Project Overview

1.What is the project about?

This project recreates the classic mobile game, released in 2013 by Dong Nguyen, known as Flappy bird in Java using AWT, Swing, Arraylist and Util packages.

2.What were the assets used for this project?

The project includes handmade assets of all the environmental features of the original game made using the gnu image manipulation program(GIMP). Including but not limited to the bird, the pipes and even the city landscape background.

3.What is the goal of this project?

The goal of this project was to learn how to effectively use AWT and Swing packages in java in order to build applications.

4.What did the contributors contribute to?

Heet Mewada(A029): Assets, GameLoop and Math & Logic of the game.

Yuvraj Chavan(A008): Code and Bug testing.

Riaan Chhadva(A009): Code and Bug testing.

5.What did you learn from doing this project?

How to effectively use AWT and Swing in java to build applications

How to use version control using github in order to easily collaborate on a project

How to make usable assets for a game

How to effectively use other packages to improve and make coding easier

How to effectively deliver an applications that your end users would enjoy

Code of the project

Flappybird.java

```
import java.awt.*;
import java.awt.event.*;
import java.util.ArrayList;
import java.util.Random;
import javax.swing.*;

public class FlappyBird extends JPanel implements ActionListener, KeyListener {
    int boardWidth = 360;
    int boardHeight = 640;

    //images
    Image backgroundImg;
    Image birdImg;
    Image topPipeImg;
    Image bottomPipeImg;

    //bird class
    int birdX = boardWidth/8;
    int birdY = boardWidth/2;
    int birdWidth = 34;
    int birdHeight = 24;

    class Bird {
        int x = birdX;
        int y = birdY;
        int width = birdWidth;
        int height = birdHeight;
        Image img;
        Bird(Image img) {
            this.img = img;
        }
    }

    //pipe class
    int pipeX = boardWidth;
    int pipeY = 0;
    int pipeWidth = 64;  //scaled by 1/6
    int pipeHeight = 512;

    class Pipe {
        int x = pipeX;
        int y = pipeY;
        int width = pipeWidth;
        int height = pipeHeight;
        Image img;
        boolean passed = false;

        Pipe(Image img) {
            this.img = img;
        }
    }

    //game logic
    Bird bird;
    int velocityX = -4; //move pipes to the left speed (simulates bird moving right)
    int velocityY = 0; //move bird up/down speed.
    int gravity = 1;

    ArrayList<Pipe> pipes;
    Random random = new Random();
```

```
Timer gameLoop;
Timer placePipeTimer;
boolean gameOver = false;
double score = 0;

FlappyBird() {
    setPreferredSize(new Dimension(boardWidth, boardHeight));
    // setBackground(Color.blue);
    setFocusable(true);
    addKeyListener(this);

    //load images
    backgroundImg = new ImageIcon(getClass().getResource("./flappybirdbg.png")).getImage();
    birdImg = new ImageIcon(getClass().getResource("./flappybird.png")).getImage();
    topPipeImg = new ImageIcon(getClass().getResource("./toppipe.png")).getImage();
    bottomPipeImg = new ImageIcon(getClass().getResource("./bottompipe.png")).getImage();

    //bird
    bird = new Bird(birdImg);
    pipes = new ArrayList<Pipe>();

    //place pipes timer
    placePipeTimer = new Timer(1500, new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
          // Code to be executed
          placePipes();
        }
    });
    placePipeTimer.start();
```

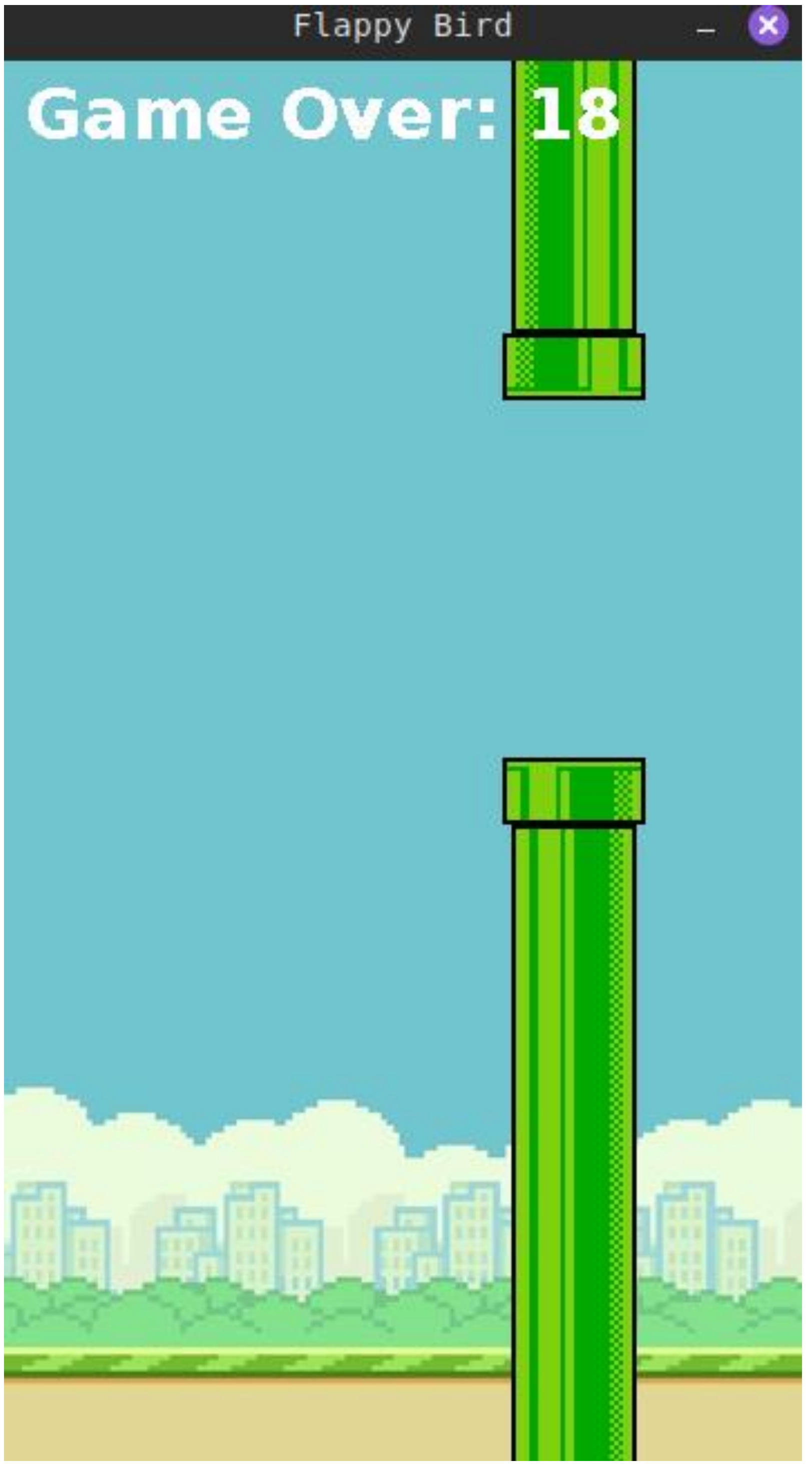
```
                    //game timer
                    gameLoop = new Timer(1000/70, this); //how long it takes to start timer,
milliseconds gone between frames
        gameLoop.start();
            }

    void placePipes() {
        //(0-1) * pipeHeight/2.
        // 0 -> -128 (pipeHeight/4)
        // 1 -> -128 - 256 (pipeHeight/4 - pipeHeight/2) = -3/4 pipeHeight
        int randomPipeY = (int) (pipeY - pipeHeight/4 - Math.random()*(pipeHeight/2));
        int openingSpace = boardHeight/4;

        Pipe topPipe = new Pipe(topPipeImg);
        topPipe.y = randomPipeY;
        pipes.add(topPipe);

        Pipe bottomPipe = new Pipe(bottomPipeImg);
        bottomPipe.y = topPipe.y  + pipeHeight + openingSpace;
        pipes.add(bottomPipe);
    }


    public void paintComponent(Graphics g) {
                    super.paintComponent(g);
                    draw(g);
            }

            public void draw(Graphics g) {
        //background
        g.drawImage(backgroundImg, 0, 0, this.boardWidth, this.boardHeight, null);
        //bird
        g.drawImage(birdImg, bird.x, bird.y, bird.width, bird.height, null);

        //pipes
        for (int i = 0; i < pipes.size(); i++) {
            Pipe pipe = pipes.get(i);
            g.drawImage(pipe.img, pipe.x, pipe.y, pipe.width, pipe.height, null);
        }

        //score
        g.setColor(Color.white);

        g.setFont(new Font("Times New Roman", Font.BOLD, 32));
        if (gameOver) {
            g.drawString("Game Over: " + String.valueOf((int) score), 10, 35);
        }
        else {
            g.drawString(String.valueOf((int) score), 10, 35);
        }

            }

    public void move() {
        //bird
        velocityY += gravity;
        bird.y += velocityY;
        bird.y = Math.max(bird.y, 0); //apply gravity to current bird.y, limit the bird.y to top of the
canvas

        //pipes
        for (int i = 0; i < pipes.size(); i++) {
            Pipe pipe = pipes.get(i);
        pipe.x += velocityX;

        if (!pipe.passed && bird.x > pipe.x + pipe.width) {
            score += 0.5; //0.5 because there are 2 pipes! so 0.5*2 = 1, 1 for each set of pipes
            pipe.passed = true;
        }

        if (collision(bird, pipe)) {
            gameOver = true;
        }
    }

    if (bird.y > boardHeight) {
        gameOver = true;
    }
}

boolean collision(Bird a, Pipe b) {
    return a.x < b.x + b.width &&   //a's top left corner doesn't reach b's top right corner
           a.x + a.width > b.x &&   //a's top right corner passes b's top left corner
           a.y < b.y + b.height &&  //a's top left corner doesn't reach b's bottom left corner
           a.y + a.height > b.y;    //a's bottom left corner passes b's top left corner
}

@Override
public void actionPerformed(ActionEvent e) { //called every x milliseconds by gameLoop timer
    move();
    repaint();
    if (gameOver) {
        placePipeTimer.stop();
        gameLoop.stop();
    }
    }

    @Override
    public void keyPressed(KeyEvent e) {
        if (e.getKeyCode() == KeyEvent.VK_SPACE) {
            // System.out.println("JUMP!");
            velocityY = -9;

            if (gameOver) {
                //restart game by resetting conditions
                bird.y = birdY;
                velocityY = 0;
                pipes.clear();
                gameOver = false;
                score = 0;
                gameLoop.start();
                placePipeTimer.start();
            }
        }
    }

    //needed for compilation process
    @Override
    public void keyTyped(KeyEvent e) {}

    @Override
    public void keyReleased(KeyEvent e) {}
}
```

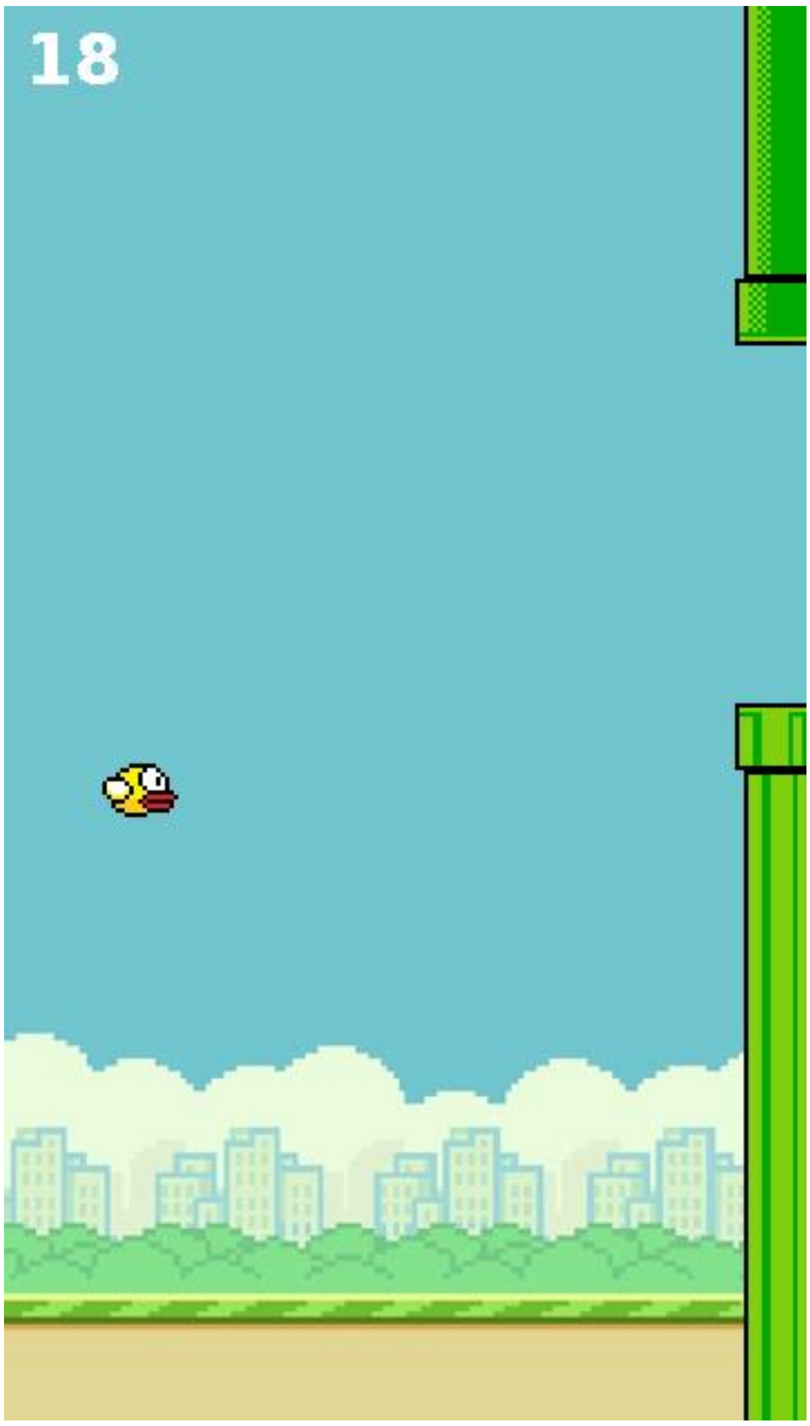
Code of the project

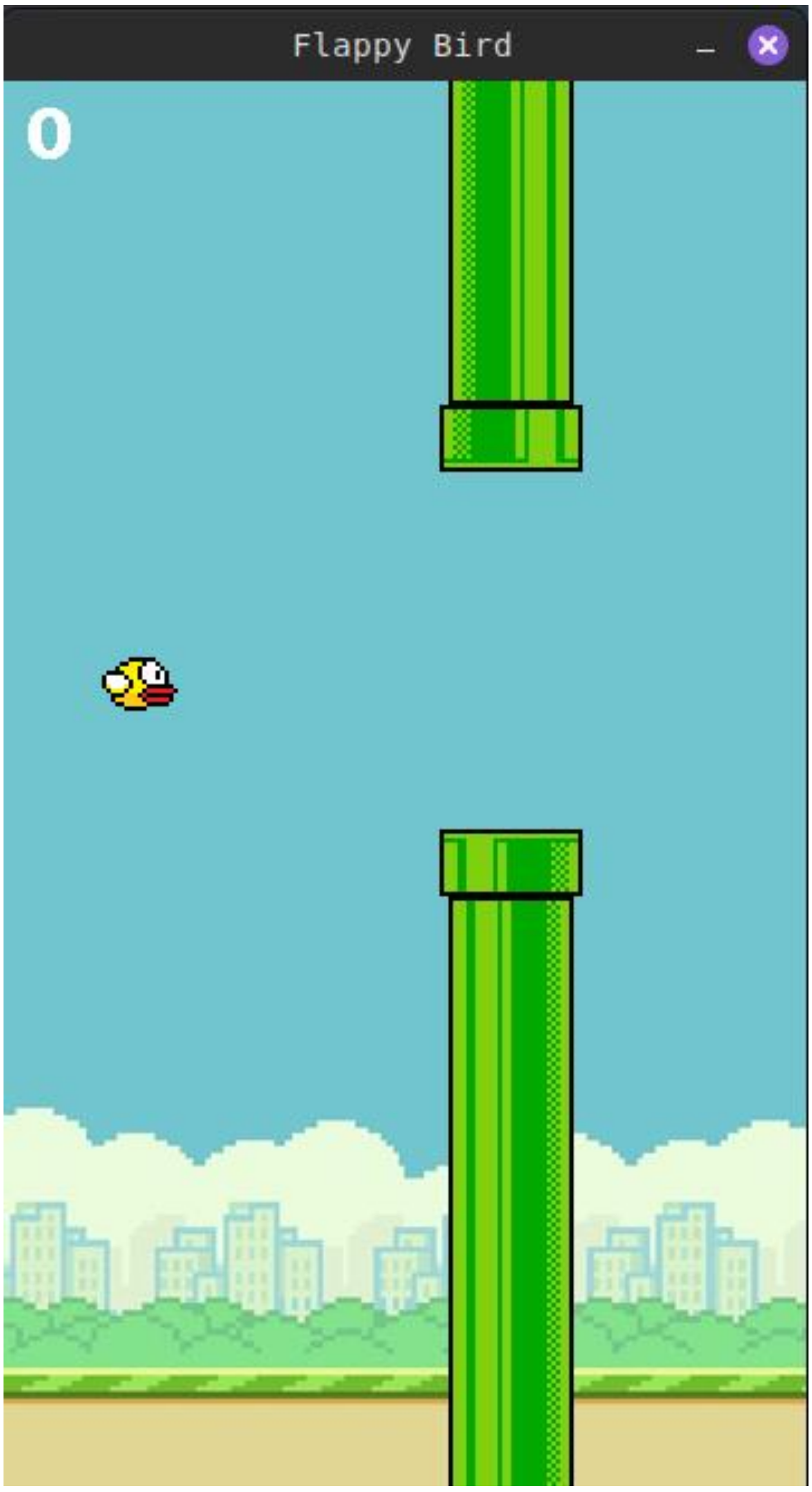
App.java

```
import javax.swing.*;


public class App {
    public static void main(String[] args) throws Exception {
        int boardWidth = 360;
        int boardHeight = 640;


        JFrame frame = new JFrame("Flappy Bird");
        // frame.setVisible(true);
                    frame.setSize(boardWidth, boardHeight);
        frame.setLocationRelativeTo(null);
        frame.setResizable(false);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);


        FlappyBird flappyBird = new FlappyBird();
        frame.add(flappyBird);
        frame.pack();
        flappyBird.requestFocus();
        frame.setVisible(true);
    }
}
```

Output of project




18


Flappy Bird
0