Declare all player/ai variables, classes and types, setscreen modes, game settings, functions and procedures.

Prepare all variables and run game without players as a title page background

True

True

False

if getKey = (KEY\_ESC)

True

False

Elsif decision = Highs core

If decision is Play

Start

True

False

If play again

Receive player’s choice of action -> play again or quit

Display winner and score

Find Winner (scorelist): string

True

False

End

Receive player’s choice of action -> play, highscore, or quit

Display top ten scores

False

Clear Screen

Clear

Drawscreen

Is the time up?

The Fish Game

The purpose of the fish game is to eat as many fish as possible to achieve the highest score. It can be played with 1, 2 or 3 players with WASD, arrow keys and the mouse as controllers. In the game, players play as fish that are able to eat anything smaller than it to gain points while avoiding bigger fish which can eat the player. After eating a certain amount of other fish, the player grows bigger and is able to eat bigger fish that grant more points. To ensure that a player is never invincible, there are also sharks that quickly swim by and eat anything in its way. Upon death, players lose a fraction of their score, risk shrinking to a smaller size (depending on the score they have after death) and are respawned after about 4 seconds. After a designated amount of time, the game ends and the player with the highest score is declared the winner.

The game works off of a series of functions and procedures which are called in a main procedure called Drawscreen located in a loop that runs the actual game. Within the Drawscreen procedure, the program uses functions for collision detection and procedures for everything else (player movement, fish spawning .etc) to obtain results for a visual image of the game at the current frame. Drawscreen has 3 major components to keep track of; players, AiFish, and collision detection. The player movement procedure works by changing the x and y coordinates of the player’s fish by receiving the input from player through the predefined controls that were assigned to the player fish (WASD, arrow key mouse). After receiving input, a new fish is drawn accordingly. The AiFish procedure works by using a for loop that keeps tracks of the Ai fish with each repetition of the for loop managing a separate fish. When a fish is nonexistent due to death or swimming off the screen, the procedure generates a random number to create a spawn point for the fish just outside of the screen. If the fish is existent, then the procedure moves it around until it is nonexistent (from being eaten/dead or swimming off the screen). The Collision detection function is used to check for any collisions for the object assigned in its parameters and returns a Boolean as a result. This is done by creating a collision radius for each Ai/player and checking if anything else on the map is within that radius. In general any Ai or player that comes in contact with the mouth of a bigger Ai is eaten and marked as dead or nonexistent by the collision function with sharks being the only exception to this rule since it will never be eaten. Although this is what the program is currently capable of, later on, complicated movement (Roaming) and tracking down players in a limited reaction range (Hunting) will be available for AiFish to increase difficulty and simulate a more realistic aqueous ecosystem.