Introduction

In this assignment, students recreate a simplified version of the game "Bejeweled" using Java. You will practice using abstract classes and/or interfaces for creating extendible programs.
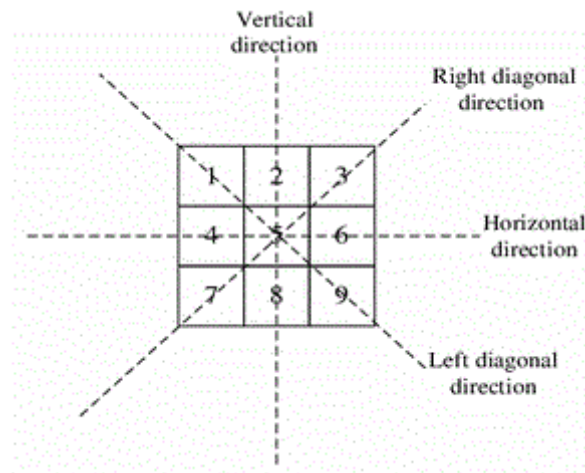
The Game

Bejeweled® is a puzzle game created by PopCap Games [1] in 2001. Various versions are available to play online, and on devices such as iPhone, iPad and Android. In this assignment, the game consists of a grid (of any size, e.g. 10x10) of "jewels". Some example jewels are Diamond (D), Square (S), Triangle (T) and Wildcard (W). 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. If the selected coordinate does not have any triple to match, then a warning message is displayed to the user and a new coordinate is requested.

Each jewel match in a different way. For the shape types defined as above, Diamond can match with other Diamonds only in diagonal coordinates. Square can match with its kind only in horizontal coordinate (x-axis). Triangle can match with its kind only in vertical coordinate (y-axis). Wildcard can match any jewel in any direction. Each jewel in a triple is worth a predetermined number of points. Diamond, Square, Triangle and Wildcard are scored as 30, 15, 15, 10 points respectively. So, a triple of all Diamonds will get 90 points, but a triple of 2 Diamonds and a Wildcard will be 70 points.

The coordinate specified by user can be either first or last item in triple. The triplet should be searched using the rules defined below.

- If the coordinate contains a D, the program searches triple first in left (1 and 9) and then in right diagonal direction (3 and 7),

- If the coordinate contains an S, the program searches triple first in left and then in right horizontal direction,

- If the coordinate points to a T, the program searches the triple first in upward and then in downward vertical direction,

- If the coordinate contains a W, the program starts searching from vertical (2 and 8) followed by horizontal (4 and 6) and finally diagonals left (1 and 9) and right (3 and 7).

Vertical direction

Right diagonal direction

Horizontal direction

Left diagonal direction

Using these jewels, let's follow a few steps of the game. Below example program prints the game grid first and ask user to select a coordinate. User selects index of array as [7][3] that means 8th row and 4th column in game grid. Selected coordinate corresponds to jewel T. In this case, program searches other two T's only in vertical coordinate to complete the triplet. Triplet of T jewels are deleted, and other jewels fall from top, leaving the top of the 4th column empty. When the user enters the coordinate 5 8, the wildcard will first search vertical direction and find a triple with two Ts. Note that the rules for matching are applied using the Jewel type in the selected coordinate cell. If we had selected "5 1" the Diamond wouldn't have matched the W in "4 0" because the Diamond allows only matches with Diamonds. But, in the case of "4 0" the W matched with the D "5 1" because we are applying the rules for Wildcard.

When the user enters E, first the games prints the total score and requests a username as input. The user score should be appended to a file leaderboard.txt. The program will print the rank of the current game and the total number of scores in the leaderboard.txt. The leaderboard file should only contain a name and a score in each line.

```
Game grid:

D D S T W S D T W S
S W T D S W T S D T
D D S T W S D T W S
S W T D S W T S D T
W D S T D D S W T S
D D S T W S D T W S
S W T D S W T S D T
W D S T D D S W T S
D D S T W S D T W S
D D S T W S D T W S

Select coordinate or enter E to end the game: 7 3

D D S   W S D T W S
S W T   S W T S D T
D D S   W S D T W S
S W T T S W T S T T
W D S D D D S W T S
D D S T W S D T W S
S W T D S W T S D T
W D S T D D S W T S
D D S T W S D T W S
D D S D W S D T W S
```

```
Score: 45 points.
Select coordinate or enter E to end the game: 5 8

D D S   W S D T   S
S W T   S W T S   T
D D S   W S D T   S
S W T T S W T S W T
W D S D D D S W D S
D D S T W S D T W S
S W T D S W T S D T
W D S T D D S W T S
D D S T W S D T W S
D D S D W S D T W S
Score: 40 points.
Select coordinate or enter E to end the game: E

Total score:  85 points.

Enter name: GE
Your rank is 5/15, your score is 20 points higher than
ELVIS and 10 points lower than KILROY

Good bye!
```

Extending the jewels with Mathematical Symbols:

Along with the jewels defined above, you should implement the following mathematical symbol jewels as well. All Mathematical symbols are worth 20 points.

"/" : Matches any other mathematical symbol jewel only in right diagonal. Search 3 first, then 7.

"-" : Matches any other mathematical symbol jewel by searching horizontally (4 and 6).

"+" : Matches any other mathematical symbol jewel by searching first horizontally (4 and 6) then vertically (2 and 8)

"\" : Matches any other mathematical symbol jewel only in left diagonal. Search 1 first, then 9.

"|" : Matches any other mathematical symbol jewel vertically. Search 2 first, then 8.

Your submission should support all jewels in the set {D, S, T, W, /, -, +, \, |}. Your program will be evaluated using a grid containing a random subset of these Jewels. Also, your Object-oriented design should allow the addition of new jewels without changing too much code.

Your program should read the gameGrid.txt file for the initial grid and jewels. You should use abstract class and/or interfaces to implement common characteristics of the jewels. Your design should maximize reuse and minimize duplicate code. Make use of polymorphism in such a way that your game's code does not have to change if we decide to add a new Jewel type.

Make sure that your program runs both by entering values from the keyboard and by using a command file that can be redirected to your process. Command file will only contain coordinates entered at each step (two numbers separated by a space and ending with line break). Finally, to end the program it will contain an "E" character and a name for the leaderboard. Your program will be evaluated by redirecting a set of test case files in Linux (bash) as follows. Make sure that your program will work with such command files.

```
java Main < test_case.txt
```

Submit Format

File hierarchy must be zipped before submitted (Not .rar, only .zip files are supported by the system)

<student id>.zip

        <src>

                - Main.java, *.java

        <report>

                - report.pdf

Notes and Restrictions

- Regardless of the length, use UNDERSTANDABLE names to your variables, classes and functions. The names of classes, attributes and methods should obey Java naming convention. Your code should be formatted with correct indentations.
- Execution of application will be **60 points**.
- Write READABLE SOURCE CODE block and comments (Readability and modularity) (**10 points**)
- Save all your work until the assignment grades are announced.
- Report will be **30 points.**
- To find the rank in leaderboard, you <u>should</u> implement Comparable [2] interface. Use the sort and binarySearch methods in Collections [3] or Arrays [4] classes.
- You should give your detailed design in solution (with UML class diagram) (**10 points**) in the report.pdf. Your report should contain your solution for the different types of Jewels and explain in a separate subsection the required steps in your program to add a new Jewel type.
- Please explain your design, data structures and algorithms (**20 points**) in the report.pdf. Give necessary details without unnecessary clutter.
- Guide: ftp://ftp.cs.hacettepe.edu.tr/pub/dersler/genel/FormatForLabReports.doc
- All assignments must be original, individual work. Duplicate or very similar assignments are both going to be considered as cheating.
- Submit your work to " http://submit.cs.hacettepe.edu.tr ". Another type of submissions WILL NOT BE ACCEPTED.
- Don't use packages.

References

[1] PopCap Games, http://www.popcap.com.

[2] https://docs.oracle.com/javase/7/docs/api/java/lang/Comparable.html

[3] https://docs.oracle.com/javase/7/docs/api/java/util/Collections.html

[4] https://docs.oracle.com/javase/7/docs/api/java/util/Arrays.html