Chapter 9

Exercise 9.2: Stock

<table>
<thead>
<tr>
<th>Stock</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>symbol: String</td>
<td>The symbol of this stock.</td>
</tr>
<tr>
<td>name: String</td>
<td>The name of this stock.</td>
</tr>
<tr>
<td>previousClosingPrice: double</td>
<td>The previous closing price of this stock.</td>
</tr>
<tr>
<td>currentPrice: double</td>
<td>The current price of this stock.</td>
</tr>
<tr>
<td>Stock(symbol: String, name: String)</td>
<td>Constructs a stock with a specified symbol and a name.</td>
</tr>
<tr>
<td>getChangePercent(): double</td>
<td>Returns the percentage of change of this stock.</td>
</tr>
</tbody>
</table>

Exercise 9.6: StopWatch

<table>
<thead>
<tr>
<th>StopWatch</th>
<th>get methods for all data fields are provided and omitted for brevity.</th>
</tr>
</thead>
<tbody>
<tr>
<td>-startTime: long</td>
<td>Constructs a StopWatch object.</td>
</tr>
<tr>
<td>-endTime : long</td>
<td>Starts the watch.</td>
</tr>
<tr>
<td>+StopWatch()</td>
<td>Stops the watch.</td>
</tr>
<tr>
<td>+start(): void</td>
<td>Returns the elapsed time for the stop watch in milliseconds.</td>
</tr>
<tr>
<td>+stop(): void</td>
<td></td>
</tr>
<tr>
<td>+getElapsedTime(): long</td>
<td></td>
</tr>
</tbody>
</table>
Exercise 9.8: Fan

<table>
<thead>
<tr>
<th>Fan</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>:SLOW = 1</td>
<td>Constant.</td>
</tr>
<tr>
<td>:MEDIUM = 2</td>
<td>Constant.</td>
</tr>
<tr>
<td>:FAST = 3</td>
<td>Constant.</td>
</tr>
<tr>
<td>:speed: int</td>
<td>The speed of this fan (default 1).</td>
</tr>
<tr>
<td>:on: boolean</td>
<td>Indicates whether the fan is on (default false).</td>
</tr>
<tr>
<td>:radius: double</td>
<td>The radius of this fan (default 5).</td>
</tr>
<tr>
<td>:color: String</td>
<td>The color of this fan (default white).</td>
</tr>
</tbody>
</table>

| Fan() | Constructs a fan with default values. |
| :getSpeed(): int | Returns the speed of this fan. |
| :setSpeed(speed: int): void | Sets a new speed for this fan. |
| :isOn(): boolean | Returns true if this fan is on. |
| :setOn(on: boolean): void | Sets this fan on to true or false. |
| :getRadius(): double | Returns the radius of this fan. |
| :setRadius(radius: double): void | Sets a new radius for this fan. |
| :getColor(): String | Returns the color of this fan. |
| :setColor(color: String): void | Sets a new color for this fan. |
| :toString(): String | Returns a string representation for this fan. |

Exercise 9.10: QuadraticEquation

<table>
<thead>
<tr>
<th>QuadraticEquation</th>
<th>get methods for all data fields are provided and omitted for brevity.</th>
</tr>
</thead>
<tbody>
<tr>
<td>:a : double</td>
<td>Three coefficients for the equation.</td>
</tr>
<tr>
<td>:b : double</td>
<td></td>
</tr>
<tr>
<td>:c : double</td>
<td></td>
</tr>
</tbody>
</table>

| QuadraticEquation(a : double, b: double, c: double) | Constructs a QuadraticEquation with the specified coefficients. |
| :getDiscriminant(): double | Returns the discriminant of this equation. |
| :getRoot1(): double | Returns the first root of this equation. |
| :getRoot2(): double | Returns the second root of this equation. |

Chapter 10

Exercise 10.4: MyPoint
### MyPoint

- `x: double`  
  x-coordinate of this point.
- `y: double`  
  y-coordinate of this point.

- `+MyPoint()`  
  Constructs a Point object at (0, 0).
- `+MyPoint(x: double, y: double)`  
  Constructs an object with specified x and y values.
- `+getX(): double`  
  Returns x value in this object.
- `+getY(): double`  
  Returns y value in this object.
- `+distance(secondPoint: MyPoint): double`  
  Returns the distance from this point to another point.
- `+distance(p1: Point, p2: MyPoint): double`  
  Returns the distance between two points.

### Exercise 10.8: Tax

```java
Tax
```

- `filingStatus: int`  
  -filingStatus: int
- `brackets: int[][]`  
  -brackets: int[]
- `rates: double[]`  
  -rates: double
- `taxableIncome: double`  
  -taxableIncome: double

- `+Tax()`  
  +Tax(filingStatus: int, brackets: int[], rates: double[], taxableIncome: double)
- `+getFilingStatus(): int`  
  +getFilingStatus(filingStatus: int): void
- `+setFilingStatus(brackets: int[]): void`  
  +setBrackets(brackets: int[]): void
- `+getRates(): double[]`  
  +getRates(rates: double[]): void
- `+setRates(rates: double[]): void`  
  +setRates(rates: double[]): void
- `+getTax(): double`  
  +getFilingStatus(): int
- `+getTaxableIncome(): double`  
  +getTaxableIncome(): double

### Exercise 10.12: MyRectangle2D
Exercise 10.14: MyDate

- **MyDate**
  - year: int
  - month: int
  - day: int

  **Methods**
  - MyDate()
  - MyDate(elapsedTime: long)
  - getYear(): int
  - getMonth(): int
  - getDay(): int
  - setDate(elapsedTime: long): void

  The year for the date.
  The month for the date.
  The day for the date.

  Constructs MyDate for the current date.
  Constructs MyDate with a specified elapsed time in milliseconds.
  Returns the year for the date.
  Returns the month for the date.
  Returns the day for the date.
  Sets a new date using the elapsed time.
**Chapter 11**

**Exercise 11.2: Person, Student, Staff, Employee**

![Class diagram for Person, Student, Staff, Employee](image_url)

### Person
- **name**: String
- **address**: String
- **phone**: String
- **email**: String

+ **new Person()**
+ **new Person(name: String, address: String, phone: String, email: String)**
+ **getName()**: String
+ **getAddress()**: String
+ **getPhone()**: String
+ **getEmail()**: String
+ **setName(name: String)**: void
+ **setAddress(address: String)**: void
+ **setPhone(phone: String)**: void
+ **setEmail(email: String)**: void
+ **toString()**: String

### Student
- **status**: String

+ **new Student()**
+ **new Student(name: String, address: String, phone: String, email: String)**
+ **getStatus()**: String
+ **setStatus(status: String)**: void
+ **toString()**: String

### Employee
- **office**: String
- **salary**: int
- **dateHired**: java.util.Date

+ **new Employee()**
+ **new Employee(name: String, address: String, phone: String, email: String)**
+ **getOffice()**: String
+ **getSalary()**: int
+ **getDateHired()**: Date
+ **setOffice(office: String)**: void
+ **setSalary(salary: int)**: void
+ **setDateHired(dateHired: Date)**: void
+ **toString()**: String

### Faculty
- **officeHour**: String
- **rank**: String

+ **new Faculty()**
+ **new Faculty(name: String, address: String, phone: String, email: String)**
+ **getOfficeHour()**: String
+ **setOfficeHour(officeHour: String)**: void
+ **getRank()**: String
+ **setRank(rank: String)**: void
+ **toString()**: String
### Staff

- title: String

- Staff()

+Staff(name: String, address: String, phone: String, email: String)

+getTitle(): String

+setTitle(title: String): void

+toString(): String

### MyDate

- year: int

- month: int

- day: int

- Faculty()

+getYear(): int

+getMonth(): int

+getDay(): int

+setYear(year: int): void

+setMonth(month: int): void

+setDay(day: int): void

+toString(): String
Exercise 11.10: MyStack

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>isEmpty()</td>
<td>Returns true if this stack is empty.</td>
</tr>
<tr>
<td>getSize()</td>
<td>Returns the number of elements in this stack.</td>
</tr>
<tr>
<td>peek()</td>
<td>Returns the top element in this stack.</td>
</tr>
<tr>
<td>pop()</td>
<td>Returns and removes the top element in this stack.</td>
</tr>
<tr>
<td>push(o: Object)</td>
<td>Adds a new element to the top of this stack.</td>
</tr>
<tr>
<td>search(o: Object): int</td>
<td>Returns the position of the first element in the stack from the top that matches the specified element.</td>
</tr>
</tbody>
</table>
Chapter 13

Exercise 13.6: ComparableCircle

```
«interface»
java.lang.Comparable<ComparableCircle>

ComparableCircle

+ComparableCircle(radius: double)
+compareTo(object: ComparableCircle): int
```
**Exercise 13.10: Rectangle**

```
«interface»
Comparable<Rectangle>
+compareTo(object: Rectangle): int
```

`GeometricObject`

`Rectangle`