**Computer Science 111**

Computer Science with Java I Fall, 2016

Lab Report – Week 10 - User-Defined Classes Assignment.

Abraham Schultz CSI 111 900 Fall 2016

**Assignment Analysis and Design**

This week’s assignment was to create a user defined class of object called “player” to be used in conjunction with a monopoly board class. The goal was to get the player to move correctly around the monopoly board deducting rent as needed. Additionally, the program should tell the user where on the board they are, what their bank balance is, and what the dice rolled. The program finishes when the players bank balance is 0 or less. Most of the guts of my code is in my player class, held within the four set methods. The main method of the monopoly.java file then calls on these methods to accomplish the task. The set location method is where the die rolling occurs, the set balance method is where the rent is deducted, the set token method allows the user to choose one of eight classic monopoly token, and the set name method allows user to enter name.

I did talk with one of the other students in the class about some issues I was having his name is Mourad El Issaoui. He even let me look at some of his code. That being said, we took two very different approaches. I am curious which one is better code.

Below is an outline I used as a starting point for the project.

*/\**

*attributes*

*- name: String*

*- token: String*

*- location: int*

*- balance: int*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*methods*

*+ Player(): void*

*+ Player(String, String, int, int): void*

*+ getName(): String*

*+ getToken(): String*

*+ getLocation() int*

*+ getBalance() int*

*+ setName(String): void*

*+ setToken(String): void*

*+ setLocation(int): void*

*+ setBalance(int): void*

*+ toString:() String (name, token, location*

*\*/*

*}*



**Assignment Code**

/\*

Monopoly plaer class

\*Class for a player object in monopoly

\*for CSCI 111

\*last edited november 17h 1:39pm

@author Abraham Schultz

\*/

package monopoly;

import java.util.Random;

import java.util.Scanner;

public class Player {

static boolean toString;

// list of attributes of player class

private String name; //name of player

private String token; // token player is using

private int location; // location of token on board

private int balance; // how much money player has

// CONSTRUCTORS

public Player() {

name = "";

token = "";

location = 0;

balance = 1500;

}// end player

public Player(String name, String token, int location, int balance) {

this.name = name;

this.token = token;

this.location = location;

this.balance = balance;

} // end player(args)

//ACCESSORS

public String getName() {

return name;

} //end get name

public String getToken() {

return token;

}//end get token

public int getLocation() {

return location;

}//end get location

public int getBalance() {

return balance;

}//end get balance

//MUTATORS

public void setName() {// code which asks for player name

String name;

Scanner inputName = new Scanner(System.in);// instance of scanner

System.out.println("Please enter the name of this player");

name = inputName.nextLine();

System.out.println("hello " + name);

this.name = name;

}// end set name

public void setToken() {

// code which asks user to pick a token iron, race car, thimble, shoe, top hat, battleship,The Scottie dog, wheelbarrow

String token = "";

int tokenChoice;

Scanner inputToken = new Scanner(System.in);//instance of scanner

System.out.printf("please choose a token by entering the number that is next to the token of your choice.");

System.out.printf("%n%5s%n%5s%n%5s%n%9s%n%9s%n%15s%n%15s%n%15s%n", " 1 = "

+ "Iron", " 2 = Race Car", " 3 = Thimble", "4 = Shoe", " 5 = Top Hat", " 6 = Battleship", " 7 = Scottie dog", " 8 = Wheelbarrow");

do {//loop that continues until user enters a number bewtween 1-8

System.out.println("please enter a number bewtween 1-8");

tokenChoice = inputToken.nextInt();

} while (tokenChoice <= 0 || tokenChoice >= 9);

if (tokenChoice == 1) {

System.out.println("you have selected the Iron as your token");

token = "Iron";

}

if (tokenChoice == 2) {

System.out.println("you have selected the Race Car as your token");

token = "Race Car";

}

if (tokenChoice == 3) {

System.out.println("you have selected the Thimble as your token");

token = "Thimble";

}

if (tokenChoice == 4) {

System.out.println("you have selected the Shoe as your token");

token = "Shoe";

}

if (tokenChoice == 5) {

System.out.println("you have selected the Top Hat as your token");

token = "Top Hat";

}

if (tokenChoice == 6) {

System.out.println("you have selected the Battleship as your token");

token = "Battleship";

}

if (tokenChoice == 7) {

System.out.println("you have selected the Scottie Dog as your token");

token = "Scottie Dog";

}

if (tokenChoice == 8) {

System.out.println("you have selected the Wheelbarrow as your token");

token = "Wheelbarrow";

}

this.token = token;

}// end set token

public void setLocation(int currentLocation) {// code which rolls dice and sets location on board

// variables

int newLocation;

int totalRoll;

int die1;

int die2;

Random random = new Random();// instance of random number generator

die1 = random.nextInt(6) + 1;// finds a number between 0-5, so we +1 to get 1-6

System.out.println("die#1 rolled a " + die1);

die2 = random.nextInt(6) + 1;

System.out.println("die#2 rolled a " + die2);

totalRoll = die1 + die2;

newLocation = totalRoll + currentLocation;

System.out.println("your total roll is: " + totalRoll);

if (newLocation > 39) {

newLocation = newLocation - 40;

}

this.location = newLocation;

}// end set location

public void setBalance(int rent) {// code which subtracts rent from total balance

int currentBalance = this.balance;

int newBalance = currentBalance - rent;

System.out.println("your new bank balance is: " + newBalance + "$");

this.balance = newBalance;

}// end set balance

public String toString() {// to string method

String info;

info = ("the players name is " + name + ", the players token is " + token

+ ", the players board location is " + location + ", the players bank balance is " + balance);

return info;

}// end to string

}\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\* Monopoly.java

\*

\* CSCI 111 Fall 2013

\* last edited November 2, 2013 by C. Herbert

\*

\* This package contains code that can be used as the basis of a monopoly game

\* It has a class of BoardSquares for the board squares in a Monopoly game,

\* and a main program that puts the squares into an array.

\*

\* The main method has code to test the program by printing the data from the array

\*

\* This is for teaching purposes only.

\* Monopoly and the names and images used in Monopoly are

\* registered trademarks of Parker Brothers, Hasbro, and others.

\*/

package monopoly;

import java.util.Scanner;

/\*\*

\*

\* @author cherbert

\*/

public class Monopoly {

/\*\*

\* @param args the command line arguments

\* @throws java.lang.Exception

\*/

public static void main(String[] args) throws Exception {

BoardSquare[] square = new BoardSquare[40]; // array of 40 monopoly squares

Player player1 = new Player();// instance of player class is created

int currentSquare = player1.getLocation(); // intialized at 0

int newSquare; // variable to hold value of newsquare that player lands on

loadArray(square);

player1.setName();// method asks for players name

player1.setToken();// method ask for player to choose token

System.out.println(player1.getName() + " is at the " + square[currentSquare].getName() + " square"); // prints out starting location

do {

promptEnterKey(); // method to hit enter to continue

System.out.println("Players current bank balance is: " + player1.getBalance() + "$");// displays players balance before each turn

player1.setLocation(player1.getLocation()); // method rolls dice

newSquare = player1.getLocation();

System.out.println(player1.getName() + "'s " + player1.getToken() + " has landed on " + square[newSquare].getName()); // tells user where they are

if (square[newSquare].getRent() > 0) {

System.out.println(" the rent is " + square[newSquare].getRent() + "$");

} // if there is a rent , tell the user

player1.setBalance(square[newSquare].getRent());

} while (player1.getBalance() > 0);

System.out.println("you have gone bankrupt!!!");

} // main()

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// method to load the BoardSquare array from a data file

public static void loadArray(BoardSquare[] square) throws Exception {

int i; // a loop counter

// declare temporary variables to hold BoardSquare properties read from a file

String inName;

String inType;

int inPrice;

int inRent;

String inColor;

// Create a File class object linked to the name of the file to be read

java.io.File squareFile = new java.io.File("squares.txt");

// Create a Scanner named infile to read the input stream from the file

Scanner infile = new Scanner(squareFile);

/\* This loop reads data into the square array.

\* Each item of data is a separate line in the file.

\* There are 40 sets of data for the 40 squares.

\*/

for (i = 0; i < 40; i++) {

// read data from the file into temporary variables

// read Strings directly; parse integers

inName = infile.nextLine();

inType = infile.nextLine();

inPrice = Integer.parseInt(infile.nextLine());

inRent = Integer.parseInt(infile.nextLine());

inColor = infile.nextLine();

// intialze each square with the BoardSquare constructor

square[i] = new BoardSquare(inName, inType, inPrice, inRent, inColor);

} // end for

infile.close();

} // endLoadArray

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public static void promptEnterKey() {// method for hitting enter key to continue

System.out.println("Press \"ENTER\" to roll dice");

Scanner scanner = new Scanner(System.in); //scanner delcared

scanner.nextLine();

}

} // end class Monopoly

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*



**Assignment Testing**

The approach I took was to first create the skeleton of the player class, with empty methods for each variable I needed. Once I had that, I started by attempting to get the setter methods of the player class to function each individually. I used System.out.print extensively as a testing tool. I would take the variable I was testing and print them out at various points in the code to see what was happening. This was particularly useful when I was working on getting the current square variable to reset correctly after passing 39.

**Assignment Evaluation**

This project taught me a lot about encapsulation, and how different code can be tied in with each other. I struggled with putting together my main method and getting my player to move around the board in the correct fashion. For a while my player seemed to be moving around the board randomly. The easy part was putting together my player class of object. I like that the project was something that most people are familiar with, which made it a lot more fun. If I were to do this project again I would probably take a very different approach, then what I did here. I was not sure which was better, to include my logic in the main method or in the player class methods. In my original plans, I put the logic in the player class, and that what I did here. But in talking with other class mates I realized I could have done this project an entirely different way, by putting the logic in the monopoly.java file and creating separate methods therein. I think my way was probably not the most elegant solution, but I am satisfied with how the results look when the program runs.