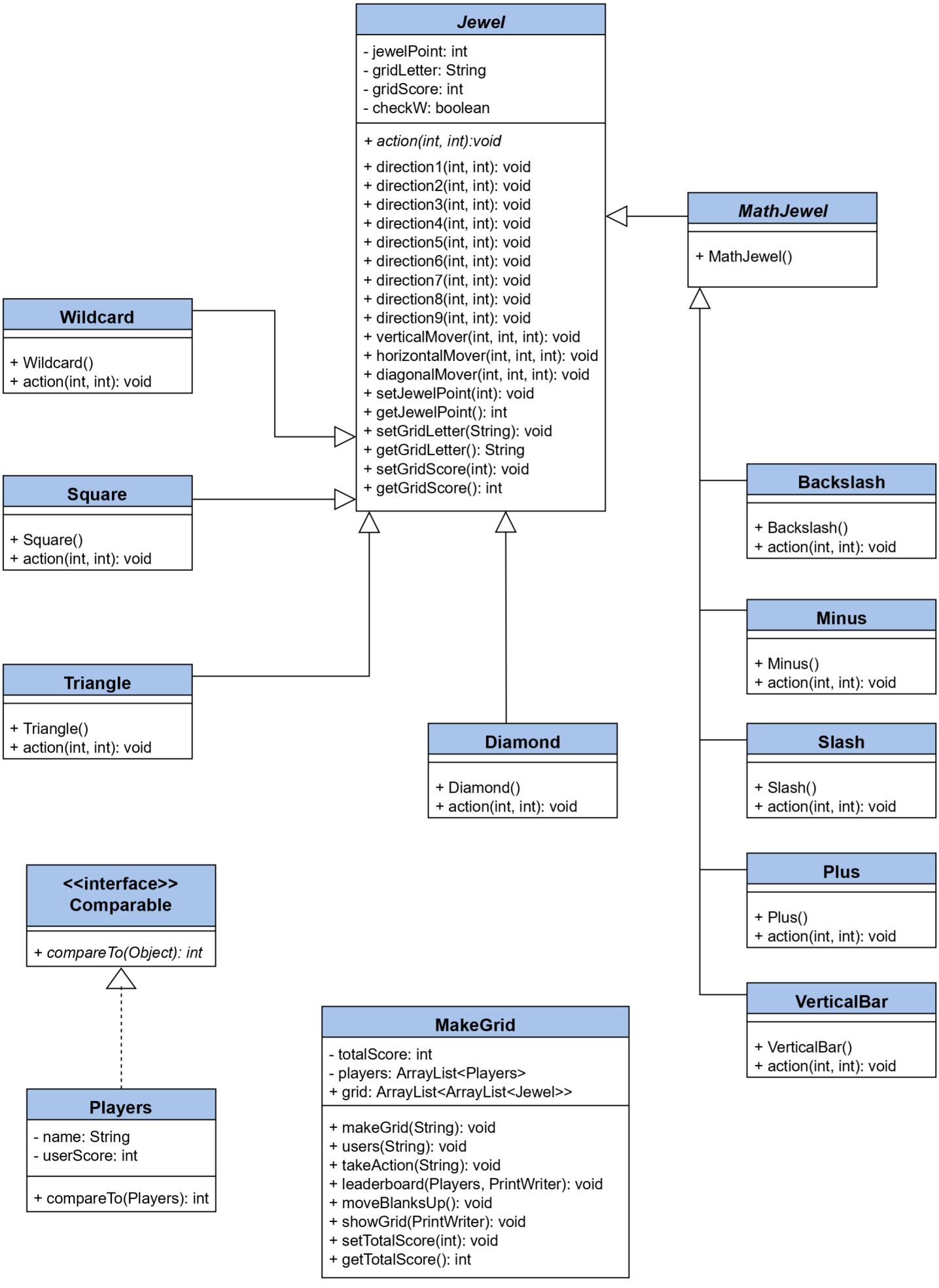
Jewel
- jewelPoint: int
- gridLetter: String
- gridScore: int
- checkW: boolean
+ action(int, int):void
+ direction1(int, int): void
+ direction2(int, int): void
+ direction3(int, int): void
+ direction4(int, int): void
+ direction5(int, int): void
+ direction6(int, int): void
+ direction7(int, int): void
+ direction8(int, int): void
+ direction9(int, int): void
+ verticalMover(int, int, int): void
+ horizontalMover(int, int, int): void
+ diagonalMover(int, int, int): void
+ setJewelPoint(int): void
+ getJewelPoint(): int
+ setGridLetter(String): void
+ getGridLetter(): String
+ setGridScore(int): void
+ getGridScore(): int
MathJewel
+ MathJewel()
Wildcard
+ Wildcard()
+ action(int, int): void
Square
+ Square()
+ action(int, int): void
Triangle
+ Triangle()
+ action(int, int): void
Diamond
+ Diamond()
+ action(int, int): void
Backslash
+ Backslash()
+ action(int, int): void
Minus
+ Minus()
+ action(int, int): void
Slash
+ Slash()
+ action(int, int): void
Plus
+ Plus()
+ action(int, int): void
VerticalBar
+ VerticalBar()
+ action(int, int): void
<<interface>>
Comparable
+ compareTo(Object): int
Players
- name: String
- userScore: int
+ compareTo(Players): int
MakeGrid
- totalScore: int
- players: ArrayList<Players>
+ grid: ArrayList<ArrayList<Jewel>>
+ makeGrid(String): void
+ users(String): void
+ takeAction(String): void
+ leaderboard(Players, PrintWriter): void
+ moveBlanksUp(): void
+ showGrid(PrintWriter): void
+ setTotalScore(int): void
+ getTotalScore(): int

The class that provides the whole image and layout is the MakeGrid class. It is used for clear main function. I used ‘makeGrid’ function to create the game grid, ‘users’ function to create the players list, ‘leaderboard’ function to sort the players, ‘takeAction’ function to check the triples with the coordinates taken from command.txt, and ‘moveBlanksUp’ function to provide the fall of the columns when gaps occur. The takeAction function calls the action function of the jewel type of the coordinate, so triplets are searched in appropriate directions. After action function is called, moveBlanksUp function is called, so that spaces are moved up after triplets are deleted.

Abstract Jewel class simply has jewel’s properties. Inside the Jewel class, there are direction functions to search for triples specific to each direction. Since each jewel is looking for a triple in different directions, I created an abstract ‘action’ method in the abstract Jewel class. This method is overrided according to the directions specified in each jewel.

Also, there are ‘mover’ functions inside the Jewel class, they set triplets to null. Mover functions are called after every direction function if match==2 (that means it found a triplet). If the chosen coordinate belongs to the Wildcard, I put chechW booelan to be checked again in the mover functions, because triplets shouldn’t be deleted if they contain mathematical symbols.

I created a new class for each jewel type and assigned its points and display letters on the grid in their constructors. Meanwhile, I created a new class ‘MathJewel’ extended Jewel, because the way of finding triplets of Mathematical Symbol jewels is different from others. It is abstract since there is no certain direction for action function (abstract method from Jewel). I have extended mathematical symbol jewels from MathJewel class.

Players class has players’ attributes such as name and score. It implements Comparable to be able to sort players and show the rank of the given user.