Project 1

<Mancala Game>

CIS-5/17/18
Name: Student
Date: 10/20/05

Introduction

Title: Mancala Game

Mancala is played on a board with two rows of six holes facing each other.
Each player owns a row of six holes, plus a Mancala to the right of the holes.
When the game begins, there are four eggs in each hole, and the two Mancalas are empty.
A player begins his clicking on one of their holes.
The eggs in that hole are picked up and distributed counter-clockwise around the board, placing one egg in each hole.
Eggs are distributed into the player's own Mancala, but the opponent's Mancala is skipped.
If the last egg distributed lands in the player's Mancala, the player gets another turn.
If the last egg lands on an empty hole owned by the player and the opponent's hole directly across the board contains at least one egg, the player takes all the eggs in that hole plus the last landed egg and puts them all in their Mancala.
Otherwise, the player's turn ends, and the opponent moves.

If all the holes of a player become empty, the player with the higher score calculated by eggs in their Mancala and eggs remaning in their holes wins the game, and the game ends.

Summary

Project size: 600+ lines
?? The number of variables: about 30
The number of method: 17

I tried to implement an object-oriented program, but there was no way to use inheritance.
Therefore, I just used the new concepts that we have learned, array, object-based and so on.
I programmed just a base game, and it still needs to be completed.
For example, one player game (with computer), sound effect, and database to display records.

It took around one week.
It was not so hard because of the former project experience.
However, I met many problems.
I referenced web pages and books and solved some of them.
I'm still not sure whether there is a better way, but it was a kind of excitement.

I used some new concepts we haven't learned.
Timer class, GridBagLayout manager, Icon, menu bar…

Description

The main point of this program is that how it works when buttons are pressed, and how Timer class functions to make slow progress.

FlowChart
Pseudo Code

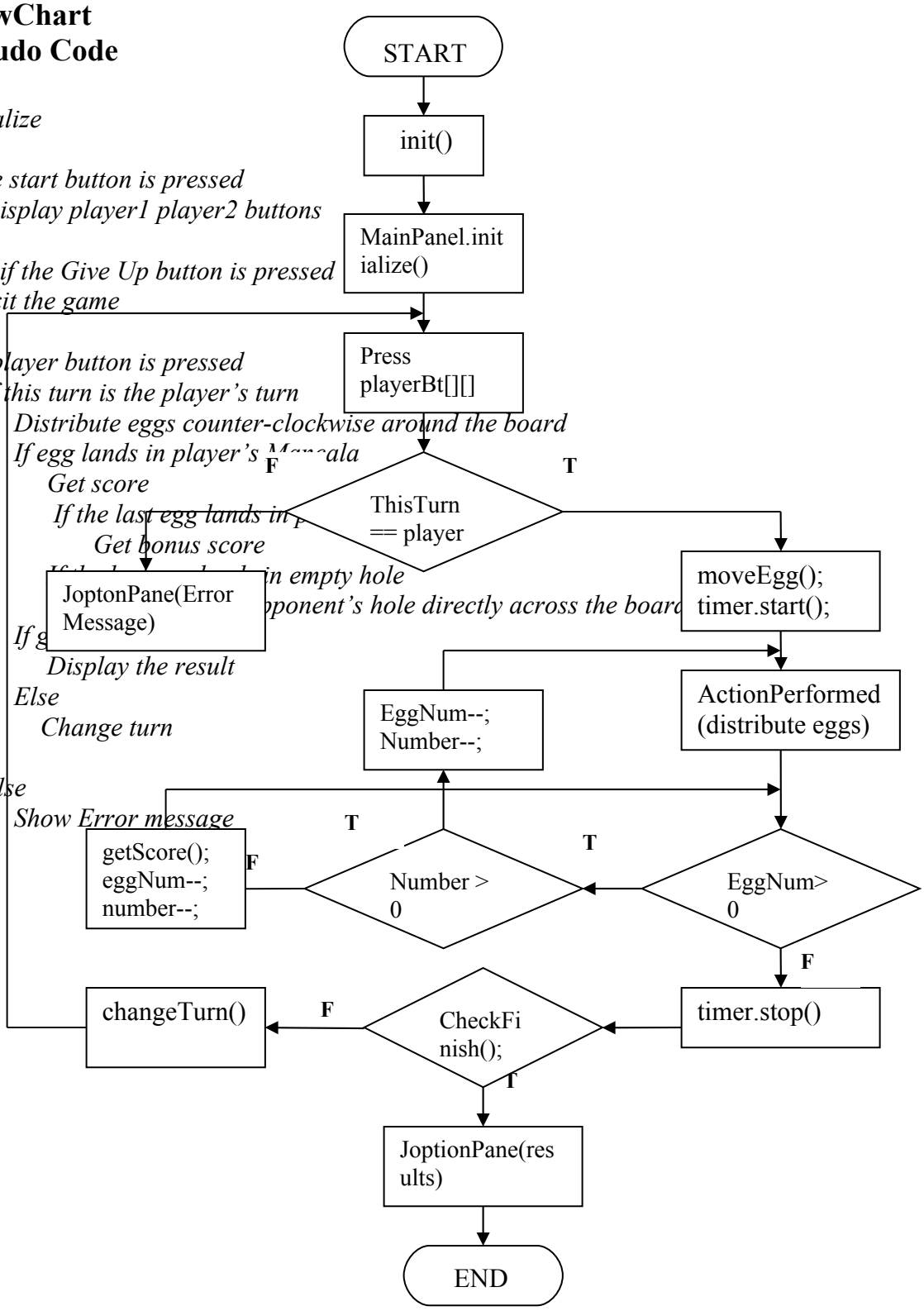
Initialize

If the start button is pressed
Display player1 player2 buttons

Else if the Give Up button is pressed
Exit the game

If a player button is pressed
If this turn is the player's turn
Distribute eggs counter-clockwise around the board
If egg lands in player's Mancala
Get score
If the last egg lands in p
Get bonus score
If the last egg lands in empty hole
Get opponent's hole directly across the board
If g
Display the result
Else
Change turn

Else
Show Error message



Major Variables

Type	Variable Name	Description	Location
Integer	PlayerEggNum[][]	The number of eggs in holes	Class gameBoardpublic -void actionPerformed(ActionEvent e)
	Socre[]	Each player's score array	GetScore()
	ChickenNum[]	The number of chickens in Mancala	GetScore()
	RemaindEggs[]	The number of eggs remaining in holes after finised	isFinished()
	thisTurn	Have the player number Whose turn now is	Class gameBoardpublic -void actionPerformed(ActionEvent e) changeTurn()
	number	The number of holes	MoveEgg(), opponent(), actionPerformed(ActionEvent e)
	eggNum	The number eggs when distributing	MoveEgg(), actionPerformed(ActionEvent e)
	player	Player zone eggs are distributed	MoveEgg(), actionPerformed(ActionEvent e)
	getScore	The score	actionPerformed(ActionEvent e)
JButton	playerBt[][]	The JButton array that contains each hole button	actionPerformed(ActionEvent e)
	start	The start button	Class Mancala – ActionPerformed(ActionEvent e)
	giveUp	The give up button	Class Mancala – actionPerformed(ActionEvent e)
JPanel	TitlePanel	The panel that contains titleLabel	Class GameBoard – GameBoard()
	gamePanel	The panel that contains player buttons, labels…	Class GameBoard – GameBoard()
	scorePanel	The panel that contains scores	Class GameBoard – GameBoard()
	buttonPanel	The panel that contains start and giveup buttons	Class Mancala – init()
JLabel	PlayerLb[][]	The JLabel for number of eggs in each hole	actionPerformed(ActionEvent e)
	Chicken[]	The Jlabel for number of chickens of each player	GetScore()
JTextField	ScoreText[]	The textField for each player's score	GetScore()
Color	Color[]	The array of button colors	SetColors()

Icon	Egg[]	The Icon array for button images	SetImageFiles()
	No[]	The Icon array for number of eggs	SetImageFiles()
Timer	timer	The object of Timer class	MoveEgg()
Boolean	flag	Check the button is the first button clicked	actionPerformed(ActionEvent e)
GameBoard	mainPanel	The object of GameBoard class	Class Mancala – init()
BorderLayout	border	The layout for GameBoard	Class GameBoard – GameBoard()
GirdBagLayout	gbLayout	The layout for gamePanel	Class GameBoard – GameBoard()

Java Constructs

Chapter	New syntax and Keywords	Location
2	public class (class definition)	Public class Mancala Public class ButtonHandler Public class GameBoard
	JoptionPane.showMessageDialog JOptionPane.INFORMATION_MESSAGE	JOptionPane.showMessageDialog(null, scroll, "How to play, JOptionPane.INFORMATION_MESSAGE);
	Import statement Javax.swing package	Iimport javax.swing.*;
	If structure	if (e.getSource() == playerBt[i][j]) {}
	Equality operators and relational operators (==, !=, >, <, >=, <=)	for (int i=0; i<2; i++)
	Arithmetic operators (+, -, *, /)	score[0] += (remainedEggs[0]*25);
	Int primitive type	Int score[]
	String	String output
	Void keyword	Void init()
3	Extends JApplet	Public class Project extends JApplet
	Appletviewer, HTML tag	Mancala.html
	Init()	Public void init()
	Java.awt package	Import java.awt.*;
	Java.awt.event package	Import java.awt.event.*;
4	If/else selection structure	If (e.getSource() == start) {} Else {}
	Assignment operator (+=)	score[thisTurn] += 25;
	Increment operator (++)	chickenNum[thisTurn]++;
	While repetition structure	while ((text = r.readLine()) != null)
5	For repetition structure	For (int i=0; i<2; i++) {}
	JTextArea	output = new JTextArea(15, 35);

	Break	if (e.getSource() == playerBt[i][j]) {
		player = i;
		number = j;
		break;

	Logical operators (true, false)	Flag = true;
6	Container c = getContentPane();	Container c = getContentPane();
	Add	c.add(mainPanel, BorderLayout.CENTER);
	setText	whoseTurn.setText("Player 1's Turn!");
	setEditable	scoreText[i].setEditable(false);
	Return	Return true;
	ActionListener	Public class ButtonHandler implements ActionListener
	actionPerformed(ActionEvent e)	Public void actionPerformed(ActionEvent e)
	JButton	PlayerBt[][] = new Jbutton[I][j]; giveUp = new JButton("Give up");
	JLabel	PlayerLb[][] = new Jlabel[I][j]; PlayerLabel1 = new JLabel("Player 1");
	JTextField	scoreText[] = new JTextField[2];
	setLayout	gamePanel.setLayout(gbLayout);
	addActionListener(this)	playerBt[i][j].addActionListener(handler);
	Method	Public Boolean isFinished() Public void setImageFiles()
7	Array	PlayerBt[][] = new JButton[2][6]; Score[] = new int[2];
	Font.BOLD, Font.ITALIC, Font.PLAIN	Font f = new Font("Arial", Font.PLAIN, 14);
	setFont	output.setFont(f);
8	Instance of a class	mainPanel = new GameBoard();
	Dot(.) operator	MainPanel.initialize();
	private	Private Container c;
9	Show method	applicationWindow.show();
	WindowAdapter	new WindowAdapter()
	WindowClosing method	public void windowClosing(WindowEvent e)
	WindowEvent class	public void windowClosing(WindowEvent e)
	WindowListener interface	applicationWindow.addWindowListener
10	Append method	buf.append(text + "\n");
	StringBuffer	StringBuffer buf = new StringBuffer();
11	setBackground	start.setBackground(color1);
	Color	Color color[] = new color[15];
12	BorderLayout	private BorderLayout border;
	Jpanel class	private JPanel buttonPanel
	Icon	private Icon egg[] = new Icon[16];
	getSource	if (e.getSource() == start)
	ImageIcon	egg[i] = new ImageIcon("images/egg" + i + ".gif");
	setEnabled	start.setEnabled(false);

	SetIcon	title.setIcon(titleIcon);
13	GridBagLayout	private GridBagLayout gbLayout;
	GridBagConstraints.NONE	gbConstraints.fill = GridBagConstraints.NONE;

	Gridheight, gridwidth, gridx, gridy	gbConstraints.gridx = column;
		gbConstraints.gridy = row;
		gbConstraints.gridwidth = width;
		gbConstraints.gridheight = height;

	JMenu class	JMenuBar bar = new JMenuBar();
	JMenuBar class	JMenuBar bar = new JMenuBar();
	JMenuItem class	JMenuItem newItem = new JMenuItem("New");
	setConstraints	gbLayout.setConstraints(com, gbConstraints);
	setJMenuBar	setJMenuBar(bar);
14	Catch (Exception e)	catch (IOException e2)
	IOException	catch (IOException e2)
	Try block	try {RandomAccessFile r = new RandomAccessFile(name, "r"); …. }
16	File class	File name = new File("help.txt");
	.gif file	egg[i] = new ImageIcon("images/egg" + i + ".gif");
	Start method	timer.start();
	Stop method	Timer.stop();
	Timer class	Timer timer = new Timer(1000, this);
17	RandomAccessFile	RandomAccessFile r = new RandomAccessFile(name, "r");
	readLine method	while ((text = r.readLine()) != null)

Reference

1. textbook
2. API / Languages (Java 2 Platform API Specification)
3. www.hangame.com (Korean game cite – flash games)

Program

```
// Mancala Game - Mancala class(main)

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.text.DecimalFormat;
import java.io.*;

public class Mancala extends JApplet {
   private GameBoard mainPanel;
   private JPanel buttonPanel;
   private JButton start, giveup;
   private Container c;

   // initialization
   public void init()
   {
      buttonPanel = new JPanel();
      mainPanel = new GameBoard();

      Color color1 = new Color(123, 127, 185);
      Color color2 = new Color(212, 124, 170);
      start = new JButton("START!");
      giveup = new JButton("Give Up!");
      start.setBackground(color1);
      giveup.setBackground(color2);

      ButtonHandler handler = new ButtonHandler();
      start.addActionListener(handler);
      giveup.addActionListener(handler);
      buttonPanel.add(start);
      buttonPanel.add(giveup);

      c = getContentPane();
      c.add(mainPanel, BorderLayout.CENTER);
      c.add(buttonPanel, BorderLayout.SOUTH);
   }
```