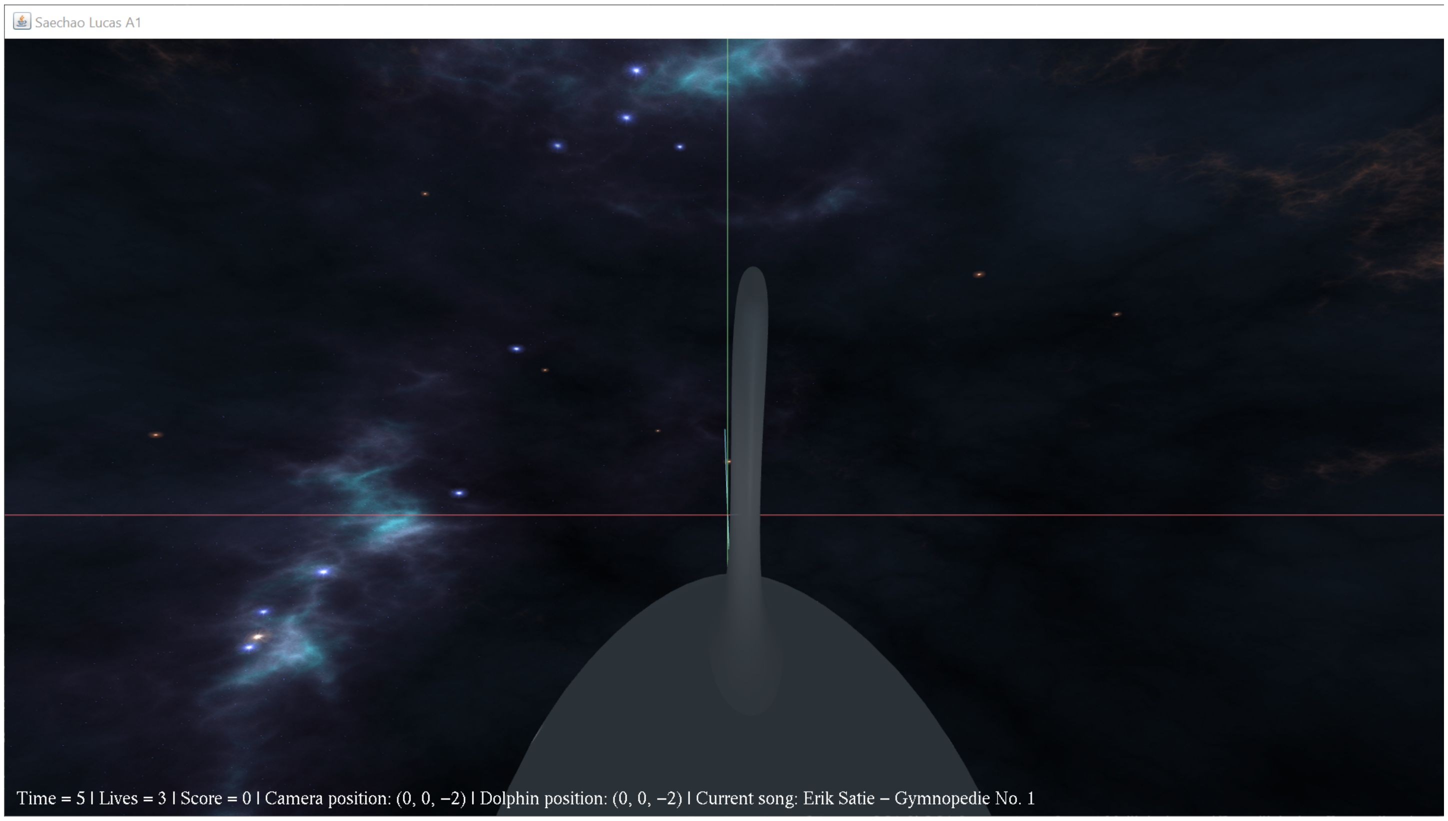
Planet Chaser

Lucas Saechao

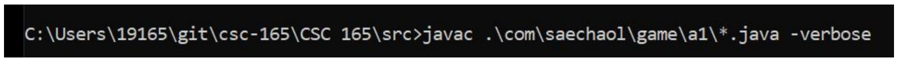


```
Time = 5 | Lives = 3 | Score = 0 | Camera position: (0, 0, −2) | Dolphin position: (0, 0, −2) | Current song: Erik Satie – Gymnopedie No. 1
```

Installation

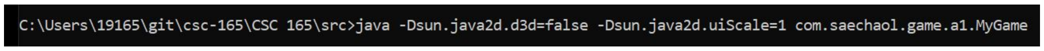
To install this game, run the following commands:

compile : `javac .\com\saechaol\game\a1\*.java -verbose`



```
C:\Users\19165\git\csc-165\CSC 165\src>javac .\com\saechaol\game\a1\*.java -verbose
```

run : `java -Dsun.java2d.d3d=false -Dsun.java2d.uiScale=1 com.saechaol.game.a1.MyGame`



```
C:\Users\19165\git\csc-165\CSC 165\src>java -Dsun.java2d.d3d=false -Dsun.java2d.uiScale=1 com.saechaol.game.a1.MyGame
```

How to Play

The goal of the game is to chase down planets and avoid being hit by their orbiting moons. You can only score when on foot. You can only survive 3 hits, but you can earn more lives by every 10th point collected. When you are hit:

- If you are dismounted: you will return to the dolphin.
- If you are on the dolphin: you will return to the origin.

Controls (Keyboard / Xbox One Gamepad)

1P Camera Controls

WASD / Left Stick — Move dolphin/camera
Arrow Keys / Right Stick — Rotate dolphin/camera
QE / Left & Right bumpers — Roll camera left/right
V / Y button — Invert yaw controls
LShift / C / LS & RS click — ascend / descend

Space / A button — mount/dismount dolphin
Tab / Menu (start) — Switch to 3P camera
P / X button — play next song
ESC / View (select) — quit game

3P Camera Controls

Arrow Keys / Right stick — orbit/elevate camera
RF / Left & Right triggers — zoom in/out
Space / A button — dismount dolphin and switch to 1P camera
Tab / View (select) — switch back to 1P camera

Extra Game Activity

In this game, you are given lives. When you collide with a moon, you take damage. When your health reaches zero, the game will automatically close and the game will be over. When the player scores by colliding with a planet, on foot, that planet is deleted and a new planet is generated somewhere in the map.

Manual Object

The manual object I have made was a cube. While the extra sides compared to a pyramid are minor, it was still required the same work to implement, and I more or less have a much stronger understanding and appreciation of manually modeling a 3D object.

Nonworking Requirements

None. All of the minimum specifications have been achieved.

Extra Stuff

- All of the camera rotations in first person mode are implemented with Quaternions, which I now understand in greater depth.
- I have implemented an audio manager that plays music or sound effects while the user flies through the space.
- A skybox is implemented.
- OrbitController does not provide a way to change the axis of a planet's orbit, but I had managed to achieve so anyway.

Assets

MODELS/MATERIALS

- star.obj — I made this in *Maya3D*
- star.mtl — Also made in *Maya3D*

TEXTURES

- Skybox — these were generated by an open source tool called *spacescape*, and processed with *pngquant* to reduce file overhead and rasterized into JPG with *Adobe Photoshop*
    - spaceSkyboxFront.jpg
    - spaceSkyboxBack.jpg
    - spaceSkyboxLeft.jpg
    - spaceSkyboxRight.jpg
    - spaceSkyboxTop.jpg
    - spaceSkyboxBottom.jpg

SOUND

- sounds/sfx — recorded by myself using digital instrumentation in *Logic Pro X*
    - destroyed.wav
    - lifeup.wav
    - score.wav
- sounds/music :
    - Erik Satie — the gymnopedies are public domain
        - gymnopedie_one.wav
        - gymnopedie_two.wav
        - gymnopedie_three.wav