[TopCoder Software]

Component Specification

1. Design

The MathMatrix component provides an ability to set up and evaluate complex matrices. The component consists of three classes:

1) MathMatrix: Designed to hold all LineItems and Sections that are part of matrix.
2) Section: Acts as a group of LineItems and Section objects with assigned weight.
3) LineItem: Basic building block to create a MathMatrix or Section. It is described with weight, maximum allowable score, actual score and description.

All of these classes are subclasses of abstract Item class that defines and partially implements the functionality common to items that are stored in MathMatrix.

Also MathMatrix and Section classes are subclasses of Container class that defines and implements the functionality common to items that can have nested items. The Container class implements all abstract methods inherited from Item class.

Section and MathMatrix look very similar but they have a difference in getting the maximum scores. Section returns a sum of maximum scores of nested items while MathMatrix returns a maximum score assigned to it. Also MathMatrix overrides the `Item.setWeight() and Item.getMaximumScore()` methods since the current design assumes that MathMatrix can not be nested within any other MathMatrix and should always have a weight 1.00.

The current design provides the ability to:

- Group LineItems into Sections
- Assign a weight to each group and LineItem
- Calculate the weighted score for each Section, LineItem and the matrix as a whole.

1.1 Reference any design patterns used

None

1.2 Reference any standards used in the design

None

1.3 Explain any required algorithms for the implementation (provide pseudo code)

Double rounding
When checking to determine if weights sum to 1.00 keep in mind the MathMatrix.EPSILON, that presents a deviation from 1.00, to prevent double roundings problem (for example, when sum of weights is 0.999999999)

Calculating scores
When calculating the actual and weighted scores for Containers (i.e. MathMatrix and Sections), there should be a loop over all items directly nested in the Container summing the scores of nested items. This also applies to calculating the maximum score for Section, while `MathMatrix.getMaximumScore()` simply returns a maximum score assigned to MathMatrix.

```
double res = 0.00;

for each Item contained within Container {
      res = res + Item.getWeightedScore()
}
```

A weighted score for LineItem is calculated in accordance with following formula :

```
weighted score = actualScore / maximumScore * weight *
                 owner.weight* total score
```

where `actualScore, maximumScore, weight` – are properties of LineItem; `owner.weight` is a weight assigned to owner of LineItem(i.e. Section or MathMatrix) and `total score` is a maximum score assigned to MathMatrix

To get the total score assigned to MathMatrix, the Item should call `owner.getTotalScore()` method. If Item does not have an owner, it should return `getMaximumScore()`. Otherwise it should return `owner.getTotalScore()`.

1.4 Component Exception Definitions

IllegalStateException is thrown from Container when any of the methods getWeightedScore(), getActualScore(), getMaximumScore() is invoked but sum of weights of nested elements is not equal to 1.0 (100%). Since the MathMatrix overrides `getMaximumScore()` method it doesn't throw IllegalStateException.

IllegalArgumentException:

- When setWeight() method is invoked with parameter that is not within the range (0 , 1]
- Method addItem() is invoked and the sum of weights of nested elements becomes greater than 1.0 or added item is an instance of MathMatrix
- Any of set methods is called with an argument less than zero
- Method LineItem.setMaximumScore() is invoked with an argument that is less than LineItem.actualScore or is not greater than 0
- Method LineItem.setActualScore() is invoked with an argument that is not within the range [0, LineItem.maximumScore]
- Method MathMatrix.setMaximumScore() is invoked with an argument that is not greater than 0

NullPointerException

Thrown by Container methods when there is an attempt to get an index of `null` object or add a `null` to Container.

IndexOutOfBoundsException

Thrown by Container methods when there is an attempt to remove or get the Item with index that is out of Container bounds.

2. Environment Requirements

2.1 TopCoder Software Components:

None

2.2 Third Party Components:

None

3. Installation and Configuration

3.1 Package Name

com.topcoder.math.matrix

3.2 Configuration Parameters

None

3.3 Dependencies Configuration

None

4. Usage Notes

4.1 Required steps to use the component

In order to use component user application should do following steps :

```
// Create MathMatrix with given total score and description
MathMatrix matrix = new MathMatrix("Initial Screening", 100);

// Create and configure any required LineItems
LineItem item1 = new LineItem("LineItem 1", 0.1, 4);
LineItem item2 = new LineItem("LineItem 2", 0.4, 4, 3);
item1.setActualScore(2);

// If needed create any Sections to hold created LineItems
Section section = new Section("Section 1", 0.25);
section.addItem(item1);
section.addItem(item2);

// add Section to MathMatrix
matrix.addItem(section);

// or directly put items into MathMatrix
LineItem item3 = new LineItem("LineItem 3", 0.35, 4, 2);
LineItem item4 = new LineItem("LineItem 4", 0.4, 4, 3);

matrix.addItem(item3);
matrix.addItem(item4);
```

```
// Get all needed info
double itemresult = item1.getWeightedScore();
double sectionresult = section.getWeightedScore();
double matrixresult = matrix.getWeightedScore();
```

4.2 Demo

None