

Einführung in die Objekt-Orientierte Modellierung und Programmierung

Praktikum 2

Abgabetermin: 21.11.2025, 23:00 Uhr
Abgabeformat: pdf für die UML, zip für die Programme
Maximale Anzahl an Punkten: 86

Arrays, Kontrollstrukturen

1. Modellierung und Implementierung der Klasse `UtilsForArrays` (30 Punkte)

Anforderungen Klasse `UtilsForArrays`:

- Attribute: keine
- Methoden
 - `createIntArray`
 - * Parameter:
 - eine ganze Zahl `minimum`
 - eine ganze Zahl `maximum`
 - * Rückgabe:
 - ein Array mit ganzen Zahlen
 - * Funktionalität
 - Erzeugen eines Arrays mit ganzen Zahlen
Das Array sollte exakt so viel Platz verwenden, wie für die Speicherung der Werte notwendig ist.
 - Füllen des Arrays mit den Werten von `minimum` bis `maximum` einschließlich der Grenzen
 - Rückgabe des erzeugten Arrays
 - `quadriere`
 - * Parameter:
 - ein Array mit ganzen Zahlen
 - * Rückgabe:
 - ein neues Array mit ganzen Zahlen
 - * Funktionalität
 - Erzeugung eines neuen Arrays
 - Quadrieren aller Elemente des alten Arrays und speichern der Quadrate im neuen Array
 - Beispiel $(1, 2, 3, 4) \rightarrow (1, 4, 9, 16)$
 - Rückgabe des neuen Arrays
 - `arrayInhaltAusgeben`
 - * Parameter:

-
- ein Array mit ganzen Zahlen
 - * Rückgabe:
 - keine
 - * Funktionalität
 - Ausgabe aller Elemente des Arrays in der Reihenfolge ihrer Indizes in einer Zeile durch Komma getrennt

- a) Modellieren Sie die Klasse `UtilsForArrays` entsprechend den Anforderungen.
- b) Implementieren Sie die Klasse `UtilsForArrays` entsprechend den Anforderungen und dem Modell.

Allgemeine Anforderungen:

- Verwenden Sie ausschließlich die in den Aufgabenstellungen angegebenen Klassen und deren Methoden aus Bibliotheken
- Achten Sie auf angemessene Sichtbarkeiten.
- Achten Sie auf angemessene Benennungen der Variablen, Parameter und Attribute (Verwenden Sie einen anderen Namen als das oben verwendete `array`!)

2. Modellierung und Implementierung der Klasse `PythagoräischeZahlentripel` (49 Punkte)

Anforderungen Klasse `PythagoräischeZahlentripel`:

- Attribute
 - Ein Array zum Speichern der zu prüfenden Zahlen
 - Ein Array zum Speichern der Quadrate der zu prüfenden Zahlen
 - Die Anzahl der gefundenen Pythagoräischen Zahlentripel
- Methoden
 - Konstruktor
 - * Parameter:
 - eine ganze Zahl `minimum`
 - eine ganze Zahl `maximum`
 - * Funktionalität
 - Erzeugen Sie ein Array den zu prüfenden Zahlen von `minimum` bis `maximum` einschließlich der Grenzen.
Verwenden Sie hierzu die Methoden der Klasse `UtilsForArrays`.
Speichern Sie dieses Array im Attribut der zu prüfenden Zahlen.
 - Erzeugen Sie ein Array mit Quadratzahlen von `minimum` bis `maximum` einschließlich der Grenzen.
Verwenden Sie hierzu die Methoden der Klasse `UtilsForArrays`.
Speichern Sie dieses Array im Attribut der Quadrate der zu prüfenden Zahlen.
 - Geben Sie den Inhalt des Attributs der Quadrate der zu prüfenden Zahlen aus.
Verwenden Sie hierzu die Methoden der Klasse `UtilsForArrays`.
 - Berechnen Sie mit der unten definierten Methode `berechne` die Pythagoräischen Zahlentripel.
 - `istQuadratzahl`
 - * Parameter:
 - eine ganze Zahl `zahl`

- * Rückgabe:
 - Wahrheitswert
- * Funktionalität
 - Testen Sie *zahl* darauf, ob sie das Quadrat einer natürlichen Zahl *n* ist:

$$zahl = n^2, n \in \mathbb{N}$$

- Falls ja, gebe **true** zurück
- Falls nein, gebe **false** zurück
- Hinweis: Verwenden Sie **Math.sqrt** und **Math.floor**

– **berechne**

- * Parameter:
 - keine
- * Rückgabe:
 - keine
- * Funktionalität
 - Berechnen Sie alle Differenzen zweier Elemente des Arrays von Quadratzahlen *array*:

$$differenz = array[j] - array[i], j > i$$

- Falls $\sqrt{differenz} \in \mathbb{N}$ und $differenz > array[i]$, geben Sie das Ergebnis gemäß folgendem Beispiel aus:
 Inhalt des Arrays an der Position *i*: 9
 Inhalt des Arrays an der Position *j*: 25
 $differenz = 16$
 Ausgabe:

$$3^2 + 4^2 = 5^2$$

- Falls $\sqrt{differenz} \in \mathbb{N}$ und $differenz > array[i]$, erhöhen Sie die Anzahl der Pythagoräischen Zahlentripel um 1
- Hinweis: Verwenden Sie die Methoden **istQuadratzahl** und **Math.abs**

– **gibAnzahlAus**

- * Parameter:
 - keine
- * Rückgabe:
 - keine
- * Funktionalität
 - Geben Sie die Anzahl der Pythagoräischen Zahlentripel aus

- a) Modellieren Sie die Klasse **PythagoraeischeZahlentripel** entsprechend den Anforderungen.
- b) Implementieren Sie die Klasse **PythagoraeischeZahlentripel** entsprechend den Anforderungen und dem Modell.

Allgemeine Anforderungen:

- Verwenden Sie ausschließlich die in den Aufgabenstellungen angegebenen Klassen und deren Methoden aus Bibliotheken

-
- Achten Sie auf angemessene Sichtbarkeiten.
 - Achten Sie auf angemessene Benennungen der Variablen, Parameter und Attribute (Verwenden Sie einen anderen Namen als das oben verwendete `array`!)

3. Implementierung der Klasse `PythagoraeischeZahltripelMain` (7 Punkte)

- a) Implementieren Sie eine Klasse `PythagoraeischeZahltripelMain` mit folgenden Eigenschaften
- Die Klasse enthält ausschließlich die `main`-Methode. Hilfsmethoden der `main`-Methode sind erlaubt.
 - Erzeugen Sie zu Testzwecken mindestens 2 Instanzen der Klasse `PythagoraeischeZahltripel` mit:
 - `minimum = 1, maximum = 1000`
 - `minimum = 1001, maximum = 2000`
 - Geben Sie zusätzlich jeweils die Anzahl der gefundenen Pythagoräischen Zahlentripel aus (Zahl plus Bedeutung!).

Allgemeine Anforderungen:

- Verwenden Sie ausschließlich die in den Aufgabenstellungen angegebenen Klassen und deren Methoden aus Bibliotheken
- Achten Sie auf angemessene Sichtbarkeiten.
- Achten Sie auf angemessene Benennungen der Variablen.