Testat 4 – 9.00 Uhr

- *Achte darauf, dass die vorgegebene Signatur in Deiner Lösung exakt eingehalten wird.*
- *Gib nur die bearbeitete Datei mit dem Namen* `Testat.java` *im Moodle-Bereich ab.*
- *Die Bearbeitungszeit endet genau um 10.00 Uhr.*

Vervollständige die Klasse `Testat` mit der Methode `apply`, die folgende Signatur besitzt:

```
public static Fraction apply( Fraction[] arr, int i )
```

Die Methode `apply` soll die folgende Funktionalität bereitstellen:

- Die Methode `apply` muss *rekursiv* arbeiten. Bei der Implementierung von `apply` dürfen die Schlüsselwörter `for` und `while` *nicht* verwendet werden.

- Für alle `i` mit `0<=i<arr.length` soll die Methode `apply` im Bereich von einschließlich `arr[0]` bis einschließlich `arr[i]` die Summe derjenigen Brüche bestimmen, deren Werte zwischen `0` und einschließlich `1` liegen.
  Die ermittelte Summe soll als Bruch zurückgegeben werden.

- Für alle Werte von `i`, die außerhalb des Bereichs `0<=i<arr.length` liegen, soll die Methode `apply` einen Bruch mit demWert `-1` zurückgeben.

- Es dürfen *keine* Attribute und *keine* weiteren Methoden in der Klasse `Testat` angelegt werden.

- Die Methode `apply` muss in einem Programm mehrfach nacheinander aufgerufen werden können und bei jedem Aufruf das entsprechend der Aufgabenbeschreibung korrekte Ergebnis zurückgeben.

Bei einer korrekten Implementierung liefert die Methode `apply` folgende Ausgaben:

Argumente		*erwartete Rückgabe*
`arr: {0/1,1/5,11/6,1/5,1/10,1/1}`	`i: 5`	`3/2`
`arr: {1/5,11/6,1/1,1/10,2/1}`	`i: 4`	`13/10`
`arr: {2/1,6/7,8/7}`	`i: 2`	`6/7`
`arr: {2/7}`	`i: 0`	`2/7`
`arr: {7/2}`	`i: 0`	`0/1`
`arr: {1/5,11/6,1/5,1/10,1/1}`	`i: -1`	`-1/1`
`arr: {1/5,11/6,1/5,1/10,1/1}`	`i: 5`	`-1/1`
`arr: {}`	`i: 1`	`-1/1`