HACETTEPE UNIVERSITY

DEPARTMENT OF COMPUTER ENGINEERING BBM104

Name : Mehmet Taha

Surname : Usta

Number : 21527472

E~mail : b21527472@cs.hacettepe.edu.tr

Subject : Abstract Classes & Interfaces

Programming Language : Java

1. What Do You Use

I created 14 class. I introduced the inputs to the system for solving problem .I started the necessary operations in the main.java class. Then I created a empty output file. I defined the required variables. Then readfile.java started reading the files with while loop.Objects were created by reading the files and stored in the arrayslists. When the object inputs to be read are finished, the input specified by the user starts to read.The necessary actions are taken and the result is recorded in Leaderboard.txt

2. Explain Your Data Structure

First , checks whether there is an argument in main.java, and determines the path according to the situation. Create a read file object and start doing the necessary operations in readfile.class. GameGrid takes place and fills 2D array with Gamegrid.class. Gamegrid.class creates jewels based on input. If it is an unidentified jewel (Gamegrid.class), ask the user for the type of jewel, score then create the object.

If the user specifies an argument, the necessary operations are completed in the while loop.If the lines in the argument file contain coordinates, the coordinates go into the Gamegrid object to perform the necessary operations.The methods defined in the Gamegrid take the necessary actions.If there is no appropriate action, print "no matching matches found in the specified direction.Enter new coordinate". If the lines in the argument file contain E,the total score is suppressed and the name is requested. The program terminates after printing the user's status after entering the name.

If the user does not specify an argument, the program requests continuous coordinates until the user writes E. Make the necessary actions to the user-entered coordinates in GameGrid.class. ıf the user enters E, the total score is suppressed and the name is requested. The program terminates after printing the user's status after entering the name.

<<Java Class>>
Main
(default package)

- Main()
- $ main(String[]):void

<<Java Class>>
leaderboard
(default package)

- leaderboard()
- okuma_yazma(String,int):void
- compareTo(ArrayList<Player>):int

<<Java Class>>
Gem
(default package)

- gemtype: String
- point: int
- coordx: int
- coordy: int
- Gem(String,int,int,int)
- getGemtype():String
- setGemtype(String):void
- getPoint():int
- setPoint(int):void
- getCoordx():int
- setCoordx(int):void
- getCoordy():int
- setCoordy(int):void

<<Java Class>>
Board
(default package)

- Board()
- getGem():Gem[][]
- setGem(Gem[][]):void
- remove(int,int,int,int,int,int):int
- update(int,int,int,int,int,int):void
- printboard():void

-gem
0..*

<<Java Class>>
readfile
(default package)

- readfile(Board,String[])
- readfile(Board)

<<Java Class>>
Gamegrid
(default package)

- lines: ArrayList<String[]>
- kazanilan_puan: int
- Gamegrid(Board)
- yerkont(int,int,Board):Boolean
- kontrol1(int,int,Board):void
- kontrol2(int,int,Board):void
- kontrol3(int,int,Board):void
- kontrol4(int,int,Board):void
- kontrol5(int,int,Board):void

<<Java Class>>
Player
(default package)

- name: String
- point: int
- Player(String,int)
- getName():String
- setName(String):void
- getPoint():int
- setPoint(int):void

<<Java Class>>
Diamond
(default package)

- Diamond(int,int)

<<Java Class>>
new_jewel
(default package)

- new_jewel(String,int,int,int)

<<Java Class>>
Wildcard
(default package)

- Wildcard(int,int)

<<Java Class>>
Triangle
(default package)

- Triangle(int,int)

<<Java Class>>
Mathematical_Symbols
(default package)

- Mathematical_Symbols(String,int,int)

<<Java Class>>
Square
(default package)

- Square(int,int)

<<Java Class>>
EmptyGem
(default package)

- EmptyGem(int,int)