Bibliotheksfunktionen, Binärzahlen und Zeichen

Lösungen zu den Übungsaufgaben

Übung: WM (1/2)


```java
/**
 * Ein WmFinalRundenSpiel hat zwei Teilnehmer und zwei Vorgänger.
 *
 * @author Philipp Jenke
 *
 */
public class WmFinalRundenSpiel {

  /**
   * Erstes Vorgänger-Spiel
   */
  private WmFinalRundenSpiel vorgaenger1;

  /**
   * Zweites Vorgänger-Spiel.
   */
  private WmFinalRundenSpiel vorgaenger2;

  /**
   * Erstes teilnehmendes Team. Gewinner des erstes Vorgängerspiels.
   */
  private String team1;

  /**
   * Zweites teilnehmendes Team. Gewinner des zweites Vorgängerspiels.
   */
  private String team2;

  /**
   * Dieses Flag ist wahr, wenn Team1 gewonnen hat, ansonsten faslsch.
   */
  boolean team1Gewinner;

  /**
   *
   * @param team1
   *          Initialisierung für team1.
   * @param team2
   *          Initialisierung für team2;
   * @param vorgaenger1
   *          Initialisierung für das erste Vorgängerspiel.
   * @param vorgaenger2
   *          Initialisierung für das zweite Vorgängerspiel.
   */
  public WmFinalRundenSpiel(String team1, String team2,
      WmFinalRundenSpiel vorgaenger1, WmFinalRundenSpiel vorgaenger2,
      boolean team1Gewinner) {
    this.team1 = team1;
    this.team2 = team2;
    this.vorgaenger1 = vorgaenger1;
    this.vorgaenger2 = vorgaenger2;
    this.team1Gewinner = team1Gewinner;
  }

  /**
   * Getter.
   *
   * @return Erster Vorgänger.
   */
  public WmFinalRundenSpiel getVorgaenger1() {
    return vorgaenger1;
  }

  /**
   * Getter.
   *
   * @return Zweiter Vorgänger.
   */
  public WmFinalRundenSpiel getVorgaenger2() {
    return vorgaenger2;
  }

  /**
   * Getter.
   *
   * @return Erstes Team.
   */
  public String getTeam1() {
    return team1;
  }

  /**
   * Getter.
   *
   * @return Zweites Team.
   */
  public String getTeam2() {
    return team2;
  }

  /**
   * Liefert das Team zurück, dass das Spiel gewonnen hat.
   *
   * @return Name des Gewinnerteams.
   */
  public String getGewinner() {
    return team1Gewinner ? team1 : team2;
  }
}
```

Übung: WM (2/2)


```java
import static org.junit.Assert.*;

/**
 * Testklasse für TestWmFinalRundenSpiel.
 *
 * @author Philipp Jenke
 *
 */
public class TestWmFinalRundenSpiel {

  @Test
  public void testKonstruktion() {
    String teamDeutschland = "Deutschland";
    String teamArgentinien = "Argentinien";
    String teamBrasilien = "Brasilien";
    String teamHolland = "Holland";
    WmFinalRundenSpiel halbfinale1 =
        new WmFinalRundenSpiel(teamDeutschland, teamBrasilien, null, null, true);
    WmFinalRundenSpiel halbfinale2 =
        new WmFinalRundenSpiel(teamHolland, teamArgentinien, null, null, false);
    WmFinalRundenSpiel finale =
        new WmFinalRundenSpiel(teamDeutschland, teamArgentinien, halbfinale1,
            halbfinale2, true);

    assertEquals(halbfinale1, finale.getVorgaenger1());
    assertEquals(halbfinale2, finale.getVorgaenger2());
    assertEquals(halbfinale1.getGewinner(), finale.getTeam1());
  }
}
```

Übung: Max3Bib


```java
/**
 * Berechnet die größte dreier Zahlen mit mathematischen
Bibliotheksfunktionen.
 */
public class Max3Bib {

  /**
   * Programmeinstieg.
   */
  public static void main(String[] args) {
    // Eingabe
    Scanner scanner = new Scanner(System.in);
    System.out.println("Bitte drei Ganzzahlen eingeben:");
    int zahl1 = scanner.nextInt();
    int zahl2 = scanner.nextInt();
    int zahl3 = scanner.nextInt();
    scanner.close();

    // Compute max
    final int maxZahl = Math.max(zahl1, Math.max(zahl2, zahl3));

    // Print result to console
    System.out.println("Größte zahl aus (" + zahl1 + ", " + zahl2 +
", "
        + zahl3 + "): " + maxZahl);
  }
}
```

Übung: Logische Operationen

Welches Ergebnis liefern diese Verknüpfungen?

- a) `0` AND 1 → 0
- b) `0` AND 0 → 0
- c) `1` AND 1 → 1
- d) `0` OR 1 → 1
- e) `0` XOR 1 → 1
- f) `1` OR 1 → 1
- g) `1` XOR 1 → 0
- h) `0` NOT 1 → Fehler
- i) NOT 0 → 1

Übung: Zeichen


```java
/**
 * Gibt ein Zeichen aus dem Alphabet anhand der Nummerierung aus.
 */
public class ZeichenAnStelle {

  /**
   * Programmeinstieg.
   */
  public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
    System.out.println("Bitte Zeichen eingeben.");
    int index = scanner.nextInt();
    scanner.close();
    if (index < 1 || index > 26) {
      System.out.println("Ungültiger Index!");
      return;
    }
    char zeichen = (char) (index + 96);
    System.out.println("Zeichen an Stelle " + index + ": " +
zeichen);
  }
}
```

Übung: Lineare Gleichungen

hier: nur eine Idee, viele Alternativen möglich

- Name der Klasse: LineareGleichungen
- `void berechneNullstellen(double a, double b, double c)`
- Rückgabe des Ergebnisses
  - `int getAnzahlNullstellen()`
  - `int getNullstelle(int index)`
- Kommentare bei
  - Klasse
  - Methoden
  - Objektvariablen