Project 9: Ice Fishing Game

Submitted by:

- Tomer Shulman - Tomer.shulman@post.runi.ac.il
- Yarin Kovalski - Yarin.kovalski@post.runi.ac.il

1. App Concept

Our application is a remake of the nostalgic "Ice Fishing" mini-game from Club Penguin. The player controls a penguin fisherman who must lower a hook to catch swimming fish for points while avoiding dangerous jellyfish that reduce the player's lives.

2. Motivation

We chose to create this particular app because it combines nostalgic value with interesting technical challenges. Implementing a real-time game with multiple moving objects (fish, jellyfish, hook) and collision detection allowed us to fully explore the capabilities and limitations of the Jack language and the Hack platform.

3. System Architecture

The system is designed using an Object-Oriented approach, where the Game class acts as the central controller managing independent entities.

- **Main.jack**: The entry point of the program. It initializes the Game instance, runs the main loop, and handles the high-level logic for restarting the game upon completion.
- **Game.jack**: The core engine of the application. It manages the game loop, handles user input (keyboard), spawns and manages dynamic objects (using Arrays for fish and jellyfish), detects collisions between the hook and other entities, and updates the game state (score, lives, game over).
- **Hook.jack**: Represents the fishing line and hook. It handles the vertical movement logic (up/down), ensures the hook stays within bounds, and manages the drawing and erasing of the line on the screen.
- **Fish.jack**: Represents a single fish entity. It contains logic for horizontal movement, state management (swimming vs. caught), and specific bitmap drawing methods. It also handles the "snap-to-hook" mechanic when caught.
- **Jellyfish.jack**: Represents an obstacle entity. Similar to the Fish class but with different graphics and movement patterns. It interacts with the Game class to trigger a "life lost" event upon collision.
- **Penguin.jack**: Handles the static graphical representation of the player character using complex bitmap drawing commands (Memory.poke) to render the penguin on the screen.
- **Ice.jack**: Responsible for drawing the static background elements, specifically the ice surface and the fishing hole, setting the visual stage for the game.

4. Video Demo

Youtube Link:
https://www.youtube.com/watch?v=ClRgJG9WAKE