

Übung Informatik I – Programmierung – Blatt 4

(Lösungsvorschlag)

Aufgabe 1)

a)

$\langle a.length = n \rangle$

$\langle 0 = 0 \wedge a.length = n \rangle$

$j = 0;$

$\langle j = 0 \wedge a.length = n \rangle$

$\langle 0 = 0 \wedge j = 0 \wedge a.length = n \rangle$

$sum = 0;$

$\langle sum = 0 \wedge j = 0 \wedge a.length = n \rangle$

$\langle sum = \sum_{i=0}^{j-1} a[i] \wedge a.length = n \wedge j \leq n \rangle$

while ($j < n$) {

$\langle sum = \sum_{i=0}^{j-1} a[i] \wedge a.length = n \wedge j < n \rangle$

$\langle sum + a[j] = \sum_{i=0}^j a[i] \wedge a.length = n \wedge j < n \rangle$

$sum = sum + a[j];$

$\langle sum = \sum_{i=0}^j a[i] \wedge a.length = n \wedge j < n \rangle$

$\langle sum = \sum_{i=0}^{(j+1)-1} a[i] \wedge a.length = n \wedge j < n \rangle$

$\langle sum = \sum_{i=0}^{(j+1)-1} a[i] \wedge a.length = n \wedge j+1 \leq n \rangle$

$j = j + 1;$

$\langle sum = \sum_{i=0}^{j-1} a[i] \wedge a.length = n \wedge j \leq n \rangle$

}

$$\langle \text{sum} = \sum_{i=0}^{j-1} a[i] \wedge a.length = n \wedge \neg j < n \wedge j \leq n \wedge \rangle$$

$$\langle \text{sum} = \sum_{i=0}^{j-1} a[i] \wedge a.length = n \wedge j = n \rangle$$

$$\langle \text{sum} = \sum_{i=0}^{a.length-1} a[i] \rangle$$

b)

$$\text{Invariante: } I : \text{sum} = \sum_{i=0}^{j-1} a[i] \wedge a.length = n \wedge j \leq n$$

c)

Wähle $n-j$ als Variante:

Da j pro Schleifendurchlauf um 1 erhöht (also $n-j$ dekrementiert) wird und $j \geq n$ die Abbruchbedingung ist, terminiert die while-Schleife stets.

Formal:

(1)

$$(j < n) \Rightarrow (n - j \geq 0) \quad \text{trivial}$$

(2)

$$\langle n - j = m \rangle$$

$$\langle n - (j + 1) = m - 1 \rangle$$

$$\text{sum} = \text{sum} + a[j]; j = j + 1;$$

$$\langle n - j = m - 1 \rangle$$

$$\langle n - j < m \rangle$$

Aufgabe 2)

a)

```
/* Das Programm berechnet das Ergebnis der Division eines Dezimalbruchs durch
 * eine positive ganze Zahl x. Das Programm liest dazu 10 Dezimalstellen durch den
 * Benutzer ein und speichert diese in einem Array. Danach wird eine positive ganze
 * Zahl x eingelesen, die den Dezimalbruch teilt.
 * Autor: Thomas von der Maßen
 * Umgebung: JDK 1.3.1, Windows 2000
 * Erstellt: 14.11.2001
 */
```

```
public class Bruchdivision {

    public static void main (String[] args) {
        final int stellen = 10;
        int rest = 0;
        int x, dummy;
        int[] bruch, resultat;

        /* Instantiieren der Bruchdarstellung */
        bruch = new int[stellen];

        /* Instantiieren des Ergebnisbruchs */
        resultat = new int[stellen];
```

```

/* Einlesen der Dezimalstellen */
for (int i=0; i < stellen; i++) {
    dummy = i + 1;
    /* Initialisieren der (i+1)-sten Dezimalstelle mit -1 */
    bruch[i] = -1;
    /* Solange eine Nachkommastelle einlesen, bis die Nachkommastelle im
Intervall [0..9] liegt */
    while (bruch[i] < 0 || bruch[i] > 9) {
        System.out.print("Bitte geben sie die " + dummy + ". Zahl ein:
");
        bruch[i] = IO.Eingabe();
    }
    resultat[i] = bruch[i];
}

/* Einlesen des Divisors */
System.out.print("Geben Sie einen Wert fuer x ein: ");
x = IO.Eingabe();

/* Berechnung des Ergebnisbruchs */
for (int i = 0; i < stellen; i++) {
    /* Es werden sukzessiv die Elemente des Arrays durch den Divisor
geteilt.
    Der ganzteilige Anteil wird an der Stelle i gespeichert. Der Rest
wird als
    Übertrag auf das Element der folgenden Stelle addiert. */
    rest = 10 * rest + resultat[i];
    resultat[i] = rest / x;
    rest = rest % x;
}

/* Ausgabe */
System.out.print("Ergebnis: 0.");
for (int i = 0; i < stellen; i++) {
    System.out.print(resultat[i]);
}
}
}

```

b)

```

/* Das Programm berechnet die negativen Potenzen -1 bis -10 einer positiven
* ganzen Zahl x. Dazu wird eine Dezimalbruchdarstellung verwendet. Dieser
* Dezimalbruch wird dabei sukzessiv durch die ingelezene Zahl dividiert.
* Autor: Thomas von der Maßen
* Umgebung: JDK 1.3.1, Windows 2000
* Erstellt: 14.11.2001
*/

```

```

public class NegativePotenzen {

    public static void main (String[] args) {
        final int stellen = 10;
        int rest = 0;
        int x, dummy;
        double wert = 0;
        int[] bruch, resultat;

        /* Instantiieren des Original- und des Ergebnisbruchs */
        bruch = new int[stellen];
        resultat = new int[stellen];

        /* Einlesen der Zahl dessen negative Potenzen berechnet werden sollen */
        x = 0;
        while (x <= 0) {

```

```

        System.out.print("Geben Sie bitte die Zahl ein: ");
        x = IO.Eingabe();
        wert = 1.0 / (double)x;
    }

    /* Initialisierung des Ergebnisbruchs durch die Darstellung von x hoch -1
*/
    for (int i = 0; i < stellen; i++) {
        wert = wert * 10.0;
        resultat[i] = (int)wert;
        wert = wert - resultat[i];
    }

    /* Berechnung der negativen Potenzen von -1 bis -10 */
    for (int potenz = 1; potenz <= 10; potenz++) {
        System.out.print(x + " hoch -" + potenz + " = 0.");
        for (int j = 0; j < stellen; j++) {
            System.out.print(resultat[j]);
        }
        System.out.println("");

        /* Es werden sukzessiv die Elemente des Arrays durch die Zahl x
geteilt.
        Der ganzzahlige Anteil wird an der Stelle i gespeichert. Der Rest
wird als
        Übertrag auf das Element der folgenden Stelle addiert. */
        for (int i = 0; i < stellen; i++) {
            rest = 10 * rest + resultat[i];
            resultat[i] = rest / x;
            rest = rest % x;
        }
    }
}

```

Aufgabe 3)

```

/* Das Programm ersetzt in einem Satz die Vokale nach einem bestimmten Schema.
* Die Wörter des Satzes werden als Kommandozeilenargumente der main-Methode
* übergeben. Der Satz mit den ersetzten Vokalen wird in einem Array gespeichert und
* ausgegeben.
* Autor: Thomas von der Maßen
* Umgebung: JDK 1.3.1, Windows 2000
* Erstellt: 14.11.2001
*/

public class Vokalersetzung {

    public static void main (String[] args) {
        String wort;
        String[] ergebnis = new String[args.length];
        char c;
        char[] buchstaben;

        for (int anzahlWoerter = 0; anzahlWoerter < args.length; anzahlWoerter++) {
            /* Speichern des aktuellen Wortes */
            wort = args[anzahlWoerter];

            /* Konvertierung des Strings in ein Array von Chars */
            buchstaben = wort.toCharArray();

```

```

/* Ersetze vokale nach vorgegebenem Schema */
for (int i = 0; i < buchstaben.length; i++) {
    c = buchstaben[i];
    switch (c) {
        case 'a': { buchstaben[i] = 'e'; break; }
        case 'A': { buchstaben[i] = 'E'; break; }
        case 'e': { buchstaben[i] = 'i'; break; }
        case 'E': { buchstaben[i] = 'I'; break; }
        case 'i': { buchstaben[i] = 'o'; break; }
        case 'I': { buchstaben[i] = 'O'; break; }
        case 'o': { buchstaben[i] = 'u'; break; }
        case 'O': { buchstaben[i] = 'U'; break; }
        case 'u': { buchstaben[i] = 'a'; break; }
        case 'U': { buchstaben[i] = 'A'; break; }
    }
}

/* Rekonstruiere String aus Chars */
wort = "";
for (int i = 0; i < buchstaben.length; i++) {
    wort = wort + buchstaben[i];
}
/* Speichern des aktualisierten Wortes in dem Ergebnis-Array */
ergebnis[anzahlWoerter] = wort;
}

/* Ausgeben des Ergebnis-Arrays */
for (int i = 0; i < ergebnis.length; i++) {
    System.out.print(ergebnis[i] + " ");
}
}
}

```

Aufgabe 4)

```

/* Das Programm berechnet das Produkt einer mxn- und einer nxp-Matrix.
 * Dazu werden zunächst die Dimensionen und die Werte der zu
 * multiplizierenden Matrizen durch Eingabe des Benutzers eingelesen.
 * Anschließend wird die Ergebnismatrix berechnet und das Produkt auf
 * dem Bildschirm ausgegeben.
 * Autor: Thomas von der Maßen
 * Umgebung: JDK 1.3.1, Windows 2000
 * Erstellt: 14.11.2001
 */

public class Matrixprodukt {

    public static int m, n, p;
    public static int[][] matrixA, matrixB, matrixC;

    public static void MatrizenEinlesen() {
        /* Einlesen der Dimensionen der Matrizen A und B */
        System.out.print("Geben Sie die Anzahl der Zeilen der ersten Matrix an: ");
        m = IO.Eingabe();
        System.out.print("Geben Sie die Anzahl der Spalten der ersten Matrix an: ");
        n = IO.Eingabe();
        System.out.print("Geben Sie die Anzahl der Spalten der zweiten Matrix an: ");
        p = IO.Eingabe();

        /* Instantiieren der Matrizen */
    }
}

```

```

matrixA = new int[m][n];
matrixB = new int[n][p];
matrixC = new int[m][p];

/* Einlesen der Eintraege von Matrix A zeilenweise */
System.out.println("Geben Sie die Eintraege der ersten Matrix ein: ");
for (int i = 0; i < m; i++) {
    for (int j = 0; j < n; j++) {
        matrixA[i][j] = IO.Eingabe();
    }
}

/* Einlesen der Eintraege von Matrix B zeilenweise */
System.out.println("Geben Sie die Eintraege der zweiten Matrix ein: ");
for (int i = 0; i < n; i++) {
    for (int j = 0; j < p; j++) {
        matrixB[i][j] = IO.Eingabe();
    }
}

}

public static void drucke() {
    /* Ausgabe von Matrix A */
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            System.out.print(matrixA[i][j] + "    ");
        }
        System.out.println("");
    }

    System.out.println("");
    System.out.println("    *    ");
    System.out.println("");

    /* Ausgabe von Matrix B */
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < p; j++) {
            System.out.print(matrixB[i][j] + "    ");
        }
        System.out.println("");
    }

    System.out.println("");
    System.out.println("    =    ");
    System.out.println("");

    /* Ausgabe von Matrix C */
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < p; j++) {
            System.out.print(matrixC[i][j] + "    ");
        }
        System.out.println("");
    }
}

}

public static void berechneProdukt() {
    /* Initialisierung von Matrix C */
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < p; j++) {
            matrixC[i][j] = 0;
        }
    }

    /* Berechnung des Matrizenprodukts A*B */
    for (int i = 0; i < m; i++) {

```

```

        for (int j = 0; j < p; j++) {
            for (int k = 0; k < n; k++) {
                matrixC[i][j] = matrixC[i][j] + matrixA[i][k] *
matrixB[k][j];
            }
        }
    }

    public static void main (String[] args) {
        MatrizenEinlesen();
        System.out.println("Das Ergebnis der Multiplikation der Matrizen A und B
lautet: ");
        berechneProdukt();
        drucke();
    }
}

```

Aufgabe 5)

```

/* Die Klasse realisiert eine Kugel und stellt spezifische Attribute
* und Operationen (Methoden) für eine Kugel bereit.
* Autor: Thomas von der Maßen
* Umgebung: JDK 1.3.1, Windows 2000
* Erstellt: 14.11.2001
*/

public class Kugel {

    public final double PI = 3.1415926535897932384626433832795;

    /* Attribute */
    public double radius = 0;
    public double gewicht = 0;
    public String farbe = "";

    /* Berechnung des Durchmessers */
    public double gibDurchmesser() {
        return (2 * radius);
    }

    /* Berechnung der Oberfläche */
    public double berechneOberflaeche() {
        return (4 * PI * radius * radius);
    }

    /* Berechnung des Volumens */
    public double berechneVolumen() {
        return (4.0 / 3.0 * PI * radius * radius * radius);
    }

    /* Berechnung des Umfangs */
    public double berechneUmfang() {
        return (gibDurchmesser() * PI);
    }
}

```

```

/* Das Programm benutzt die Klasse Kugel und erzeugt drei Instanzen
 * der Klasse Kugel, weist ihnen Attributausprägungen zu und
 * gibt Durchmesser, Oberfläche, Volumen und Umfang der Kugelobjekte aus
 * Autor: Thomas von der Maßen
 * Umgebung: JDK 1.3.1, Windows 2000
 * Erstellt: 14.11.2001
 */

public class Kugelprogramm {

    public static void main(String[] args) {

        Kugel kugell, kugel2, kugel3;
        kugell = new Kugel();
        kugel2 = new Kugel();
        kugel3 = new Kugel();

        /* Weise Kugel 1 Attributausprägungen zu */
        kugell.farbe = "blau";
        kugell.gewicht = 56;
        kugell.radius = 10;

        /* Weise Kugel 2 Attributausprägungen zu */
        kugel2.farbe = "rot";
        kugel2.gewicht = 12;
        kugel2.radius = 7.5;

        /* Weise Kugel 3 Attributausprägungen zu */
        kugel3.farbe = "gelb";
        kugel3.gewicht = 115;
        kugel3.radius = 23;

        /* Ausgabe der Ergebnisse */
        System.out.println("Kugel 1:");
        System.out.println("Farbe: " + kugell.farbe + " Gewicht: " +
kugell.gewicht);
        System.out.println("Durchmesser: " + kugell.gibDurchmesser());
        System.out.println("Umfang: " + kugell.berechneUmfang());
        System.out.println("Oberflaeche: " + kugell.berechneOberflaeche());
        System.out.println("Volumen: " + kugell.berechneVolumen());

        System.out.println("Kugel 2:");
        System.out.println("Farbe: " + kugel2.farbe + " Gewicht: " +
kugel2.gewicht);
        System.out.println("Durchmesser: " + kugel2.gibDurchmesser());
        System.out.println("Umfang: " + kugel2.berechneUmfang());
        System.out.println("Oberflaeche: " + kugel2.berechneOberflaeche());
        System.out.println("Volumen: " + kugel2.berechneVolumen());

        System.out.println("Kugel 3:");
        System.out.println("Farbe: " + kugel3.farbe + " Gewicht: " +
kugel3.gewicht);
        System.out.println("Durchmesser: " + kugel3.gibDurchmesser());
        System.out.println("Umfang: " + kugel3.berechneUmfang());
        System.out.println("Oberflaeche: " + kugel3.berechneOberflaeche());
        System.out.println("Volumen: " + kugel3.berechneVolumen());
    }
}

```