EECS 3311-W20 Project Software Design Documentation

Submitted electronically by:

Team members	Name	Prism Login	Signature
Member 1:	Saad Qamer	Saad1996	SaadQamer

Instructor: Jonathan Ostroff
Student ID: 213559638
Course: EECS 3311
Term: W20
Project Name: simOdyssey2
Date: 04/01/2020

Table of Contents

1 Requirements for Project simOdyssey2

1.1 Introduction

Our customer provided us with the a specification and general theoretical construction of their needs for a game called simOdyssey2. The subject playing the game simOdyssey2 experiences a galaxy exploration simulator to prepare a new generation for deep space exploration. The purpose of the game is to provide a simulation to train space explorers to search different sectors of our own galaxy containing stars of the same type as our sun. These stars and known as "Yellow Dwarfs" and are believed to hold the best hope of discovering planets that support life as we know it. This software design document describes the architecture and system design of such a game and displays the constructs needed and intended design decisions for an optimal solution.

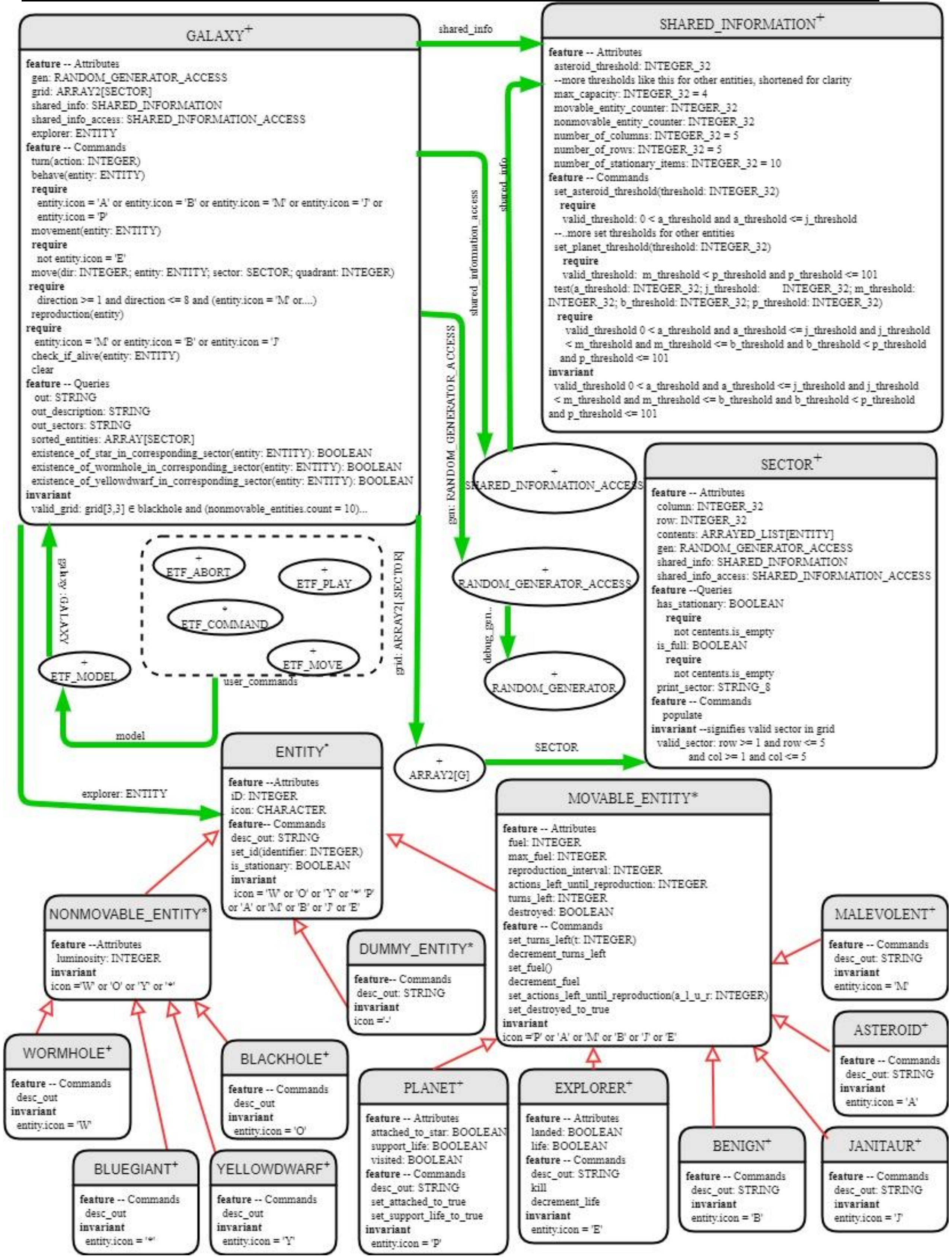
1.2 Scope

The subject plays the role of the explorer in the galaxy and as such is intended to search different sectors of the galaxy containing stars of the same type as our sun known as yellow dwarfs. These stars are believed to hold the best hope of discovering planets that support life, so the goal of the explorer is to see if such stars in the galaxy have any planets orbiting them. If a planet is discovered and is orbiting a yellow dwarf the explorer can land on the planet and determine if life is supportable. The game and the simulation ends when a planet capable of supporting life is discovered.
The galaxy is simulated as a 5 by 5 grid in which a sector is recognized as the coordinates in the grid in terms of row number and column number as shown below.

1,1	1,2	1,3	1,4	1,5
2,1	2,2	2,3	2,4	2,5
3,1	3,2	3,3	3,5	3,5
4,1	4,2	4,3	4,4	4,5
5,1	5,2	5,3	5,4	5,5

The explorer starts the game in sector 1,1 and is able to move to any sector adjacent to it using a compass based movement namely, N, NE, E, SE, S, SW, W and NW. The grid wraps along the boundaries meaning going north from row 1 will end take the explorer to row 5 and likewise for the southern, eastern and western boundaries. Each sector of the galaxy contains 4 quadrants in which entities may or may not be present, the distribution is based on a user inputted threshold for certain entities. There are 2 types of entities, movable and stationary where the movable entities consist of the explorer, malevolents, benigns, planets, janitaurs and asteroids and the stationary entities consists of the blackhole, yellow dwarfs, blue giants and wormholes. The subject is offered the ability as the explorer to move, land on planets orbiting yellow dwarfs, liftoff from said planets, obtain their current status in the simulation and abort the game.

2 System Architecture | 2.1 Architectural Design - (BON Class Diagram)


GALAXY+
feature -- Attributes
gen: RANDOM_GENERATOR_ACCESS
grid: ARRAY2[SECTOR]
shared_info: SHARED_INFORMATION
shared_info_access: SHARED_INFORMATION_ACCESS
explorer: ENTITY
feature -- Commands
turn(action: INTEGER)
behave(entity: ENTITY)
require
entity.icon = 'A' or entity.icon = 'B' or entity.icon = 'M' or entity.icon = 'J' or entity.icon = 'P'
movement(entity: ENTITY)
require
not entity.icon = 'E'
move(dir: INTEGER; entity: ENTITY; sector: SECTOR; quadrant: INTEGER)
require
direction >= 1 and direction <= 8 and (entity.icon = 'M' or....)
reproduction(entity)
require
entity.icon = 'M' or entity.icon = 'B' or entity.icon = 'J'
check_if_alive(entity: ENTITY)
clear
feature -- Queries
out: STRING
out_description: STRING
out_sectors: STRING
sorted_entities: ARRAY[SECTOR]
existence_of_star_in_corresponding_sector(entity: ENTITY): BOOLEAN
existence_of_wormhole_in_corresponding_sector(entity: ENTITY): BOOLEAN
existence_of_yellowdwarf_in_corresponding_sector(entity: ENTITY): BOOLEAN
invariant
valid_grid: grid[3,3] ∈ blackhole and (nonmovable_entities.count = 10)...
shared_info
SHARED_INFORMATION+
feature -- Attributes
asteroid_threshold: INTEGER_32
--more thresholds like this for other entities, shortened for clarity
max_capacity: INTEGER_32 = 4
movable_entity_counter: INTEGER_32
nonmovable_entity_counter: INTEGER_32
number_of_columns: INTEGER_32 = 5
number_of_rows: INTEGER_32 = 5
number_of_stationary_items: INTEGER_32 = 10
feature -- Commands
set_asteroid_threshold(threshold: INTEGER_32)
require
valid_threshold: 0 < a_threshold and a_threshold <= j_threshold
--...more set thresholds for other entities
set_planet_threshold(threshold: INTEGER_32)
require
valid_threshold: m_threshold < p_threshold and p_threshold <= 101
test(a_threshold: INTEGER_32; j_threshold: INTEGER_32; m_threshold: INTEGER_32; b_threshold: INTEGER_32; p_threshold: INTEGER_32)
require
valid_threshold 0 < a_threshold and a_threshold <= j_threshold and j_threshold < m_threshold and m_threshold <= b_threshold and b_threshold < p_threshold and p_threshold <= 101
invariant
valid_threshold 0 < a_threshold and a_threshold <= j_threshold and j_threshold < m_threshold and m_threshold <= b_threshold and b_threshold < p_threshold and p_threshold <= 101
shared_information_access
shared_info
gen: RANDOM_GENERATOR_ACCESS
SHARED_INFORMATION_ACCESS
SECTOR+
feature -- Attributes
column: INTEGER_32
row: INTEGER_32
contents: ARRAYED_LIST[ENTITY]
gen: RANDOM_GENERATOR_ACCESS
shared_info: SHARED_INFORMATION
shared_info_access: SHARED_INFORMATION_ACCESS
feature --Queries
has_stationary: BOOLEAN
require
not contents.is_empty
is_full: BOOLEAN
require
not contents.is_empty
print_sector: STRING_8
feature -- Commands
populate
invariant --signifies valid sector in grid
valid_sector: row >= 1 and row <= 5 and col >= 1 and col <= 5
RANDOM_GENERATOR_ACCESS
debug_gen...
RANDOM_GENERATOR
galaxy: GALAXY
ETF_ABORT
ETF_PLAY
ETF_COMMAND
ETF_MOVE
ETF_MODEL
user_commands
model
grid: ARRAY2[SECTOR]
ARRAY2[G]
SECTOR
explorer: ENTITY
ENTITY*
feature --Attributes
iD: INTEGER
icon: CHARACTER
feature-- Commands
desc_out: STRING
set_id(identifier: INTEGER)
is_stationary: BOOLEAN
invariant
icon = 'W' or 'O' or 'Y' or '*' 'P' or 'A' or 'M' or 'B' or 'J' or 'E'
MOVABLE_ENTITY*
feature -- Attributes
fuel: INTEGER
max_fuel: INTEGER
reproduction_interval: INTEGER
actions_left_until_reproduction: INTEGER
turns_left: INTEGER
destroyed: BOOLEAN
feature -- Commands
set_turns_left(t: INTEGER)
decrement_turns_left
set_fuel()
decrement_fuel
set_actions_left_until_reproduction(a_l_u_r: INTEGER)
set_destroyed_to_true
invariant
icon ='P' or 'A' or 'M' or 'B' or 'J' or 'E'
NONMOVABLE_ENTITY*
feature --Attributes
luminosity: INTEGER
invariant
icon ='W' or 'O' or 'Y' or '*'
DUMMY_ENTITY*
feature-- Commands
desc_out: STRING
invariant
icon ='-'
MALEVOLENT+
feature -- Commands
desc_out: STRING
invariant
entity.icon = 'M'
ASTEROID+
feature -- Commands
desc_out: STRING
invariant
entity.icon = 'A'
WORMHOLE+
feature -- Commands
desc_out
invariant
entity.icon = 'W'
BLACKHOLE+
feature -- Commands
desc_out
invariant
entity.icon = 'O'
PLANET+
feature -- Attributes
attached_to_star: BOOLEAN
support_life: BOOLEAN
visited: BOOLEAN
feature -- Commands
desc_out: STRING
set_attached_to_true
set_support_life_to_true
invariant
entity.icon = 'P'
EXPLORER+
feature -- Attributes
landed: BOOLEAN
life: BOOLEAN
feature -- Commands
desc_out: STRING
kill
decrement_life
invariant
entity.icon = 'E'
BENIGN+
feature -- Commands
desc_out: STRING
invariant
entity.icon = 'B'
JANITAUR+
feature -- Commands
desc_out: STRING
invariant
entity.icon = 'J'
BLUEGIANT+
feature -- Commands
desc_out
invariant
entity.icon = '*'
YELLOWDWARF+
feature -- Commands
desc_out
invariant
entity.icon = 'Y'

2 System Architecture | 2.2 Design and Decomposition Description

The design for simOdyssey2 features several decisions employing design patterns, inheritance relationships as well as abstractions and information hiding.

2.2.1 Main Class Relationships

The design for simOdyssey2 features multiple different classes with specific features, attributes, commands, queries and purposes. The design for simOdyssey2 as seen in the BON diagram uses a galaxy class to provide the main control over the galaxy construction and manipulation. The galaxy class is a client of the ARRAY2 supplier in that it uses a two dimensional array for the construction of the galaxy as a grid. The sector class is a client of the ARRAYED_LIST supplier in order to access its elements known as quadrants. The ARRAY2 grid used in the galaxy class is of type SECTOR that uses the ARRAYED_LIST as their underlying data structures. The galaxy class is also a client of the entity supplier class relating to all the different types of entities in the galaxy.

2.2.2 Decomposition Description

The ENTITY class is the highest ancestor of all of it's child classes in the hierarchical relationship. The ENTITY is deferred and is parent to 2 main subclasses in the MOVABLE_ENTITY class and the NONMOVABLE_ENTITY class which are both also deferred. Both classes inherit attributes from the ENTITY class such as the iD and icon and add to it their own features as shown in the BON class diagram. The MOVABLE_ENTITY class is parent to several subclasses including the EXPLORER, MALEVOLENT, BENIGN, PLANET, JANITAUR and ASTEROID classes. Each class inherits certain features and adds certain functionality in terms of commands that pertain to the specific object that it is creating a logical abstraction of. For example, planets have the attribute of being attached whereas others do not and every class in the movable entities has the attribute of being able to destroyed by some other entity. The NONMOVABLE_ENTITY class is parent to several subclasses including the BLACKHOLE, YELLOWDWARF, BLUEGIANT and WORMHOLE classes. Similar to the other case the subclasses add certain features that are intrinsic to that class or they may not, for example the blue giant and yellow dwarf have a luminosity feature that is intrinsic to those 2 as they are stars. The overarching classes that construct the user input and use the ETF framework are the ETF_ABORT, ETF_MOVE, ETF_STATUS, ETF_PLAY, ETF_TEST, ETF_WORMHOLE, ETF_LAND, ETF_LIFTOFF and ETF_COMMAND classes that are a client of the ETF_MODEL supplier classes. The ETF_MODEL class is a client to the GALAXY supplier class that controls the rest of the functionality of the game.

2.2.3 Decomposition Rationale

The rationale for selecting the architecture shown in 2.1 and described in detail in 2.2.1 and 2.2.2 is to obey the concepts of the single choice principle and to maintain a well constructed design. The architecture chosen uses the singleton design pattern for the SHARED_INFORMATION classes and allows for extendibility by using the movable, nonmovable and entity hierarchy.

3 Table of Modules

1	GALAXY	**Responsibility:** Maintain a grid of entities and support galaxy functionality	**Alternative:** Implement with 5 arrays to represent 5 rows of the galaxy grid
	Abstract	**Secret:** implemented using ARRAY2[SECTOR] as grid for galaxy	

1.1	ARRAY2[G]	**Responsibility:** see GALAXY	**Alternative:** none
	Concrete	**Secret:** implemented via sectors that hold an arrayed list of quadrants for entities to be placed	

2	SECTOR	**Responsibility:** Maintain 4 element arrayed_list of quadrants for entities to be placed	**Alternative:** Implement using a linked list of quadrants to allow easier removal and addition of entities
	Abstract	**Secret:** implemented using ARRAYED_LIST[ENTITY] structure for quadrants	

2.1	ARRAYED_LIST[G]	**Responsibility:** see SECTOR	**Alternative:** none
	Concrete	**Secret:** implemented via an entity class that stores information about entities in the galaxy	

3	ENTITY	**Responsibility:** Maintain information about an entity in the galaxy	**Alternative:** none
	Abstract	**Secret:** none	

3.1	MOVABLE_ENTITY	**Responsibility:** Maintain information about specifically a movable entity	**Alternative:** none
	Abstract	**Secret:** none	

3.1.1	EXPLORER	**Responsibility:** Maintain a grid of entities and support galaxy functionality	**Alternative:** none
	Concrete	**Secret:** none	

3.1.2	PLANET	**Responsibility:** Maintain a grid of entities and support galaxy functionality	**Alternative:** none
	Concrete	**Secret:** none	

3.1.3	MALEVOLENT	**Responsibility:** Maintain a grid of entities and support galaxy functionality	**Alternative:** none
	Concrete	**Secret:** none	

3.1.4	JANITAUR	**Responsibility:** Maintain a grid of entities and support galaxy functionality	**Alternative:** none
	Concrete	**Secret:** none	

3.1.5	BENIGN	**Responsibility:** Maintain a grid of entities and support galaxy functionality	**Alternative:** none
	Concrete	**Secret:** none	

3.1.6	ASTEROID	**Responsibility:** Maintain a grid of entities and support galaxy functionality	**Alternative:**
	Concrete	**Secret:** none	

3.2	NONMOVABLE_ENTITY	**Responsibility:** Maintain a grid of entities and support galaxy functionality	**Alternative:** none
	Abstract	**Secret:** none	

3.2.1	BLUEGIANT	**Responsibility:** Maintain a grid of entities and support galaxy functionality	**Alternative:** none
	Concrete	**Secret:** none	

3.2.2	YELLOWDWARF	**Responsibility:** Maintain a grid of entities and support galaxy functionality	**Alternative:** none
	Concrete	**Secret:** none	

3.2.3	WORMHOLE	**Responsibility:** Maintain a grid of entities and support galaxy functionality	**Alternative:** none
	Concrete	**Secret:** none	

3.2.4	BLACKHOLE	**Responsibility:** Maintain a grid of entities and support galaxy functionality	**Alternative:** none
	Concrete	**Secret:** none	

4 Expanded Description of Design Decisions

Galaxy module chosen for documentation

4.1 Description

The most important module in this design for the simOdyssey2 project is the galaxy class as it is the main controller for all the functionality of the galaxy and is a client of many other supplier classes. The galaxy class features many attributes, commands and queries specific to the behaviour of the galaxy in the project.

4.2 Sub-systems and Modules

The galaxy has a grid feature that uses the underlying data structure and is a client of the ARRAY2[G] supplier, more formally a two-dimensional array. Each element in the two dimensional array is of type SECTOR which features an ARRAYED_LIST of quadrants at a maximum of four pertaining to the 4 possible entities that might be present in that sector of the grid. The galaxy class is also a client of the RANDOM_GENERATOR_ACCESS class to be used for several commands that control the way the board is being manipulated based on the user input. The galaxy class is also a client of the SHARED_INFO and SHARED_INFO_ACCESS classes that employ the singleton design pattern so that when the galaxy is made the shared information can only be made and should only be made once to access features such as the max number of rows and columns in the grid as well as several other attributes.

4.3 Relationship to Rest of Design

The galaxy class is the main controller of all the functionality of the galaxy and is a client of many classes as such. The galaxy relates to the rest of the design by using other objects such as ENTITY in the commands and queries to access different elements in the galaxy grid and provide specific behaviour for those entities. The galaxy class has a command called turn that is the main action that takes place when a user inputs a command via the command line. The galaxy class also has a command for the movement of different entities, a command for the behaviour of specific entities, the reproduction of different entities that adhere to the requirements of the program. The galaxy class is a supplier to the ETF_MODEL class as that class controls the user input and determines how the user will be acting by supplying different commands and queries that fit the needs of the specific instruction that the ETF_MODEL class is trying to execute based on the user input. One of the main functions of the galaxy class is to be used for the board creation and as such has many commands that adhere to the requirements of the specific way the board is to be constructed. The galaxy class has the commands of setting stationary items, creating stationary items as well as a clear feature to wipe out the galaxy in the scenario that the user loses. The galaxy class is at the epicentre of the design and relates to the rest of the design in a direct way.

4.4 Design Decisions

The galaxy class has a feature for the access of the shared information that uses the singleton design pattern. This design was chosen so that the creation of the galaxy has only one instance of the shared information a provides a global point of access to that instance. As a result there can only be at most one active instance of the shared information including the constants that the galaxy needs for its construction, such as number of rows, column, etc.

5 Contracts for Galaxy Class

Galaxy class chosen at it contains the most significant contracts

Constructor
make(a_thresh; j_thresh; m_thresh; b_thresh; p_thresh)
require
  a_thresh <= j_thresh <= m_thresh <= benign_thresh <= p_thresh
Description: The contract for the constructor requires that the input for the threshold of specific movable entities be less than the successive movable entity. The threshold for asteroids in the galaxy must be less than or equal to that of the janitaurs, which must be less than or equal to that of the malevolents, which must be less than or equal to that of the benigns, which must be less than or equal to that of the planets. The significance of this condition is relative to the specification of the requirements of the game and as such should the input must adhere to those requirements

Commands
turn(action: STRING)
require
  action ~ move or action ~ pass or action ~ land or action ~ liftoff action ~ abort…
Description: Here the command that constitutes a turn has a precondition that requires that in order for this turn to occur the command entered must be one that constitutes a turn as some functionality is handled through ETF_MODEL. Therefore the turn action will only accept valid inputs. The significance of this contract is to only allow valid commands to be able to constitute a movement without having to implement it explicitly.
----
wormhole(entity: ENTITY)
require
  entity.icon = 'E' or entity.icon ='M' or entity.icon = 'B'
Description: Here the command for wormhole has a precondition that requires that the entity to go through a wormhole must be an explorer or a malevolent or a benign. The significance of this requirement is relative to the requirements put in place by the customer and therefore the implementation must adhere to that
----
movement(entity: ENTITY)
require
 entity.icon = 'M' or entity.icon = 'B' or entity.icon = 'P' or entity.icon = 'A' or entity.icon ='J'
Description: The command for movement has the precondition requirement that only entities that are either a malevolent a benign or janitaur or planet or asteroid may experience random movement. The significance of this is relative to the constructs of the functionality of the game provided by the customer.
----

check_if_alive(entity: ENTITY)
require
entity.icon = ‘M’ or entity.icon = ‘B’ or entity.icon = ‘J’ or entity.icon = ‘A’ or entity.icon = ‘P’
Description: The command for check checks if the entity is alive and the precondition requires that the entity must be an entity that is capable of dying, which based on the specification by the customer includes all the entities that are not stationary.
----
reproduce(entity: ENTITY)
require
entity.icon = ‘M’ or entity.icon = ‘B’ or entity.icon = ‘J’
Description: The command for reproduce has the precondition requirement that only entities that are either a malevolent a benign or janitaur are capable of reproduction. The significance of this contract is relative to the specification of the customer on how the entities are to reproduce.
----
behave(entity: ENTITY)
require
entity.icon = ‘M’ or entity.icon = ‘B’ or entity.icon = ‘P’ or entity.icon = ‘A’ or entity.icon ='J’
Description: The command for behave has the precondition requirement that only entities that are either a malevolent a benign or janitaur or planet or asteroid may experience specific behaviour. The significance of this contract is to ensure that only valid items are able to experience behaviour as per specification of the program by the customer
----
move(dir: INTEGER; entity: ENTITY; sector: SECTOR; quadrant: INTEGER)
require
entity.icon = ‘M’ or entity.icon = ‘B’ or entity.icon = ‘P’ or entity.icon = ‘A’ or entity.icon ='J’or entity.icon = ‘E’ and not sector.is_full
Description: The command for move has the precondition requirement that only entities that are capable of movement are able to move in the board and that the destination sector of said entity is not currently at full capacity. The significance of this requirement is so that the movement of an entity will not cause inappropriate behaviour on a sector such as having more than 4 entities.
----
put_entity_in_next_avail_quadrant(sector: SECTOR; entity: ENTITY)
require
entity.icon = ‘M’ or entity.icon = ‘B’ or entity.icon = ‘P’ or entity.icon = ‘A’ or entity.icon ='J’or entity.icon = ‘E’ and not sector.is_full
Description: This command puts an entity into the next available quadrant of the sector that is provided as its input, it requires that the entity is movable and that the sector that it is going to is not full. The significance of the requirement is to adhere to the requirements of the customer and make sure that a sector does not have more than 4 entities.
----

6 Testing

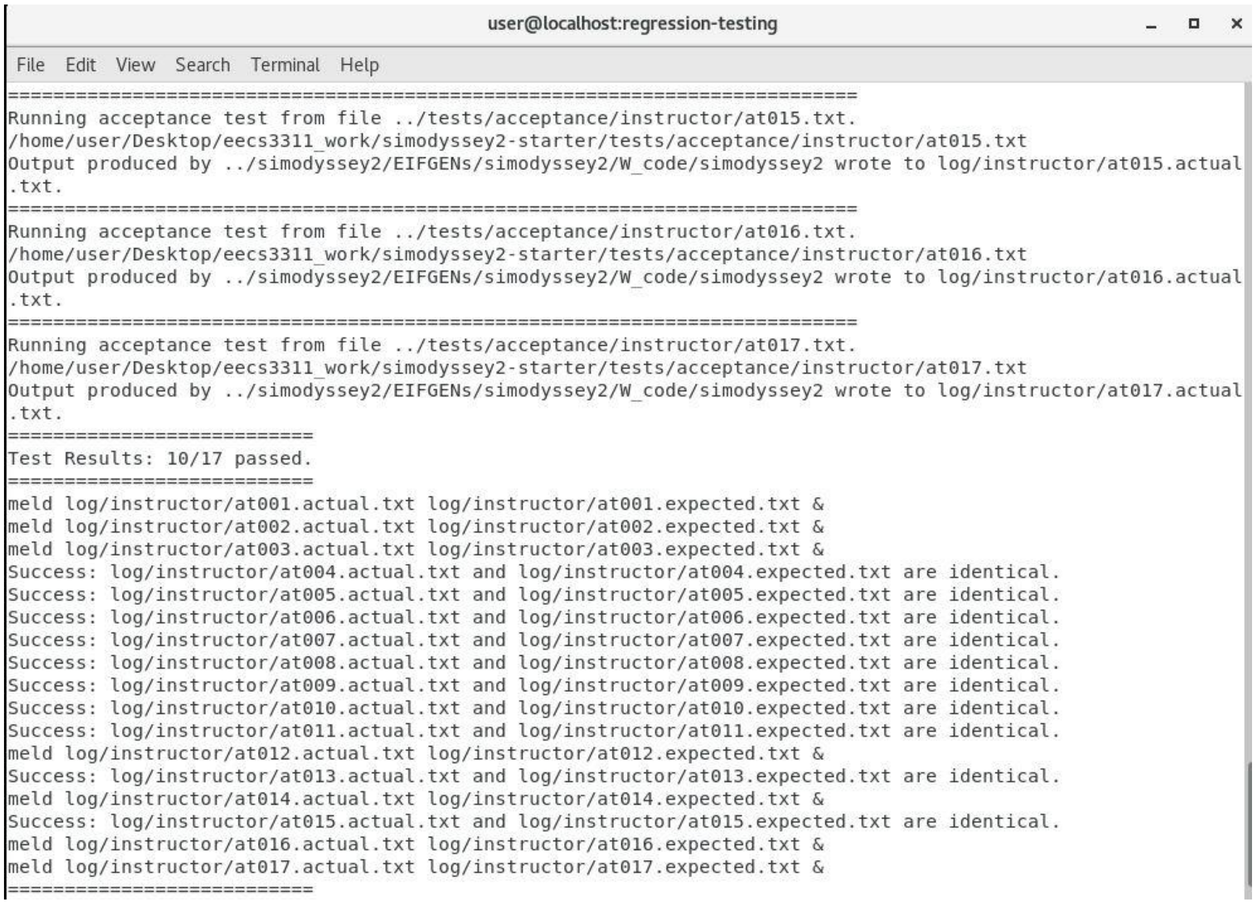
Instructor Tests

Test file	**Description**	**Passed**
At001.txt	Tests a winning condition in test mode.	Failed
At002.txt	Tests a winning condition in play mode.	Failed
At003.txt	Tests losing condition in test mode (lose by out of life)	Failed

Student Tests

Test file	**Description**	**Passed**
At004.txt	Tests basic construction of board to follow random number generation accurately	Passed
At005.txt	Tests basic construction of board to follow random number generation accurately with different inputs	Passed
At006.txt	Tests basic construction of board to follow random number generation accurately with different inputs	Passed
At007.txt	Tests basic construction of board to follow random number generation accurately with different inputs	Passed
At008.txt	Tests basic construction of board to follow random number generation accurately with aborted game condition	Passed
At009.txt	Tests basic construction of board to follow random number generation accurately different input with abort condition if user aborted game	Passed
At010.txt	Tests basic construction of board to follow random number generation accurately with abort and pass condition	Passed
At011.txt	Testing basic construction of board to follow random number generation accurately with addition pass condition from previous	Passed
At012.txt	Testing basic construction of board to follow random number generation accurately with addition pass condition from previous	Failed
At013.txt	Tests basic construction of board to follow random number generation accurately with movement of player	Failed
At014.txt	Tests basic construction of board to follow random number generation accurately with wormhole of player	Failed
At015.txt	Tests basic construction of board to follow random number generation accurately with status of player	Failed
At016.txt	Tests basic construction of board to follow random number generation accurately with landing of player	Failed
At017.txt	Tests basic construction of board to follow random number generation accurately with liftoff of player	Failed

6 Testing (Regression Testing Screenshots)


user@localhost:regression-testing
File Edit View Search Terminal Help

```
===========================================================================
Running acceptance test from file ../tests/acceptance/instructor/at015.txt.
/home/user/Desktop/eecs3311_work/simodyssey2-starter/tests/acceptance/instructor/at015.txt
Output produced by ../simodyssey2/EIFGENs/simodyssey2/W_code/simodyssey2 wrote to log/instructor/at015.actual
.txt.
===========================================================================
Running acceptance test from file ../tests/acceptance/instructor/at016.txt.
/home/user/Desktop/eecs3311_work/simodyssey2-starter/tests/acceptance/instructor/at016.txt
Output produced by ../simodyssey2/EIFGENs/simodyssey2/W_code/simodyssey2 wrote to log/instructor/at016.actual
.txt.
===========================================================================
Running acceptance test from file ../tests/acceptance/instructor/at017.txt.
/home/user/Desktop/eecs3311_work/simodyssey2-starter/tests/acceptance/instructor/at017.txt
Output produced by ../simodyssey2/EIFGENs/simodyssey2/W_code/simodyssey2 wrote to log/instructor/at017.actual
.txt.
===========================
Test Results: 10/17 passed.
===========================
meld log/instructor/at001.actual.txt log/instructor/at001.expected.txt &
meld log/instructor/at002.actual.txt log/instructor/at002.expected.txt &
meld log/instructor/at003.actual.txt log/instructor/at003.expected.txt &
Success: log/instructor/at004.actual.txt and log/instructor/at004.expected.txt are identical.
Success: log/instructor/at005.actual.txt and log/instructor/at005.expected.txt are identical.
Success: log/instructor/at006.actual.txt and log/instructor/at006.expected.txt are identical.
Success: log/instructor/at007.actual.txt and log/instructor/at007.expected.txt are identical.
Success: log/instructor/at008.actual.txt and log/instructor/at008.expected.txt are identical.
Success: log/instructor/at009.actual.txt and log/instructor/at009.expected.txt are identical.
Success: log/instructor/at010.actual.txt and log/instructor/at010.expected.txt are identical.
Success: log/instructor/at011.actual.txt and log/instructor/at011.expected.txt are identical.
meld log/instructor/at012.actual.txt log/instructor/at012.expected.txt &
Success: log/instructor/at013.actual.txt and log/instructor/at013.expected.txt are identical.
meld log/instructor/at014.actual.txt log/instructor/at014.expected.txt &
Success: log/instructor/at015.actual.txt and log/instructor/at015.expected.txt are identical.
meld log/instructor/at016.actual.txt log/instructor/at016.expected.txt &
meld log/instructor/at017.actual.txt log/instructor/at017.expected.txt &
===========================
```

Screenshot of regression testing done on program through linux command line.

7 Appendix (Contract View of All Classes Mentioned)

Galaxy Class

```
-- Automatic generation produced by ISE Eiffel --
note
	description: "Galaxy represents a game board in simodyssey."
	author: "Kevin B"
	date: "$Date$"
	revision: "$Revision$"

class interface
	GALAXY

create
	make,
	make_dummy

feature -- attributes
```

```
        grid: ARRAY2 [SECTOR]
                        -- the board

        gen: RANDOM_GENERATOR_ACCESS

        shared_info_access: SHARED_INFORMATION_ACCESS

        shared_info: SHARED_INFORMATION

        movement_has_occured: BOOLEAN

        movement_string: STRING_8

        deaths_this_turn_flag: BOOLEAN

        death_string: STRING_8

        dead_entities: ARRAY [ENTITY]

        explorer_landed_on_planet_supports_life: BOOLEAN

feature --constructor

        make (a_thresh: INTEGER_32; j_thresh: INTEGER_32; m_thresh: INTEGER_32...)
                        -- creates a dummy of galaxy grid

        make_dummy

feature --commands

        set_deaths_this_turn_flag_false

        set_movement_has_occured_false

        clear

        this_is_the_explorer: EXPLORER

        turn (action: INTEGER_32): STRING_8
                        --RENAME THIS TO TURN and make a pass condition

        check_if_alive (entity: MOVABLE_ENTITY)
                        --check(entity)

        reproduce (entity: MOVABLE_ENTITY): STRING_8

        put_entity_in_next_avail_quadrant2 (sector: SECTOR; entity: ENTITY)
                        --this is only for reproduce to use

        out_movement: STRING_8

        out_deaths_this_turn: STRING_8

        out_entity_exact_location (entity: ENTITY): STRING_8

        sector_of_entity_print (entity: ENTITY): STRING_8

        get_rid_of_mentity_from_board (entity: ENTITY)

        behave (entity: MOVABLE_ENTITY): STRING_8

        movement (entity: ENTITY): STRING_8

        find_and_move_explorer (direction: INTEGER_32): STRING_8

        move (dir: INTEGER_32; entity: ENTITY; sector: SECTOR; quadrant: INTEGER_32): STRING_8

        put_entity_in_next_avail_quadrant (sector: SECTOR; entity: ENTITY): STRING_8

        wormhole (entity: ENTITY): STRING_8
```

```
	put_wormholed_entity_in_next_avail_quadrant (sector: SECTOR; entity: ENTITY): STRING_8

	set_stationary_items
			-- distribute stationary items amongst the sectors in the grid.
			-- There can be only one stationary item in a sector

	create_stationary_item: NONMOVABLE_ENTITY
		-- this feature randomly creates one of the possible types of stationary actors

feature -- query

	existence_of_wormhole_in_corresponding_sector (entity: ENTITY): BOOLEAN
			--check if there is a wormhole in the sector associated with this entity

	existence_of_yellowdwarf_in_corresponding_sector (entity: ENTITY): BOOLEAN
			--check if there is a yellowdwarf in the sector associated with this
entity

	existence_of_star_in_corresponding_sector (entity: ENTITY): BOOLEAN
			--check if there is a star in the sector associated with this entity

	out_sectors: STRING_8

	sorted_entities: ARRAY [ENTITY]

	out_description: STRING_8

	out: STRING_8
			--Returns grid in string form

end -- class GALAXY
			-- Generated by ISE Eiffel --
			-- For more details: http://www.eiffel.com --
```

Sector Class

```
-- Automatic generation produced by ISE Eiffel --
note
	description: "Represents a sector in the galaxy."
	author: ""
	date: "$Date$"
	revision: "$Revision$"

class interface
	SECTOR

create
	make,
	make_dummy

feature -- attributes

	shared_info_access: SHARED_INFORMATION_ACCESS

	shared_info: SHARED_INFORMATION

	gen: RANDOM_GENERATOR_ACCESS

	contents: ARRAYED_LIST [ENTITY]
			--holds 4 quadrants

	row: INTEGER_32

	column: INTEGER_32

feature -- constructor
```

```
	make (row_input: INTEGER_32; column_input: INTEGER_32; a_explorer: ENTITY)
			--initialization
		require
			valid_row: (row_input >= 1) and (row_input <= shared_info.Number_rows)
			valid_column: (column_input >= 1) and (column_input <= sh...

feature -- commands

	make_dummy
			--initialization without creating entities in quadrants

	populate
			-- this feature creates 1 to max_capacity-1 components to be intially 
stored in the
			-- sector. The component may be a planet or nothing at all.

feature -- Queries

	print_sector: STRING_8
			-- Printable version of location's coordinates with different formatting

	is_full: BOOLEAN
			-- Is the location currently full?

	has_stationary: BOOLEAN
			-- returns whether the location contains any stationary item

end -- class SECTOR
			-- Generated by ISE Eiffel --
			-- For more details: http://www.eiffel.com --
```

Entity Class

```
-- Automatic generation produced by ISE Eiffel --
note
	description: "Summary description for {ENTITY}."
	author: ""
	date: "$Date$"
	revision: "$Revision$"

deferred class interface
	ENTITY

feature --Attributes

	id: INTEGER_32

	icon: CHARACTER_8

feature --Query

	is_stationary: BOOLEAN
			-- Return if current item is stationary.

	set_id (identifier: INTEGER_32)

	desc_out: STRING_8

end -- class ENTITY
			-- Generated by ISE Eiffel --
			-- For more details: http://www.eiffel.com --
```

Movable Entity Class

```
-- Automatic generation produced by ISE Eiffel --
note
	description: "Summary description for {MOVABLE_ENTITY}."
	author: ""
	date: "$Date$"
	revision: "$Revision$"

deferred class interface
	MOVABLE_ENTITY

feature --Attributes

	is_fueled: BOOLEAN
			--flag to check if movable entity requires fuel or not

	death_message: STRING_8

	max_fuel: INTEGER_32

	fuel: INTEGER_32

	reproduction_interval: INTEGER_32

	turns_left: INTEGER_32

	actions_left_until_reproduction: INTEGER_32

	destroyed: BOOLEAN

	set_turns_left (t: INTEGER_32)

	decrement_turns_left

	set_dstroyed_to_true

	set_actions_left_until_reproduction (a_l_u_r: INTEGER_32)

	decrement_actions_left_until_rep

	decrement_fuel

	set_fuel (fuel_input: INTEGER_32)

end -- class MOVABLE_ENTITY
			-- Generated by ISE Eiffel --
			-- For more details: http://www.eiffel.com --
```

Explorer Class

```
		-- Automatic generation produced by ISE Eiffel --
note
	description: "Summary description for {EXPLORER}."
	author: ""
	date: "$Date$"
	revision: "$Revision$"

class interface
	EXPLORER

create
	make

feature -- Attributes

	landed: BOOLEAN

	life: INTEGER_32
```

```
feature --Constructor

	make

feature -- COmmand

	desc_out: STRING_8

	kill

	decrement_life

end -- class EXPLORER
			-- Generated by ISE Eiffel --
			-- For more details: http://www.eiffel.com --
```

Malevolent Class

```
		-- Automatic generation produced by ISE Eiffel --
note
	description: "Summary description for {MALEVOLENT}."
	author: ""
	date: "$Date$"
	revision: "$Revision$"

class interface
	MALEVOLENT

create
	make

feature --Constructor

	make

feature -- Commands

	desc_out: STRING_8

end -- class MALEVOLENT
			-- Generated by ISE Eiffel --
			-- For more details: http://www.eiffel.com --
```

Benign Class

```
	-- Automatic generation produced by ISE Eiffel --
note
	description: "Summary description for {BENIGN}."
	author: ""
	date: "$Date$"
	revision: "$Revision$"

class interface
	BENIGN

create
	make

feature --Constructor

	make

feature -- Commands

	desc_out: STRING_8
```

```
end -- class BENIGN
                       -- Generated by ISE Eiffel --
                       -- For more details: http://www.eiffel.com --
```

Janitaur Class

```
-- Automatic generation produced by ISE Eiffel --
note
	description: "Summary description for {JANITAUR}."
	author: ""
	date: "$Date$"
	revision: "$Revision$"

class interface
	JANITAUR

create
	make

feature --Constructor

	make

feature --Attributes

	load: INTEGER_32

	Max_load: INTEGER_32 = 2

feature -- Commands

	desc_out: STRING_8

	increment_load

	set_load (load_input: INTEGER_32)

end -- class JANITAUR
                       -- Generated by ISE Eiffel --
                       -- For more details: http://www.eiffel.com --
```

Asteroid Class

```
-- Automatic generation produced by ISE Eiffel --
note
	description: "Summary description for {ASTEROID}."
	author: ""
	date: "$Date$"
	revision: "$Revision$"

class interface
	ASTEROID

create
	make

feature --Constructor

	make

feature --Commands

	desc_out: STRING_8
```

```
end -- class ASTEROID
                       -- Generated by ISE Eiffel --
                       -- For more details: http://www.eiffel.com --
```

Planet Class

```
-- Automatic generation produced by ISE Eiffel --
note
	description: "Summary description for {PLANET}."
	author: ""
	date: "$Date$"
	revision: "$Revision$"

class interface
	PLANET

create
	make

feature --Constructor

	make

feature -- Attributes

	attached_to_star: BOOLEAN

	support_life: BOOLEAN

	visited: BOOLEAN

feature -- COmmands

	desc_out: STRING_8

	set_attached_to_true

	set_support_life_to_true

end -- class PLANET
                       -- Generated by ISE Eiffel --
                       -- For more details: http://www.eiffel.com --
```

Nonmovable Entity Class

```
-- Automatic generation produced by ISE Eiffel --
note
	description: "Summary description for {NONMOVABLE_ENTITY}."
	author: ""
	date: "$Date$"
	revision: "$Revision$"

deferred class interface
	NONMOVABLE_ENTITY

feature --Attributes

	luminosity: INTEGER_32

end -- class NONMOVABLE_ENTITY
                       -- Generated by ISE Eiffel --
                       -- For more details: http://www.eiffel.com --
```

Blue Giant Class

```
-- Automatic generation produced by ISE Eiffel --
note
	description: "Summary description for {BLUEGIANT}."
	author: ""
	date: "$Date$"
	revision: "$Revision$"

class interface
	BLUEGIANT

create
	make

feature --Constructor

	make

feature --Commands

	desc_out: STRING_8

end -- class BLUEGIANT
                      -- Generated by ISE Eiffel --
                      -- For more details: http://www.eiffel.com --
```

Yellow Dwarf Class

```
-- Automatic generation produced by ISE Eiffel --
note
	description: "Summary description for {YELLOWDWARF}."
	author: ""
	date: "$Date$"
	revision: "$Revision$"

class interface
	YELLOWDWARF

create
	make

feature --Constructor

	make

feature --Commands

	desc_out: STRING_8

end -- class YELLOWDWARF
                      -- Generated by ISE Eiffel --
                      -- For more details: http://www.eiffel.com -
```

Wormhole Class

```
-- Automatic generation produced by ISE Eiffel --
note
	description: "Summary description for {WORMHOLE}."
	author: ""
	date: "$Date$"
	revision: "$Revision$"

class interface
	WORMHOLE

create
	make
```

```
feature --Constructor

	make

feature --Commands

	desc_out: STRING_8

end -- class WORMHOLE
                       -- Generated by ISE Eiffel --
                       -- For more details: http://www.eiffel.com --
```

Blackhole Class

```
-- Automatic generation produced by ISE Eiffel --
note
	description: "Summary description for {BLACKHOLE}."
	author: ""
	date: "$Date$"
	revision: "$Revision$"

class interface
	BLACKHOLE

create
	make

feature --Constructor

	make

feature --Commands

	desc_out: STRING_8

end -- class BLACKHOLE
                       -- Generated by ISE Eiffel --
                       -- For more details: http://www.eiffel.com --
```
