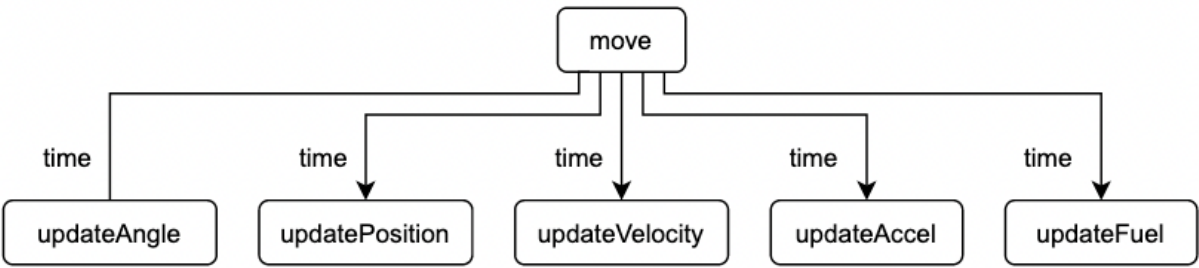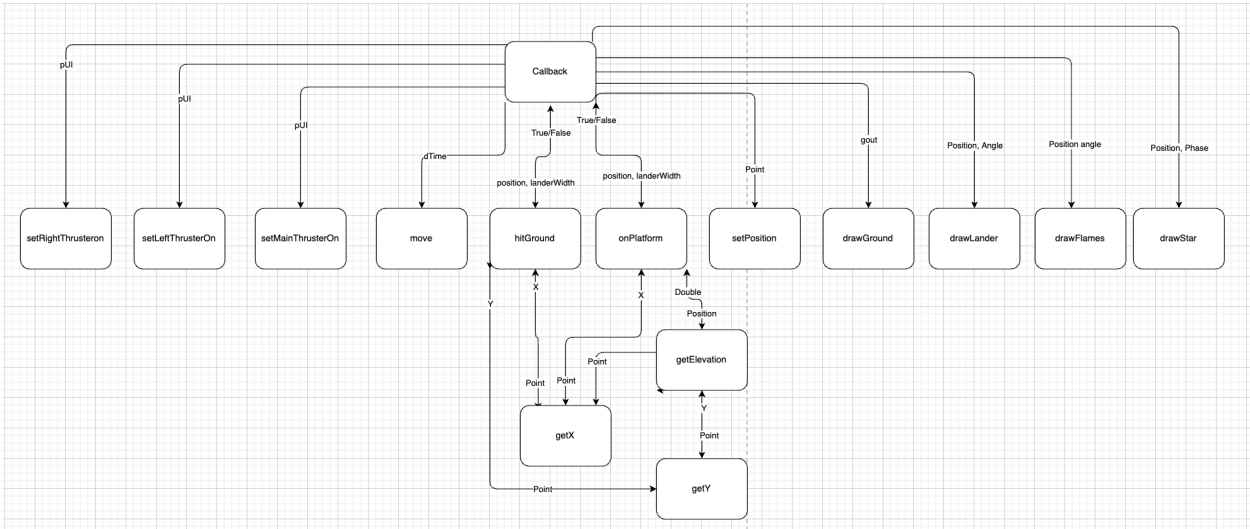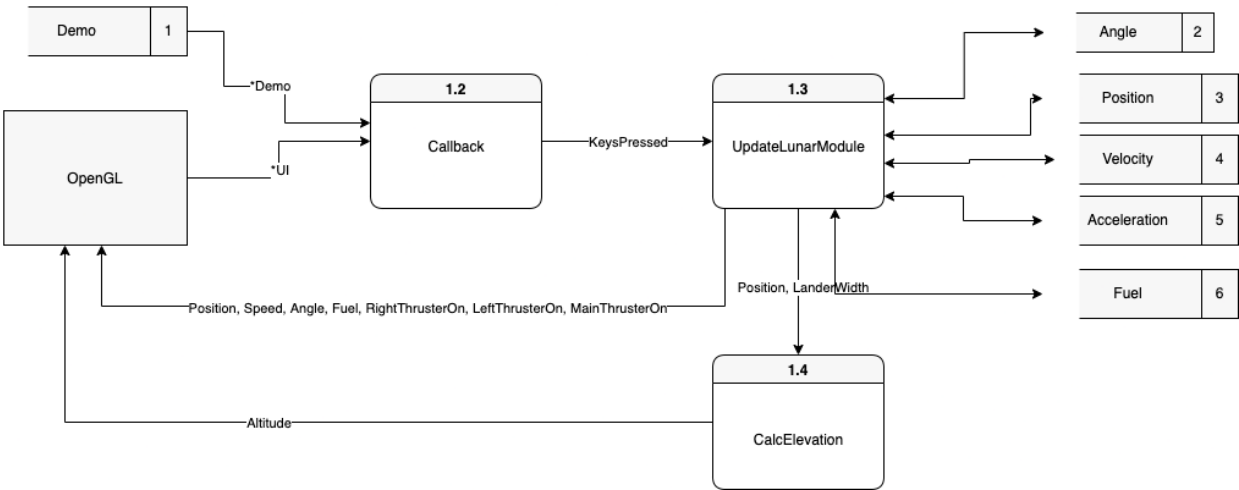# Structure Chart

Data Flow Diagram:

Class Diagrams:

Star

-minHeight : INT
-maxHeight: INT
-position: Point
-phase: unsigned char

-Star()
-Twinkle()
+drawStar()

Ground

-ground : Real [ ]
-iLZ : Integer
-width: Real
-height: Real

+Ground(width: Real, height: Real)
+reset()
+draw()
+getElevation(): double
+hitGround(): bool
+onPlatform(): bool

Demo

- lander : Lunar Lander
- angle: Real
+ ptStar: Star
+ phase: char
+ width : Real
+ height: Real
+ ground: Ground

+Demo(width: Real, height: Height)

LunarLander

- isMainThrustOn : boolean
- isLeftThrustOn : boolean
- isRightThrustOn : boolean
- WEIGHT : double
- GRAV_ACCEL : double
- MAIN_THRUST : double
- ROTATION_SPEED : double
- MAIN_FUEL_PER_SECOND : double
- AUX_FUEL_PER_SECOND : double
- WIDTH : int
- MAX_LANDING_SPEED : double
- MAX_LANDING_ANGLE : int
- MAX_POSITION_LIMITS : Point
- position : Point
- velocity : Point
- angle : double
- Fuel : double
- acceleration : Point

+ LunarLander (position: Point, velocity: Point = Point(0,0), angle: double=0, $\begin{matrix} fuel:double \\ = 1500 \end{matrix}$)
+ doesLand () : boolean
+ setMainThrusterOn (isOn : boolean)
+ setLeftThrusterOn (isOn : boolean)
+ setRightThrusterOn (isOn : boolean)
+ Move (time : double)
+ getMainThrusterOn () : boolean
+ getLeftThrusterOn () : boolean
+ getRightThrusterOn () : boolean
+ getSpeed () : double
+ getPosition () : double
+ getFuel () : double
+ getAngle () : double
+ getWidth() : int

- updateAcceleration ()
- updateVelocity (time : double)
- updatePosition (time : double)
- updateAngle (time : double)
- updateFuel (time : double)

Pseudocode:

Lunar Lander :: updatePosition()

```
def updatePosition (double time):
    # Apply inertia from velocity.
    double ChangeX = this.velocity.getX()
    double ChangeY = this.velocity.getY()
    # Apply motion from acceleration.
    ChangeX = changeX + this.acceleration.getX() * time * time / 2.0
    ChangeY = change Y + this.acceleration.getY() * time * time / 2.0
    # Add to position.
    this.position.addX ( changeX)
    this.position.addY ( changeY)
    # Validate position.
    if this.position.getX() < 0:
        this.position.setX (0)
    elif this.position.getX() > this.MAX_POSITION_LIMITS.getX():
        this.position.setX( this.MAX_POSITION_LIMITS.getX())
    if this.position.getY() < 0:
        this.position.setY(0)
    elif this.position.getY() > this.MAX_POSITION_LIMITS.getY():
        this.position.setY( this.MAX_POSITION_LIMITS.getY())
```

Lunar Lander :: updateFuel()

```
def updateFuel (double time):
    double fuelChange = 0.0
    if this.isMainThrustOn:
        fuelChange = fuelChange - this.MAIN_FUEL_PER_SECOND * time
    if this.isRightThrustOn:
        fuelChange = fuelChange - this.AUX_FUEL_PER_SECOND * time
    if this.isLeftThrustOn:
        fuelChange = fuelChange - this.AUX_FUEL_PER_SECOND * time
    this.fuel = this.fuel + fuelChange
    if this.fuel <= 0:
        this.fuel = 0
        this.isMainThrustOn = False
        this.isLeftThrustOn = False
        this.isRightThrustOn = False
```

Test Cases:

Name	Pre-condition	Input	Output	Reasoning
No Inputs	Lander has some horizontal and vertical velocity and is above the ground.	NONE	Lander maintains horizontal velocity and begins falling to the ground. No thruster flame is shown.	When there is no input, there is no thrust, and the only force acting on the lander is gravity.
Main Thruster Up	Lander has some fuel, angle is zero.	Down Arrow	Lander gains altitude and main flames are drawn	Make sure the main thruster works
Right Thruster	Lander has some fuel	Right Arrow	Lander angle moves counterclockwise, lander's fuel decreases, and right flames are drawn	If the right arrow is pressed, then the lander changes angle.
Left Thruster	Lander has some fuel	Left Arrow	Lander angle moves clockwise, lander's fuel decreases, and left flames are drawn	If the left arrow is pressed, then the lander rotates clockwise.
Left Right Thrusters	Lander has fuel	Left & Right Arrow	Lander Fuel goes down, left and right flames are drawn, lander angle stays the same	Having both left and right thrusters on should cancel out any angle movement.
Left Right and Main Thrusters	Lander has fuel, lander angle is zero	Left, Right, & Down arrow	Lander fuel goes down; left, right, and main flames are drawn, lander accelerates upward and does not rotate.	Having all thrusters on maintains angle, but does cause acceleration in the direction the lander is facing.
1st Quadrant Angle	Lander is at an angle between 0 and 90 degrees	Down Arrow	Lander accelerates up and to the right.	If the lander is at an angle and the main thruster is engaged, the lander should accelerate in that direction

4th Quadrant Angle	Lander angle is between 270 and 360 degrees.	Down Arrow	Lander accelerates up and to the left	If the lander is at an angle and the main thruster is engaged, the lander should accelerate in that direction.
2nd Quadrant Angle	Lander is at an angle between 90 and 180 degrees	Down Arrow	Lander accelerates down and to the right	If the lander is pointed down and the main thruster is engaged, the lander should accelerate in that direction.
3rd Quadrant Angle	Lander angle is between 180 and 270 degrees.	Down Arrow	Lander accelerates down and to the left.	If the lander is at an angle and the main thruster is engaged, the lander should accelerate in that direction.
4th Quadrant Negative Angle	Lander angle is between 0 and –90 degrees.	Down Arrow	Lander accelerates up and to the left.	The lander angle should handle negatives.
Left Edge Collision	Lander is near the left edge of the screen with a negative horizontal velocity.	None	Lander moves to the left edge of the screen, and then stops moving horizontally.	The lander is not allowed to go off the screen.
Right edge collision	Lander is near the right edge of the screen with a positive horizontal velocity	None	Lander moves to the right edge of the screen and then stops moving horizontally	The lander is not allowed to go off screen
Top Edge Collision	Lander is near the top edge of the screen with a positive vertical velocity	None	Lander moves to the top edge of the screen, then stops moving vertically until its vertical velocity decays due to gravity.	The lander is not allowed to go off the screen.

Ground Collision: Right base of lander	Lander is approaching ground where ground is at a positive slope. Lander has a negative vertical velocity.	None	Right base touches the ground first and lander crashes	The lander cannot land at a slope
Ground Collision: Left Base of Lander	Lander is approaching ground where ground is at a negative slope. Lander has a negative vertical velocity.	None	Left base of the lander touches the ground first, and the lander crashes.	The lander cannot land at a slope.
Landing Zone Collision: Too fast	Lander is approaching landing zone with too much speed	None	Lander crashes when it collides with the landing zone	The lander will crash if it picks up too much speed
Landing Zone Collision: Wrong Angle	Lander is approaching landing zone in excess of the max angle.	None	Lander crashes when it collides with the landing zone.	The lander must be mostly upright to land.
Landing Zone Collision: Safe Landing	Lander is approaching landing zone with zero speed and zero angle	None	Lander lands safely and the simulation ends.	The lander *can* land, it just needs the right situation.
Landing Zone: Partial Landing	Lander is approaching landing zone where half the lander will land on the landing zone and half will land on the ground	None	Lander crashes when it collides with the ground and landing zone	The lander cannot partially land on the landing zone
Lander Runs out of Fuel	All thrusters are on, and the lander has little fuel.	Down, Left, and Right Arrows	Lander runs out of fuel, and all thrusters shut off. The only acceleration comes from gravity, and there is no change in angle.	If the user runs out of fuel while engaged in a maneuver, they can't maneuver any more.
Lander Does not have fuel	Lander has no fuel and no thrusters are on	Any arrow key	Thrusters will not turn on and lander will fall towards the ground	If there is no fuel the lander cannot turn on the thrusters