create the model objects and corresponding display objects.
- output: $out := self$
- exception: None

run()

- transition: The controller should do the following things:
  - Let player1 move space by pressing ← and →.
  - Let player1 shoot bullet by pressing `SPACE`.
  - Let player2 move space by pressing `A` and `D`.
  - Let player2 shoot bullet by pressing `S`.
  - If any monster is dead or any game item is shot, let them disappear from the game window.
  - If any monsters are shot, increase the score.
  - If any game items are shot, do corresponding operations.
  - If a round is finished, go to the next round.
  - If the spaceship is shot, decrease spaceship lives.
  - If the obstacle is shot, decrease obstacle areas.
- output: none
- exception: none

getScore()

- transition : none
- output : out := score of this game
- exception: none

doesWin()

- transition : none
- output : out := if the player wins
- exception: none

26 TotalController Module

Template Module

TotalController

Uses

~~SetUpDisplay~~ WindowSetUp

Syntax

Exported Constants

None

Exported Types

TotalController = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new TotalController		TotalController	
run	Keyboard Input		

Semantics

State Variables

s : WindowSetUp

State Invariant

None

Assumptions

None

Access Routine Semantics

new TotalController():

- transition: $s := newSetUpDisplay()$
- output: $out := self$
- exception: None

run()

- transition:
  $s.run()$
  If user chooses single player model $\Rightarrow$ Invoke SingleController
  If user chooses double player model $\Rightarrow$ Invoke DoubleController
- output: none
- exception: none

27 Driver Module

Template Module

Driver

Uses

TotalController

Syntax

Exported Constants

None

Exported Types

Driver = ?

Exported Access Programs

Routine name	**In**	**Out**	**Exceptions**
new Driver		Driver	
run			

Semantics

State Variables

t : TotalController

State Invariant

None

Assumptions

None

Access Routine Semantics

new Driver():

- transition: $t := new\ TotalController()$
- output: $out := self$
- exception: None

run()

- transition: $t.run()$
- output: none
- exception: none

28 LifeDisplay Module Newly Added module

Template Module

LifeDisplay

Uses

pygame.sprite.Sprite

Syntax

Exported Constants

None

Exported Types

LifeDisplay = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new LifeDisplay	pygame screen	LifeDisplay	
show	$\mathbb{R}, \mathbb{R}, \mathbb{Z}, \mathbb{Z}$		

Semantics

State Variables

$SCREEN$: pygame window object

State Invariant

None

Assumptions

None

Access Routine Semantics

new LifeDisplay(screen):

- transition: $SCREEN := screen$
- output: $out := self$
- exception: None

show(x, y, life, index)

- transition: display life at (x, y) # Index is used to indicate which spaceship's life to be displayed
- output: none
- exception: none

29 WelcomeMessageDisplay Module Newly Added module

Template Module

WelcomeMessageDisplay

Uses

pygame.sprite.Sprite

Syntax

Exported Constants

None

Exported Types

WelcomeMessageDisplay = ?

Exported Access Programs

Routine name	**In**	**Out**	**Exceptions**
new WelcomeMessageDisplay	pygame screen	WelcomeMessageDisplay	

Semantics

State Variables

None

State Invariant

None

Assumptions

None

Access Routine Semantics

new WelcomeMessageDisplay(screen):

- transition: display welcome message at screen
- output: $out := self$
- exception: None

30 ModeSelectionDisplay Module Newly Added module

Template Module

ModeSelectionDisplay

Uses

pygame.sprite.Sprite

Syntax

Exported Constants

None

Exported Types

ModeSelectionDisplay = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new ModeSelectionDisplay	pygame screen	ModeSelectionDisplay	

Semantics

State Variables

None

State Invariant

None

Assumptions

None

Access Routine Semantics

new ModeSelectionDisplay(screen):

- transition: display mode selection message at screen
- output: $out := self$
- exception: None

31 GameInstruction Module Newly Added module

Template Module

GameInstruction

Uses

pygame.sprite.Sprite

Syntax

Exported Constants

None

Exported Types

GameInstruction = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new GameInstruction	pygame screen, game mode	GameInstruction	

Semantics

State Variables

None

State Invariant

None

Assumptions

None

Access Routine Semantics

new GameInstruction(screen, mode):

- transition: display proper game instruction according to mode at screen
- output: $out := self$
- exception: None

32 GameItemIntroductionDisplay Module Newly Added module

Template Module

GameItemIntroductionDisplay

Uses

pygame.sprite.Sprite

Syntax

Exported Constants

None

Exported Types

GameItemIntroductionDisplay = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new GameItemIntroductionDisplay	pygame screen	GameItemIntroductionDisplay	

Semantics

State Variables

None

State Invariant

None

Assumptions

None

Access Routine Semantics

new GameItemIntroductionDisplay(screen):

- transition: display game items introduction on screen
- output: $out := self$
- exception: None

33 LoseDisplay Module Newly Added module

Template Module

LoseDisplay

Uses

pygame.sprite.Sprite

Syntax

Exported Constants

None

Exported Types

LoseDisplay = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new LoseDisplay	pygame screen, $\mathbb{Z}$	LoseDisplay	

Semantics

State Variables

None

State Invariant

None

Assumptions

None

Access Routine Semantics

new LoseDisplay(screen, score):

- transition: display lose message and score on screen
- output: $out := self$
- exception: None

34 WinDisplay Module Newly Added module

Template Module

WinDisplay

Uses

pygame.sprite.Sprite

Syntax

Exported Constants

None

Exported Types

WinDisplay = ?

Exported Access Programs

Routine name	**In**	**Out**	**Exceptions**
new WinDisplay	pygame screen, $\mathbb{Z}$	WinDisplay	

Semantics

State Variables

None

State Invariant

None

Assumptions

None

Access Routine Semantics

new WinDisplay(screen, score):

- transition: display win message and score on screen
- output: $out := self$
- exception: None