2. Software Using Documentation

2.1. Software Usage

The program works by taking one file.

- “gameGrid.txt” an input file with initial grid and jewels.

There is another file(“leaderboard.txt”) that is created when the program first run. This file contains the names and scores of the players who played the game. The program writes the user's information to this file at every run.

3. Software Design Notes

3.1. Description of the Program

3.1.1. Problem

In this experiment, we are expected to recreate a simplified version of the game “Bejeweled”. The game consists of a grid of “jewels”. The goal is to find three jewels that are match in a row, column or diagonal by selecting the right coordinate. When this occurs, the three jewels are deleted, and other jewels fall from the top to fill in gaps. There are 9 different jewels:

•Diamond<D> : Diamond can match with other Diamonds only in diagonal coordinates.

•Square<S> : Square can match with its kind only in horizontal coordinate.

•Triangle<T> : Triangle can match with its kind only in vertical coordinate.

•Wildcard<W> : Wildcard can match any jewel in any direction.

Mathematical symbols match any other mathematical symbol jewel:

•<”/”> : In right diagonal •<”-“> : Search horizontaly •<”+”> : First horizontally then vertically

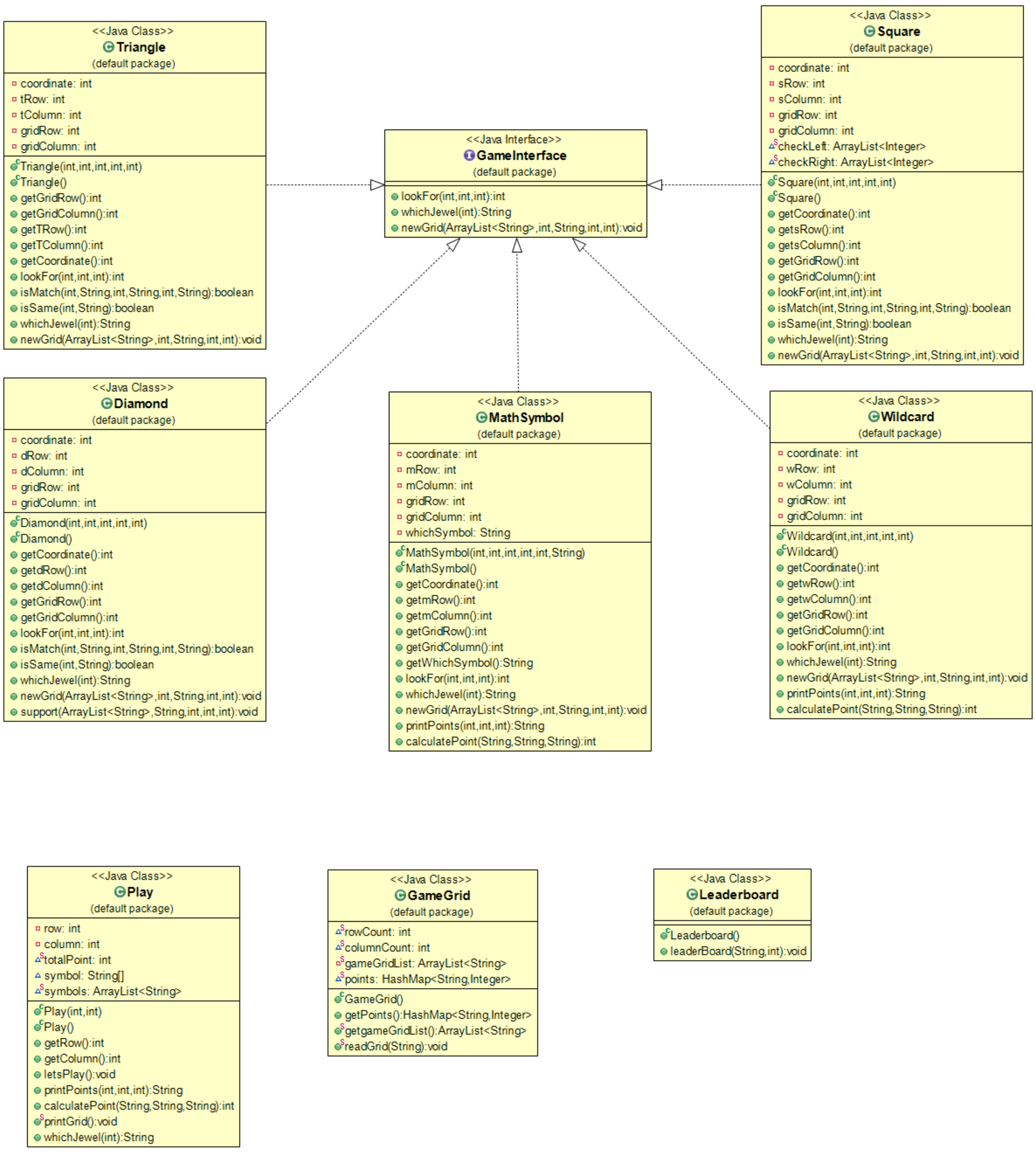
•<”\”> : In left diagonal •<”|”> : Search vertically

Each jewel has a different point. When the user finishes the game, the rank of the user is determined according to the point received.

3.1.2. Solution

We see that there is a grid and different jewels that can change according to the input file given. I use an ArrayList for the grid so that the program will work the same regardless of the size of the grid in the input file. Each of the jewels has a different mate and direction. In this case I used an interface and explained the way each jewel is mapped under its own class.

4. Class Diagram



<<Java Class>>
Triangle
(default package)
coordinate: int
tRow: int
tColumn: int
gridRow: int
gridColumn: int
Triangle(int,int,int,int,int)
Triangle()
getGridRow():int
getGridColumn():int
getTRow():int
getTColumn():int
getCoordinate():int
lookFor(int,int,int):int
isMatch(int,String,int,String,int,String):boolean
isSame(int,String):boolean
whichJewel(int):String
newGrid(ArrayList<String>,int,String,int,int):void
<<Java Interface>>
GameInterface
(default package)
lookFor(int,int,int):int
whichJewel(int):String
newGrid(ArrayList<String>,int,String,int,int):void
<<Java Class>>
Square
(default package)
coordinate: int
sRow: int
sColumn: int
gridRow: int
gridColumn: int
checkLeft: ArrayList<Integer>
checkRight: ArrayList<Integer>
Square(int,int,int,int,int)
Square()
getCoordinate():int
getsRow():int
getsColumn():int
getGridRow():int
getGridColumn():int
lookFor(int,int,int):int
isMatch(int,String,int,String,int,String):boolean
isSame(int,String):boolean
whichJewel(int):String
newGrid(ArrayList<String>,int,String,int,int):void
<<Java Class>>
Diamond
(default package)
coordinate: int
dRow: int
dColumn: int
gridRow: int
gridColumn: int
Diamond(int,int,int,int,int)
Diamond()
getCoordinate():int
getdRow():int
getdColumn():int
getGridRow():int
getGridColumn():int
lookFor(int,int,int):int
isMatch(int,String,int,String,int,String):boolean
isSame(int,String):boolean
whichJewel(int):String
newGrid(ArrayList<String>,int,String,int,int):void
support(ArrayList<String>,String,int,int,int):void
<<Java Class>>
MathSymbol
(default package)
coordinate: int
mRow: int
mColumn: int
gridRow: int
gridColumn: int
whichSymbol: String
MathSymbol(int,int,int,int,int,String)
MathSymbol()
getCoordinate():int
getmRow():int
getmColumn():int
getGridRow():int
getGridColumn():int
getWhichSymbol():String
lookFor(int,int,int):int
whichJewel(int):String
newGrid(ArrayList<String>,int,String,int,int):void
printPoints(int,int,int):String
calculatePoint(String,String,String):int
<<Java Class>>
Wildcard
(default package)
coordinate: int
wRow: int
wColumn: int
gridRow: int
gridColumn: int
Wildcard(int,int,int,int,int)
Wildcard()
getCoordinate():int
getwRow():int
getwColumn():int
getGridRow():int
getGridColumn():int
lookFor(int,int,int):int
whichJewel(int):String
newGrid(ArrayList<String>,int,String,int,int):void
printPoints(int,int,int):String
calculatePoint(String,String,String):int
<<Java Class>>
Play
(default package)
row: int
column: int
totalPoint: int
symbol: String[]
symbols: ArrayList<String>
Play(int,int)
Play()
getRow():int
getColumn():int
letsPlay():void
printPoints(int,int,int):String
calculatePoint(String,String,String):int
printGrid():void
whichJewel(int):String
<<Java Class>>
GameGrid
(default package)
rowCount: int
columnCount: int
gameGridList: ArrayList<String>
points: HashMap<String,Integer>
GameGrid()
getPoints():HashMap<String,Integer>
getgameGridList():ArrayList<String>
readGrid(String):void
<<Java Class>>
Leaderboard
(default package)
Leaderboard()
leaderBoard(String,int):void