Jeremy Fanuel  
 218116757

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package hmmmm;

import java.util.ArrayList;

import java.util.Scanner;

import java.io.\*;

import java.util.Random;

import java.util.logging.Level;

import java.util.logging.Logger;

public class Hmmmm {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

// TODO code application logic here

Scanner sc = new Scanner(System.in);

int menu\_utama;

String[][] Map = new String[12][22];

ArrayList<Player> p = new ArrayList<>();

ArrayList<Game> g = new ArrayList<>();

boolean ingame = false;

File f = new File("save.data");

if(f.exists())

{

try

{

FileInputStream in = new FileInputStream("save.data");

ObjectInputStream obj = new ObjectInputStream(in);

p = (ArrayList) obj.readObject();

obj.close();

}

catch(IOException ex)

{

ex.printStackTrace();

}

catch(ClassNotFoundException err)

{

err.printStackTrace();

}

//print

/\*for(Player i : p)

{

System.out.println(i);

}\*/

}

while(true)

{

System.out.print("Menu\n" +

"======================\n"

+ "1. Register\n"

+ "2. Login\n"

+ "3. Save\n"

+ "4. Highscore\n"

+ ">>>");

menu\_utama = sc.nextInt();

if(menu\_utama == 1)

{

System.out.print("Username: ");

String username = sc.next();

System.out.print("Password: ");

String password = sc.next();

System.out.print("Confirm Pasword: ");

String confirm\_password = sc.next();

if(confirm\_password.equals(password))

{

boolean ada = false;

if(p.size()>0)

{

for (int i = 0; i < p.size(); i++) {

if(p.get(i).getUsername().equals(username))

{

ada = true;

}

}

if(!ada)

{

System.out.println("Registration successful");

p.add(new Player(username,password));

}

else

{

System.out.println("This username has existed");

}

}

else

{

System.out.println("Registration successful");

p.add(new Player(username,password));

}

}

else

{

System.out.println("Both password & confirm password has to be identical ");

}

}

else if (menu\_utama == 2)

{

System.out.print("Username: ");

String username = sc.next();

System.out.print("Password: ");

String password = sc.next();

int temp = -1;

boolean ada = false;

if(p.size()>0)

{

for (int i = 0; i < p.size(); i++) {

if(p.get(i).getUsername().equals(username)&&p.get(i).getPassword().equals(password))

{

temp = i;

System.out.println(temp);

}

else

{

ada = true;

}

}

if(ada && temp == -1)

{

System.out.println("Incorrect username or password");

}

else

{

g.add(new Game(p.get(temp).getUsername()));

int ctr = -1;

for (int i = 0; i < g.size(); i++) {

if(g.get(i).getUsername().equals(p.get(temp).getUsername()))

{

ctr = i;

}

}

ingame = true;

while(ingame)

{

if(g.get(ctr).getMoves()<=0)

{

ingame = false;

}

for (int i = 0; i < 12; i++) {

for (int j = 0; j < 22; j++) {

if(i==0||i==11||j==0||j==21)

{ Map[i][j] = "#";}

else

{ Map[i][j] = " ";}

}

}

Map[g.get(ctr).getY()][g.get(ctr).getX()] = "P";

System.out.println("Score: " + g.get(ctr).getScore());

System.out.println("Moves: " + g.get(ctr).getMoves());

for (int i = 0; i < 12; i++) {

for (int j = 0; j < 22; j++) {

System.out.print(Map[i][j]);

}

System.out.println();

}

System.out.println("Press t to teleport");

System.out.println("Press e to exit");

String move = sc.next();

if(move.equalsIgnoreCase("e"))

{

ingame = false;

}

}

}

}

else

{

System.out.println("Incorrect username or password");

}

}

else if (menu\_utama == 3)

{

try

{

FileOutputStream save = new FileOutputStream("save.data");

ObjectOutputStream obj = new ObjectOutputStream(save);

obj.writeObject(p);

obj.writeObject(g);

obj.close();

System.out.println("Success save");

}

catch(IOException e)

{

e.printStackTrace();

}

}

else if (menu\_utama == 4)

{

}

}

}

}

/\*

String[][] Map = new String[12][22];

for (int i = 0; i < 12; i++) {

for (int j = 0; j < 22; j++) {

if(i==0||i==11||j==0||j==21)

{

Map[i][j] = "#";

}

else

{

Map[i][j] = " ";

}

}

}

for (int i = 0; i < 12; i++) {

for (int j = 0; j < 22; j++) {

System.out.print(Map[i][j]);

}

System.out.println();

}

\*/

class Player implements Serializable

{

private static final long serialVersionUID = 6529685098267757690L;

protected String username;

protected String password;

public Player(String nama, String password) {

this.username = nama;

this.password = password;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

@Override

public String toString() {

return "Player{" + "username=" + username + ", password=" + password + '}';

}

}

class Game implements Serializable

{

private static final long serialVersionUID = 6529685098267757690L;

protected String username;

Random r = new Random();

protected int x,y;

protected int Score,Moves;

public Game(String username) {

this.username = username;

this.x = 1 + r.nextInt(20);

this.y = 1 + r.nextInt(10);

this.Score = 0;

this.Moves = 40;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public int getX() {

return x;

}

public void setX(int x) {

this.x = x;

}

public int getY() {

return y;

}

public void setY(int y) {

this.y = y;

}

public int getScore() {

return Score;

}

public void setScore(int Score) {

this.Score = Score;

}

public int getMoves() {

return Moves;

}

public void setMoves(int Moves) {

this.Moves = Moves;

}

public void Move(String s)

{

if(s.equalsIgnoreCase("w") && this.y < 10)

{

this.y -= 1;

this.Moves -= 1;

}

else if(s.equalsIgnoreCase("s") && this.y > 0)

{

this.y += 1;

this.Moves -= 1;

}

else if(s.equalsIgnoreCase("a") && this.x > 0 )

{

this.x -= 1;

this.Moves -= 1;

}

else if(s.equalsIgnoreCase("d") && this.x < 20)

{

this.x += 1;

this.Moves -= 1;

}

}

}

abstract class makanan

{

abstract public void move();

abstract public void transfer();

}

class makanan\_small extends makanan

{

@Override

public void move() {

throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.

}

@Override

public void transfer() {

throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.

}

}

class makanan\_medium extends makanan

{

@Override

public void move() {

throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.

}

@Override

public void transfer() {

throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.

}

}

class makanan\_large extends makanan

{

@Override

public void move() {

throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.

}

@Override

public void transfer() {

throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.

}

}