C

OOP Final – Part 2 ian.-11

[Write on your own sheets of paper. On each sheet, on the front, write in the top right corner your **lastname, firstname(s), group id and** (C). Leave a 2cm margin at the left on the front page / at the right on the back for stapling. Open book. Time: 90 minutes]

Fluffy the Cat Looking for a Hat

Fluffy the cat is moving on a board of 10 by 40 squares. The board configuration (unknown to Fluffy and the user controlling her) is read from a text file whose name is given on the command line. Fluffy is looking for a hat i.e. an area of 3 by 3 squares representing a shape composed of a letter 'H' and spaces (see figure below). On the board there are walls marked with stars (character '*') and corridors, marked with spaces (blanks). Fluffy is able to see contents of the neighboring squares surrounding her (all 8 squares situated at N, NE, E, SE, S, SW, W, and NW). Fluffy is controlled by the user and accepts as commands the direction to move as one or two letters, read from `System.in` (the cardinal directions N, NE, E, etc). Fluffy also has bad memory, so she does not remember what she saw before, so at all times, the user can see only what Fluffy sees. Anyway, when it is in the center position of the hat (remember, letter 'H' 3 by 3 square configuration), she says `"Miao, I bet I got my hat"` at `System.out` and the game ends. Besides the cat move commands (**`N, NE, E, SE, S, SW, W,`** and **`NW`**), there are other two commands: `S` – for Save game, and **`R`** – for Restore game. When such a command is read from **`System.in`** all objects in the game are saved/restored to/from file `FluffyHat.dat`.

Example configuration file `FluffyWorld.txt`. The contents of this file IS NOT SHOWN to the user. Fluffy is initially on an empty square and is marked in the configuration with letter `F`. Notice the two hats (H letter configurations). If Fluffy gets in the center of either hats she 'miaos':

```
****************************************
          *                   **********
********* * ***************** *        *
*       * * *      H H      *  *********
*       * * * *    HHH    * * *   H H  *
*       * * * ***  H H **** * *   HHH  *
*       *   *F  *******   *   *   H H  *
************* **************************
          *                   **********
********* * ***************** *        *
```

What Fluffy sees (at what is printed after each command on `System.out`) from its location in the figure above. Note that the prompt for user command is "> " (a greater than sign and a blank):

```
* *
*F
* *
>
```

After the command 'N' fluffy will see (the command input is also included)

```
> N
* *
*F*
*
>
```

Draw the class diagram and develop a java standalone application to simulate this game. Don't forget to briefly document it.

```
/**
 * Write a description of class Game here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class Game
{
    public static void main()
    {
        Board dd=new Board();
        int [] aux=new int[2];
        System.out.println("Press 4 for new game!");
        aux=IOClass.readCommand();
        if(aux[0]==3) dd=(Board)IOClass.loadGame();
        else dd.setMaze(IOClass.readInitialConfig());
        if(dd==null)dd.setMaze(IOClass.readInitialConfig());
        boolean ok=true;
        while((!dd.gameOver())&&(ok==true)){
            dd.view();
            aux=IOClass.readCommand();
            System.out.println(aux[0]+" "+aux[1]);
            if(aux[0]==2){ IOClass.saveGame(dd); ok=false;};
            if(aux[0]==3) dd=(Board)IOClass.loadGame();
            if(aux[0]==4) System.exit(0);;
            if(aux[0]<2) dd.setMove(aux);

        }

    }
}



/**
 * Write a description of class Cat here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
import java.io.Serializable;
public class Cat implements Serializable
{
   private int catx=0;
   private int caty=0;
   public Cat(int a, int b){
    catx=a;
    caty=b;
    }
    public void setCoord(int a,int b)
    {
      catx=a;
      caty=b;
    }
    public int [] getCoord()
    {
    int[] ret=new int[2];
    ret[0]=catx;
    ret[1]=caty;
    return ret;

    }
}






/**
 * Write a description of class Board here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
import java.lang.ArrayIndexOutOfBoundsException;
import java.io.Serializable;
public class Board implements Serializable
{
    private char[][] maze=new char[10][40];
    public Cat cat=new Cat(0,0);
    private int catx=0;
    private int caty=0;
    private boolean gameO=false;
    private char occupied=' ';
    public Board()
    {
        gameO=false;
    }


    public void setMaze(char [][] c)
    {
        maze=c;
        for(int i=0;i<10;i++)
            for(int j=0;j<40;j++)
                if(maze[i][j]=='F'){catx=i; caty=j;};
        cat.setCoord(catx,caty);
    }

    public boolean gameOver()
    {
        if(gameO)System.out.println("Mioa I bet I found my hat!");
        return gameO;
    }

    public boolean checkHat(int a, int b){
```

```
    boolean ok=false;
    System.out.println("Checking hat!");
    try{
        if((maze[a-1][b-1]=='H')&&(maze[a-1][b+1]=='H')&&(maze[a+1][b-1]=='H')&&(maze[a+1][b+1]=='H'))
        //&&(maze[a][b+1]=='H')&&(maze[a][b-1]=='H'))&&(maze[a+1][b]==' ')&&(maze[a-1][b]==' '))
            ok=true;
            System.out.println("In the hat!");
    }catch(ArrayIndexOutOfBoundsException e){System.out.println("ERROR HAT SEARCH"); }
    return ok;
}


public void moveCat(int a, int b){
    int[] aux2=cat.getCoord();
    int ax=aux2[0];
    int ay=aux2[1];
    System.out.println("Moving cat from "+ax+" "+ay+" to "+a+" "+b);
    maze[ax][ay]=occupied;
    maze[a][b]='F';
    cat.setCoord(a,b);
}


public void setMove(int[] aux)
{
    int[] aux2=cat.getCoord();
    int ax=aux2[0];
    int ay=aux2[1];
    ax+=aux[0];
    ay+=aux[1];
    if((ax>=0)&&(ax<10)&&(ay>=0)&&(ay<40))
    {
        if(maze[ax][ay]!='*')
        {
            if(maze[ax][ay]==' ') {

                moveCat(ax,ay);
                occupied=' ';
            }else
            {//maybe is a H

                gameO=checkHat(ax,ay);
                if(!gameO) moveCat(ax,ay);
                 occupied='H';
            }
        }
        else System.out.println("I hit a wall!");
    }

}


public void view()
{
        int[] aux2=cat.getCoord();
        int a=aux2[0];
        int b=aux2[1];
        System.out.println("What I see now:");
        try{
        System.out.println((char)maze[a-1][b-1]+""+(char)maze[a-1][b]+""+(char)maze[a-1][b+1]);
        System.out.println((char)maze[a][b-1]+""+(char)maze[a][b]+""+(char)maze[a][b+1]);
        System.out.println((char)maze[a+1][b-1]+""+(char)maze[a+1][b]+""+(char)maze[a+1][b+1]);
        }catch(ArrayIndexOutOfBoundsException e){
        System.out.println("Out of range!");
        }
        System.out.println("The entire board:");
        for(int i=0;i<10;i++){
            for(int j=0;j<40;j++)
                System.out.print((char)maze[i][j]);
            System.out.println("");
        }
    }
}
```

```
/**
 * Write a description of class IOClass here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
import java.util.*;
import java.lang.*;
import java.io.*;
public class IOClass
{
    public static char[][] readInitialConfig(){
        Scanner inr=new Scanner(System.in);
        System.out.print("Give input file name: file <<fluffy>> exists");
        String S=inr.nextLine();
        if(S.indexOf(".")<0)        S=S+".txt";
        BufferedReader in=null;
        String line=null;
        int i=0;
        char[][] ret=new char[10][40];
        try{
            in=new BufferedReader(new FileReader(S));
            while((line=in.readLine())!=null){
                ret[i]=line.toCharArray();
                i++;
                System.out.println(line);
            }
        }
        catch(IOException e)
    {
        System.out.println("File not open!");
    }
    finally
    {
        try{
            if(in!=null) in.close();
        }catch (IOException e2){}
    }
    return ret;
}


public static int[] readCommand(){
    Scanner inr=new Scanner(System.in);
    System.out.println("Enter a command N,NE,E,SE,S,SV,V,NV,SAVE,LOAD,EXIT :");
    System.out.print("> ");
    String line=inr.nextLine();
    System.out.println(line);
    int x=0; int y=0;
    if(line.equals("N")){x=-1; y=0;}
    if(line.equals("NE")){x=-1; y=1;}
    if(line.equals("E")){ y=1;}
    if(line.equals("SE")){x=1; y=1;}
    if(line.equals("S")){ x=1;}
    if(line.equals("SV")){x=1; y=-1;}
    if(line.equals("V")){ y=-1;}
    if(line.equals("NV")){x=-1; y=-1;}
    if(line.equals("SAVE")){x=2;    }
    if(line.equals("LOAD")){x=3;   }
    if(line.equals("EXIT")){x=4;   }
    int [] ret=new int[2];
    ret[0]=x;
    ret[1]=y;
    System.out.println(x+" "+y);
    return ret;
}


public static void saveGame(Object oo){
    FileOutputStream outputFile =null;
    try{
        outputFile = new FileOutputStream("fluffyhat.dat");
    }catch (FileNotFoundException e2){}
    ObjectOutputStream outputStream=null;
    try{
        outputStream = new ObjectOutputStream(outputFile);

        outputStream.writeObject(oo);
        outputStream.flush();
        outputStream.close();
    }catch (IOException e){}
}
```

```
    public static Object loadGame(){
        FileInputStream inputFile=null;
        ObjectInputStream inputStream=null;
        Object oo=null;
        try{
            inputFile = new FileInputStream("fluffyhat.dat");
        }    catch (FileNotFoundException e){}
        try{
            inputStream = new ObjectInputStream(inputFile);
            try{
                oo=inputStream.readObject();
            }catch(ClassNotFoundException e3){}
            inputStream.close();
        }catch (IOException e){}
        return oo;
    }
    public static void main(){

    readInitialConfig();
    }
}
```