1. **­Eduardo Guzman, CSC-165 Section­ 2­­, A2 – Dolphin Adventure 2**
2. *Diagram

   Description automatically generated*
3. **How to play:**
   1. The game consists of one main character/avatar: the dolphin. You must move the dolphin throughout the Gameworld and collect prizes to win. To “win”, the player must collect a certain number of prizes (can be set in the code and will also be described in the “Scoring” section). The prizes are in the shape of spheres, so we can refer to them as prize orbs. To collect a prize orb, the avatar must collide with the prize orb.
   2. There is an additional restriction: the game avatar has a food level and to collect prizes, the food level must be above a certain threshold (also explained in “Scoring”). Since a player can become hungry, there are also Food Stations scattered throughout the world that look like drawers and act as a food source for your avatar. To collect food, the avatar makes contact with the station. The food is not consumed right away; it is stored in a food storage buffer and the player must press a specific key in order to eat food.
   3. The Torus shape will represent food and the dolphin will have a child game object that rotates around it when the food storage buffer contains food. If the buffer is empty, the food torus is fixed next to the dolphin.
   4. A ground plane is now implemented a the dolphin can only move above the plane.

**Scoring:**

* 1. There are two scores to keep track of:
     1. **Prizes:** You will collect prize orbs and when you collect *30*, you win.
        1. The winning prize number can be changed in the code to enhance/lessen the difficulty.
     2. **Food Level:** You must try to stay above a food level of *50* to collect prizes and win. The food level starts at *200*.

1. **Inputs for Moving the Dolphin:**
   1. **“A”, “Left Joystick”**: The “A” key on a keyboard, and right motion with the “Left Joystick**”** on a gamepad,simulates a left turn for the dolphin.
   2. **“D”, “Left Joystick”:** The “D” key on a keyboard, and right motion with the “Left Joystick**”** on a gamepad, simulates a right turn for the dolphin.
   3. **“W”, “Left Joystick”:** The “W” key on a keyboard, and upwards motion with the “Left Joystick**”** on a gamepad, simulates a forward movement for the dolphin.
   4. **“S”, “Left Joystick”:** The “S” key on a keyboard, and downwards motion with the “Left Joystick**”** on a gamepad, simulate a backwards movement for the dolphin.
   5. **“K”, “RB”:** The “K” key on a keyboard and “RB” button on a gamepad simulates an upwards pitch for the dolphin.
   6. **“J”, “LB”:** The “J” arrow key on a keyboard and “LB” button on a gamepad simulates a downwards pitch for the dolphin.
   7. **“E”, “A”:** The “E” key on a keyboard and the “A” button on a gamepad acts as the eating trigger for the dolphin.
   8. **“1”**, **“Y”:** The “1” key on a keyboard and “Y” button on the gamepad toggles the dolphin’s view as a wireframe.
2. **Inputs for Moving the Orbit Controller:**
   1. **“Left arrow”, “Right Joystick”**: The “Left arrow” key, and left motion with the “Right Joystick**”** on a gamepad,simulates an Azimuth left orbit for the 3D Orbit Camera.
   2. **“Right arrow”, “Right Joystick”:** The “Right arrow” key on a keyboard, and right motion with the “Right Joystick**”** on a gamepad, simulates an Azimuth right orbit for the 3D Orbit Camera.
   3. **“Up Arrow”, “Right Joystick”:** The “Up arrow” key on a keyboard, and upwards motion with the “Right Joystick**”** on a gamepad, simulates an orbit elevation for the 3D Orbit Camera.
   4. **“Down arrow”, “Right Joystick”:** The “Down arrow” key on a keyboard, and downwards motion with the “Right Joystick**”** on a gamepad, simulate a negative orbit elevation for the 3D Orbit Camera.
   5. **“I”, “Right Joystick”:** The “I” key on a keyboard and a “Right Joystick” button click on a gamepad simulates zooming in with the 3D Orbit Camera.
   6. **“O”, “Left Joystick”:** The “O” arrow key on a keyboard and s “Left Joystick” button click on a gamepad simulates zooming out with the 3D Orbit Camera.
3. **Inputs for Overhead Camera:**
   1. **“NUMPAD8”**, **“D-PAD UP”:** The “NUMPAD8” key on a keyboard and “D-PAD UP” button on the gamepad simulates panning the overhead camera up.
   2. **“NUMPAD6”**, **“D-PAD RIGHT”:** The “NUMPAD6” key on a keyboard and “D-PAD RIGHT” button on the gamepad simulates panning the overhead camera right.
   3. **“NUMPAD2”**, **“D-PAD DOWN”:** The “NUMPAD2” key on a keyboard and “D-PAD DOWN” button on the gamepad simulates panning the overhead camera down.
   4. **“NUMPAD4”**, **“D-PAD LEFT”:** The “NUMPAD4” key on a keyboard and “D-PAD LEFT” button on the gamepad simulates panning the overhead camera left.
   5. **“NUMPAD9”**, **“RT”:** The “NUMPAD9” key on a keyboard and “RT” button on the gamepad simulates zooming in with the overhead camera.
   6. **“NUMPAD7”**, **“LT”:** The “NUMPAD7” key on a keyboard and “LT” button on the gamepad simulates zooming out with the overhead camera.
4. **Additional Inputs:**
   1. **“Esc”:** the “Esc” key on a keyboard will terminate the game.
   2. **“F”,** “**BACK”:** The “F” key on a keyboard and “BACK” button on the gamepad will toggle the screen from full screen to windowed.
   3. **“Space”, “START”:** The “Space” key on a keyboard and “START” button on the gamepad will pause/un-pause the game.
   4. **“T”**, **“X”:** The “T” key on a keyboard and “X” button on the gamepad toggles the XYZ axis lines instantiated in the world.
5. **Description of Node Controllers**
   1. **tage/nodeControllers/BounceController.java**
      1. This node controller will make food stations bounce when the dolphin first collides with them. The idea behind this bounce is that it simulated the food station depositing food to the players food storage.
6. **Scenegraph Relationships**
   1. The game now includes a food torus, which is a child of the Dolphin. This food torus is the indicator that tells the player the state of the food storage buffer.
      1. If the food buffer is empty, the torus is in a fixed position near the dolphin.
      2. If the food buffer is not empty, the torus will rotate around the dolphin.
   2. This behavior will continue throughout the game.
7. **Changes made to TAGE Engine**
   1. **Camera.java *(unchanged from a1)***
      1. Added a global yaw function to turn the camera around the World Y Axis
      2. Added a local yaw function to turn the camera around the its local Y Axis
      3. Added a local pitch function to simulate the player looking up and down
   2. **GameObject.java *(unchanged from a1)***
      1. Added a global yaw function to give objects turning capabilities
      2. Added a local pitch function to give objects upward/downward movement
   3. **HUDmanager.java *(unchanged from a1)***
      1. Created functionality for a third and fourth HUD display
8. **I was able to meet all the requirements for the assignment.**
9. **Addition Game Features not required:**
10. **Assets Used: *(this list is unchanged from a1)***
    1. **assets/textures/Dolphin\_HighPolyUV.png** 
       1. Already existed in the TAGE engine
    2. **assets/textures/Floral\_Sheet.png**
       1. Image was captured by me on my mobile phone (jpg converted to a PNG)
    3. **assets/textures/Wood\_Desk.png**
       1. Image was captured by me on my mobile phone (JPG converted to a PNG)
    4. **assets/textures/Drawer\_Door.jpg**
       1. Image was captured by me on my mobile phone
    5. ***All other assets that TAGE instantiates itself***