**PRELIMINARY PROJECT**

**REPORT**

**CHINEBEN NELSON OGBUIGWE |**

**2057119 |**

**PROJECT: PLANETARY ODYSSEY |**

**Description of Classes and Inheritance Relationships:**

**Spaceship Class Hierarchy:**

* **Spaceship (Base Class):** Represents the basic attributes and behaviors of a spaceship.
  + **CargoSpaceShip (Derived Class):** Inherits from Spaceship, offers extended cargo capacity.
  + A screen shot of a computer program

    Description automatically generated*Justification:* The CargoSpaceShip class inherits from Spaceship to utilize its common functionalities while extending it with specific features related to cargo capacity. This inheritance hierarchy ensures reusability and maintains a clear relationship between different spaceship types.

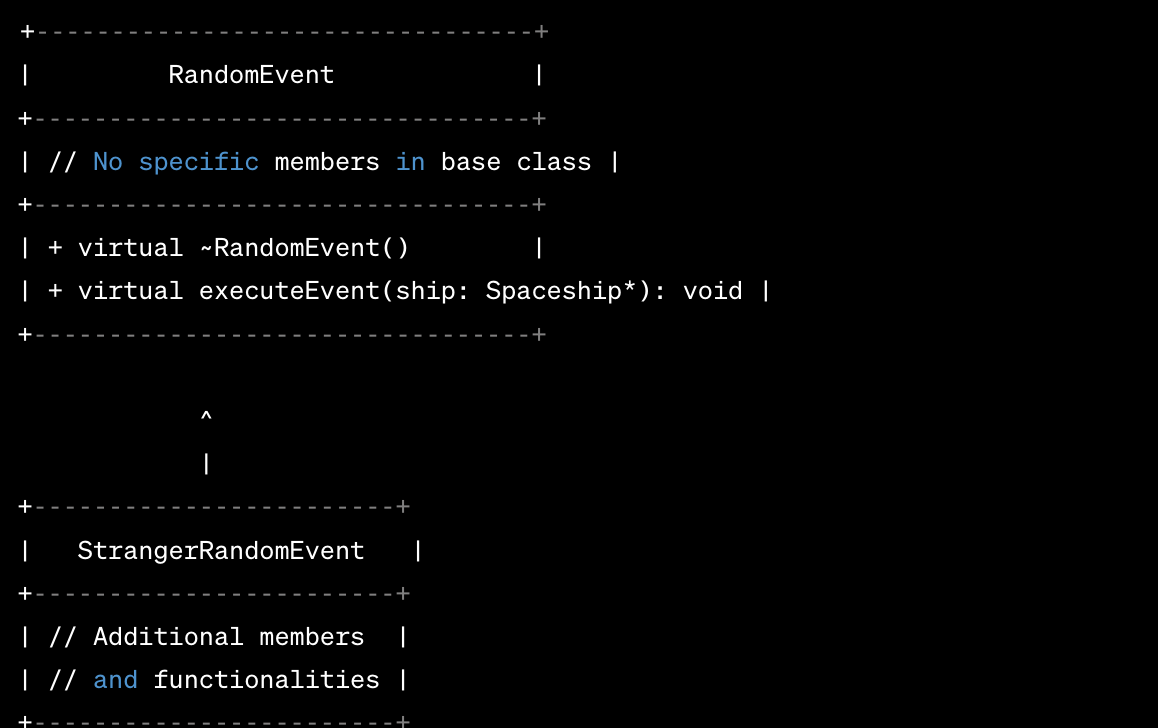
**Planet Class:**

* **Planet:** Represents a celestial body with specific characteristics.
  + *Explanation:* The Planet class stands independently without any inheritance, as each planet has unique properties and behaviors. This design choice allows for flexible implementation and management of diverse planets without creating unnecessary class hierarchies.

A screen shot of a computer program

Description automatically generated

**RandomEvent Class Hierarchy:**



* **RandomEvent (Base Class):** Represents a generic random event during planet exploration.
  + **StrangerRandomEvent (Derived Class):** Inherits from RandomEvent, simulates an encounter with a stranger.
  + *Rationale:* The StrangerRandomEvent class is derived from RandomEvent to extend the generic event behavior and introduce specific outcomes related to encountering a stranger. This design maintains a hierarchical structure for different random events while ensuring cohesive event handling.

**Relationship Among the Classes:**

* The Spaceship hierarchy (Spaceship and CargoSpaceShip) exhibits an 'is-a' relationship, where a CargoSpaceShip 'is-a' type of Spaceship, inheriting its properties and functionalities.
* The RandomEvent hierarchy (RandomEvent and StrangerRandomEvent) follows a similar 'is-a' relationship, with StrangerRandomEvent being a specialized type of RandomEvent, inheriting its base behaviors.

**Use Cases for the Program:**

**Menu Options and Their Use Cases:**

1. **Explore Planet:** Allows the player to select a planet for exploration.
   * Use Case:
     + Display available planets.
     + Player selects a planet.
     + Initiate exploration on the chosen planet.
2. **Save Game:** Enables the player to save the current game state.
   * Use Case:
     + Trigger saving of spaceship, planets, and game progress to a file.
3. **Exit Game:** Allows the player to exit the game.
   * Use Case:
     + Terminate the game loop and exit gracefully.

**Pseudocode for Critical Functionality:**

*Explore Planet Functionality Pseudocode:*

function explorePlanet(selectedPlanet):

if spaceship has enough fuel and crew:

deduct fuel and crew from the spaceship

initiate exploration on selectedPlanet

handle random events and interactions on the planet

else:

display insufficient resources message

*Main Game Loop Pseudocode:*

function mainGameLoop():

while gameRunning:

displayMainMenuOptions()

playerChoice = getUserInput()

switch playerChoice:

case 1:

explorePlanet(selectedPlanet)

case 2:

saveGame()

case 3:

exitGame()

default:

displayInvalidChoiceMessage()