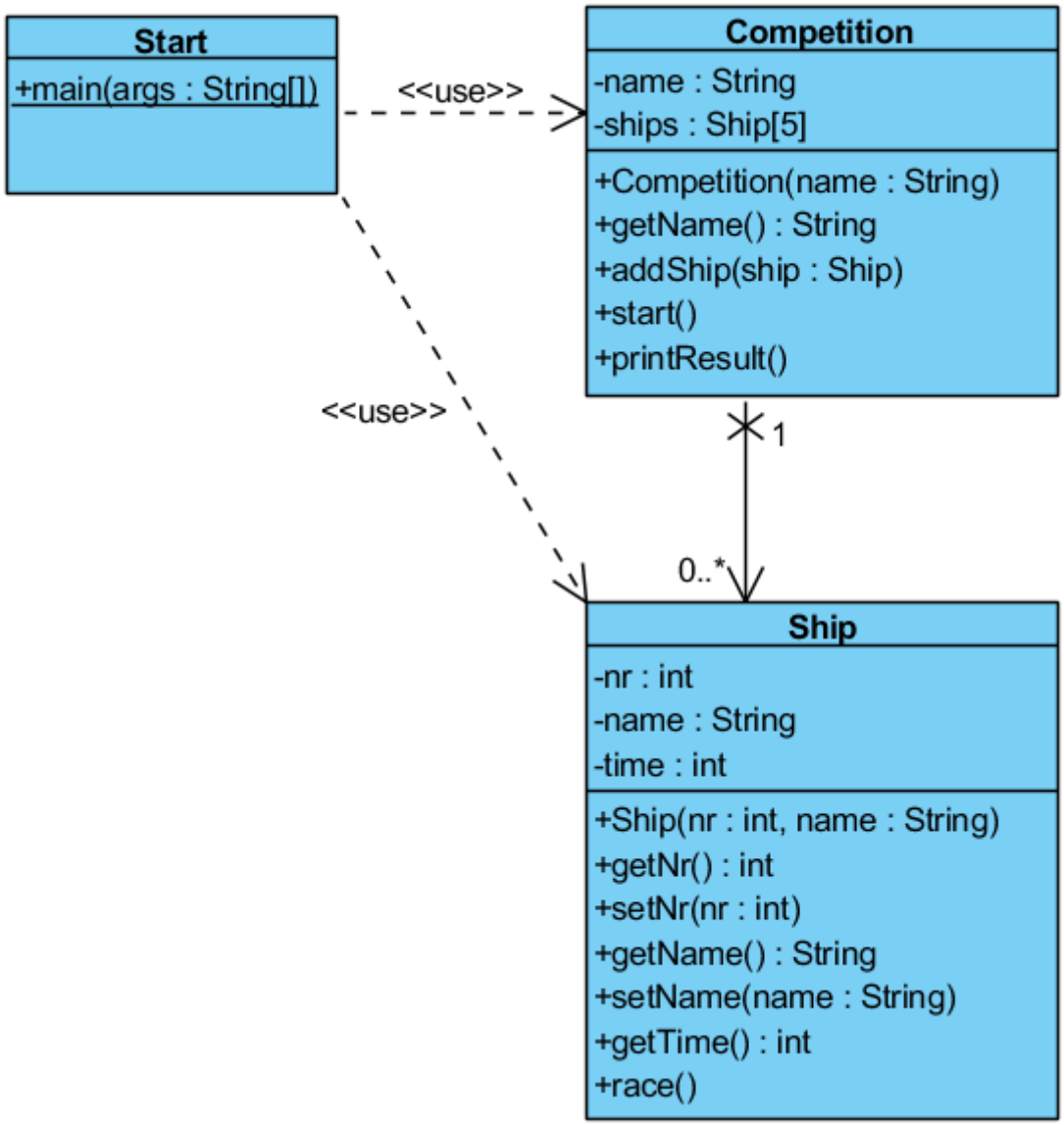
Auftrag Regatta

Bisher haben Sie meistens mit einer einzigen Fachklasse (z.B. Account) gearbeitet. Wir erweitern dies in diesem Beispiel um eine weitere Klasse.

In dieser Aufgabe sollen Segelwettkämpfe simuliert werden können. An einem Wettkampf können vorerst maximal 5 Schiffe teilnehmen. Jedes Schiff erreicht eine zufällige Rennzeit zwischen 300 und 600 Sekunden. Schiffe besitzen ausserdem eine Nummer und einen Namen. An einem Wettkampf sollen Schiffe angemeldet werden können. Wird ein Wettkampf gestartet, wird für jedes Schiff, welches zum Wettkampf hinzugefügt wurde, die zufällige Rennzeit berechnet. Schiffe sollen diese Zeit in einer Instanzvariable speichern. Nach einem Wettkampf sollen alle Schiffe mit jeweils Nr, Name und Rennzeit auf die Konsole ausgegeben werden können.

Eine mögliche Implementation führt zu folgendem Klassendiagramm:



Um einen Einblick zu erhalten, wie die Objekte in Start verwendet werden können:

```
Competition c = new Competition("Rotsee Regatta");

Ship ship1 = new Ship(1, "Alinghi");
Ship ship2 = new Ship(2, "Red Baron");
Ship ship3 = new Ship(3, "Blue Lagoon");

c.addShip(ship1); //add ships to competition
c.addShip(ship2);
c.addShip(ship3);

c.start(); //start competition

c.printResult(); //print ships with time
```

Hier ein Beispiel Output des obigen Programms:

```
Wettkampf: Rotsee Regatta
Schiff Nr: 1 Name: Alinghi Zeit: 383
Schiff Nr: 2 Name: Red Baron Zeit: 349
Schiff Nr: 3 Name: Blue Lagoon Zeit: 489
```

Folgende Punkte sollen Ihnen bei der Implementierung helfen, lesen Sie diese zuerst komplett durch und gehen Sie danach vor:

1. Erzeugen Sie ein neues Eclipse-Projekt Regatta
2. Erstellen Sie die Klasse `Start` (inkl. Main-Methode, worin Sie dann einen Testwettkampf starten können), die Klasse `Competition` und die Klasse `Ship`
3. Schiffe besitzen eine Nummer und einen Namen, welche initial über den Konstruktor, später über die Methoden `setNr`(…) und `setName`(…) gesetzt werden können.
4. Ein Schiff soll eine zufällige Rennzeit zurücklegen. Implementieren Sie dies in der Methode `race`(). Diese berechnet zufällig einen Wert zwischen 300 und 600 Sekunden (Random). Diesen Wert speichern Sie in der Instanzvariable `time`. Für `time` implementieren Sie die getter-Methode `getTime`().
5. Wettkämpfe besitzen einen Namen. Dieser soll initial über den Konstruktor gesetzt werden können und später mit der getter-Methode `getName`() abgefragt werden können.
6. Wettkämpfe besitzen maximal 5 Schiffe. Sie haben bisher Arrays mit primitiven Datentypen kennengelernt. Sie können Arrays genau gleich mit Referenzdatentypen definieren
   Sie schreiben also z.B. anstelle:
   ```
   private int[] zahlen = new int[5];
   ```
   für Arrays mit Referenzdatentypen:
   ```
   private Ship[] ships = new Ship[5];
   ```

So können Sie anstelle von Zahlen, Schiff Objekte in einem Array ablegen. Der Zugriff auf einzelne Elemente des Arrays erfolgt exakt gleich wie bei Arrays mit primitiven Datentypen. Konsultieren Sie dazu Ihre Unterlagen aus dem Vormodul.
Hinweis: Solange kein Schiffobjekt in obiges Array eingesetzt wurde, steht an einer Stelle im Array der Wert „`null`“. -> also keine Referenz auf ein Objekt.

7. Der Methode `addShip`(…) in Wettkampf soll ein Schiff-Objekt als Parameter übergeben werden können. Dieses Schiff soll in den Array `ships` an einen noch freien Platz eingefügt werden (siehe Hinweis bei Punkt 6 für Detektion von freien Plätzen)
8. Die Methode `start`() soll einen Wettkampf starten. Dabei wird für jedes Schiff, welches sich im Array befindet, die Methode `race`() aufgerufen (also auf jedem Schiff-Objekt), damit die Rennzeiten berechnet werden.
9. Die Methode `printResult()` gibt den Namen des Wettkampfs sowie Nr, Name und Rennzeit aller Schiffe auf die Konsole aus. Die Schiffe müssen dabei noch nicht nach Rennzeit sortiert werden.
10. Schreiben Sie nun in der Klasse `Start` in der `main` Methode ein Testszenario, analog obigem Codeausschnitt.