**4.1 Vorbereitung**

**4.1.1**

|  |  |
| --- | --- |
| **Klasse** | **Methoden in eigenen Klassen** |
| Array | 1. == (Folie V6/b S.16) |
| Hash / Set | 1. eql? und hash (Folie V6/b S. 40/41) 2. eql? und hash (Folie V6/b S. 40/41) |

**4.1.2**

**1. Implementierung der Wertgleichheit**

Die Methode == von **Object** der eigenen Klasse überschreiben, so dass sie für die eigene Klasse Gleichheit und nicht Identität prüft.

**def ==(other)**

**return false if other.nil?**

**return true if self.equal?(other)**

**return [@param1,@param2] == [other.param1, other.param2]**

**end**

**2. Implementierung der Wert- und Typgleichheit**

Set nutzt **eql?** Gleichheit und **hash**, um Elemente im Set zu suchen. Bei Einfügen der eigenen Objekte in Set werden die Elemente auf Identität geprüft.

**def eql?(other)**

**return false if other.nil?**

**return true if self.equal?(other)**

**return false if self.class != other.class**

**return [@param1,@param2].eql?([other.param1, other.param2])**

**end**

**3. Implementierung der hash-Methode**

Nach dem Überschreiben der **hash**-Methode können eigene Objekte als Schlüssel in Hashes korrekt verwendet werden.

**def eql?(other)**

**. . .**

**end**

**def hash**

**return @param1.hash**

**end**

**4.2 Studium des mitgelieferten Projektes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Klasse: datum.rb** | | | |
| **Methodenname** | **Parameter** | **Funktion** | **Arbeitsweise** |
| def initialize | tag,monat,jahr,stunde,minute | Erzeugt die Instanzvariable (@time) des Datum-Objektes | Erzeugt die Instanzvariable @time der Time-Klasse mit den formalen Parametern als Argumente |
| def <=>(other) | other | Ermöglicht das Vergleichen von Datumsobjekten mit <, <= etc. | Vergleicht self mit einem formalen Parameter der Klasse Time |
| attr\_reader | @time | Liest die Instanzvariable @time | Liefert den Wert der Instanzvariable zurück |
| def +(dauer) | dauer | Addiert die Dauer-parameter auf das Datum-Objekt. | Dauer wird in Sekunden gewandelt und dann zu addiert @time. Liefert als Ergebnis das neue Datum zurück. |
| def monat\_im\_jahr |  | Gibt den Monat des Datums zurück | Liefert ein neues Objekt der Klasse „Monat“ mit den Argumenten von dem Datum-Objekt. |
| def tag\_im\_monat |  | Gibt den Tag des Datums zurück | Liefert als Ergebnis der Methode ein neues Objekt der Klasse „Tag“ mit dem Argumenten von dem Datum-Objekt. |
| def to\_s |  | Bereitet die Darstellung des Objekts in einer Zeichenkette auf | Formatiert das Datum mit "%d.%m.%Y %H:%M"  ins gewünschte Format |

|  |  |  |  |
| --- | --- | --- | --- |
| **Klasse: dauer.rb** | | | |
| **Methodenname** | **Parameter** | **Funktion** | **Arbeitsweise** |
| def initialize | minuten | Erzeugt die Instanzvariable (@minuten) des Dauer-Objektes | Erzeugt die Instanzvariable @minuten und weist den formalen Parameter als Argument zu |
| def stunden\_anteil |  | Ermittelt den Anteil der Stunden aus der Anzahl von Minuten | Berechnet und liefert Anteil von Stunden aus Anzahl von Minuten |
| def minuten\_anteil |  | Ermittelt den Anteil der Minuten aus der Anzahl von Minuten | Berechnet und liefert Anteil von Minuten aus Anzahl von Minuten Modulo 60 |
| def in\_sekunden |  | Ermittelt den Anteil der Sekunden aus der Anzahl von Minuten | Berechnet und liefert Anteil von Sekunden aus Anzahl von Minuten multipliziert mit 60 |
| def +(other) | other | Addiert ein Objekt von Typ Dauer auf das Dauer-Objekt self | Erzeugt ein neues Datum-Objekt mit den bestehenden Minuten und addiert die Minuten des formalen Parameters hinzu |
| def <=>(other) | other | Ermöglicht das Vergleichen von Datumsobjekten mit <, <= etc. | Vergleicht die Instanzvariablen @minuten von zwei Dauer-Objekten. Liefer -1,0,1 zurück |
| def to\_s |  | Bereitet die Darstellung des Objekts in einer Zeichenkette auf | Formatiert das Dauer-Objekt in eine Uhrzeitdarstellung mit Stunden und Minuten und fügt mit .rjust eine 0 hinzu |

|  |  |  |  |
| --- | --- | --- | --- |
| **Klasse: monat.rb** | | | |
| **Methodenname** | **Parameter** | **Funktion** | **Arbeitsweise** |
| def initialize | monat, jahr | Erzeugt die Instanzvariable (@time) des Monat-Objektes | Erzeugt die Instanzvariable @time der Time-Klasse mit den formalen Parametern als Argumente |
| def <=>(other) | other | Ermöglicht das Vergleichen von Datumsobjekten mit <, <= etc. | Vergleicht die Instanzvariablen @time von zwei Monat-Objekten |
| attr\_reader | @time | Liest die Instanzvariable @time | Liefert den Wert der Instanzvariable zurück |
| def to\_s |  | Bereitet die Darstellung des Objekts in einer Zeichenkette auf | Formatiert das Monat-Objekt in eine Zeichenkette mit der Formatierung "%m.%Y" |

|  |  |  |  |
| --- | --- | --- | --- |
| **Klasse: tag.rb** | | | |
| **Methodenname** | **Parameter** | **Funktion** | **Arbeitsweise** |
| def initialize | jahr, monat, tag | Erzeugt die Instanzvariable (@time) des Tag-Objektes | Erzeugt die Instanzvariable @time der Time-Klasse mit den formalen Parametern als Argumente |
| def <=>(other) | other | Ermöglicht das Vergleichen von Datumsobjekten mit <, <= etc. | Vergleicht die Instanzvariablen @time von zwei Tag-Objekten |
| attr\_reader | @time | Liest die Instanzvariable @time | Liefert den Wert der Instanzvariable zurück |
| def to\_s |  | Bereitet die Darstellung des Objekts in einer Zeichenkette auf | Formatiert das Monat-Objekt in eine Zeichenkette mit der Formatierung "%d.%m.%Y" |

|  |  |  |  |
| --- | --- | --- | --- |
| **Klasse: termin.rb** | | | |
| **Methodenname** | **Parameter** | **Funktion** | **Arbeitsweise** |
| def initialize | beschreibung, datum, dauer | Erzeugt die Instanzvariablen (@beschreibung, @datum, @dauer) des Termin-Objektes | Erzeugt die Instanzvariablen @beschreibung, @datum, @dauer und weist diesen die formalen Parameter als Argumente zu |
| def to\_s |  | Bereitet die Darstellung des Objekts in einer Zeichenkette auf | Formatiert das Termin-Objekt in eine Zeichenkette |

|  |  |  |  |
| --- | --- | --- | --- |
| **Klasse: termin\_kalender.rb** | | | |
| **Methodenname** | **Parameter** | **Funktion** | **Arbeitsweise** |
| def initialize | besitzer | Erzeugt die Instanzvariablen (@besitzer und @termine) des TerminKalender-Objektes | Erzeugt die Instanzvariablen @besitzer und @termine. Der formale Parameter wird @besitzer als Argument zugewiesen und @termine wird als leeres Array initialisiert |
| def delete | termin | Löscht einen Termin aus dem Terminkalender | Nutzt .delete der Klasse Array um ein Element aus dem @termine-Array zu löschen |
| def passt\_nach? | termin, termin\_vor | Prüft, ob ein Termin zeitlich nach einem anderen Termin eingetragen werden kann | Liefert einen boolschen Ausdruck zurück. Vergleicht das Ende von dem vorherigen Termin mit dem Anfang des nächsten Termins. |
| def passt\_vor? | termin, termin\_nach | Prüft, ob ein Termin zeitlich vor einem anderen Termin eingetragen werden kann | Liefert einen boolschen Ausdruck zurück. Vergleicht den Anfang von dem nächsten Termin mit dem Ende des vorherigen Termins. |
| def << neuer\_termin | neuer\_termin | Einen neuen Termin in den Terminkalender eintragen | Erzeugt je eine lokale Variable für den vorherigen und den nächsten Termin des neuen Termins, der eingetragen werden soll.  Falls keine Termine vorhanden sind, wird der Termin eingetragen.  Prüft, ob nach dem neuen Termin ein anderer Termin liegt. Falls nicht, wird geprüft, ob der vorherige Termin und der neue sich überschneiden. Wenn nicht, neuen Termin eintragen. Falls doch, erfolgt eine Ausgabe auf der Konsole, mit einer Meldung darüber.  Diese Prüfung wird für den letzten Termin wiederholt, sowie für einen Termin zwischen dem vorherigen und dem nächsten Termin |