SIMODYSEEY — Project Report

Submitted electronically by:

|  |  |  |  |
| --- | --- | --- | --- |
| **Team members** | **Name** | **Prism Login** | **Signature** |
| Member 1: |  |  |  |
| Member 2: |  |  |  |
| \*Submitted under Prism account: | |  |  |

**Contents**

[1.](#_30j0zll) Requirements for simodyseey 2

[2.](#_1fob9te) BON class diagram overview (architecture of the design) 7

[3.](#_3znysh7) Table of modules — responsibilities and information hiding 8

[4.](#_2et92p0) Expanded description of design decisions 9

[5.](#_tyjcwt) Significant Contracts (Correctness) 10

[6.](#_3dy6vkm) Summary of Testing Procedures 11

[7.](#_1t3h5sf) Appendix (Contract view of all classes) 12

# Requirements for Project - Simodyssey

Our clients supplied us with the following requirements for the game Simodyssey: The topic at hand is game development. This space exploration game will allow a player to choose between two playing modes. The first mode is referred to as “test”, this mode provides a more detailed outlook on the current state of the galaxy and as well as, the amount and type of entities generated. This entity creation can be influenced by the number that is passed when the test mode is created. The higher the number implies that the probability of generating more planets increases. The second mode is referred to as “play” , this mode will provide less information about the current state of the galaxy and the number of entities that are initially created cannot be influenced.

Within both modes, random numbers are generated in a sequential manner that is always the same and maintained, but changes often from “play” to “play” mode. After a game has started, a 5-by-5 galaxy is created. This would also place an explorer entity at quadrant (1,1) and a blackhole entity at (3,3). The other entities that are stationary and movable are randomly placed in the galaxy. Moreover, these entities are placed in sectors within the galaxy; each sector can contain a maximum of four quadrants. This means that each sector can only hold a maximum of four entities ( one entity per quadrant).

After the galaxy is created, the player would be able to issue commands to move the explorer through the galaxy. Some of these commands would be considered as a turn, which may then cause some of the movable entities to shift their position in the galaxy. However, other commands would not modify the entities within the galaxy but may just display the status of the explorer. The game can end in a few ways, such as; when the explorer runs out of fuel, lives, or when a planet with life is found or when the game is aborted. Once a game has ended a new game can be started.

The document in the appendix will provide further details on the user interface grammar. Also, the acceptance tests in the report will provide further details on the input and output status of the game operating in console mode.

# BON class diagram overview (architecture of the design)

# Table of modules — responsibilities and information hiding

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | LIST[G] | **Responsibility**: a sequence of items of type G | **Alternative**: see ARRAY[G] |
|  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | LIST[G] | **Responsibility**: a sequence of items of type G | **Alternative**: see ARRAY[G] |
|  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | LIST[G] | **Responsibility**: a sequence of items of type G | **Alternative**: see ARRAY[G] |
|  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | LIST[G] | **Responsibility**: a sequence of items of type G | **Alternative**: see ARRAY[G] |
|  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | LIST[G] | **Responsibility**: a sequence of items of type G | **Alternative**: see ARRAY[G] |
|  |  |

# Expanded description of design decisions

*Only for the most important module in your design.*

*What alternative designs were considered and rejected based on the criteria of reliability, simplicity, and maintainability?* The design is maintainable if it exhibits conceptual integrity that defines the key abstractions so that designers and programmers can reason about the system you describe and predict its behaviour. Software developers reading your SDD should be able to grasp your design without having to read thousands of lines of code. This will make you system extendible and re-usable.

# Significant Contracts (Correctness)

(only for the module with the most significant contracts)

.

# Summary of Testing Procedures

|  |  |  |
| --- | --- | --- |
| **Test file** | **Description** | **Passed** |
| *at001.txt* |  | ✓ |
| *at002.txt* |  |  |
| *at003.txt* |  |  |
| *at004.txt* |  |  |
| *at005.txt* |  |  |
| *at006.txt* |  |  |
| *at007.txt* |  |  |
| *at008.txt* |  |  |
| *at009.txt* |  |  |
| *at010.txt* |  |  |
| *at011.txt* |  |  |
| *at012.txt* |  |  |
| *at013.txt* |  |  |
| *at014.txt* |  |  |
| *at015.txt* |  |  |
| *at016.txt* |  |  |
| *at017.txt* |  |  |
| *at018.txt* |  |  |
| *at019.txt* |  |  |
| *at020.txt* |  |  |

# Appendix (Contract view of all classes)

(Only classes that you created; do not include user input command classes, only model classes)