Aufgaben Programmiervorkurs
Übungsblatt 3

TECHNISCHE UNIVERSITÄT DARMSTADT

von Uli Fahrer & Dennis Albrecht

Wintersemester 2014/2015

Aufgabe 1 Multiple Choice

Kreuze zu jeder Antwort an, ob sie zutrifft (**w**) oder nicht (**f**).

Aufgabe 1.1 Arrays (Stufe 1)

w f

☐ ■ Ein bestehendes Array kann in seiner Länge verändert werden.

■ ☐ meinArray[0] liefert einen gültigen Wert.

☐ ■ meinArray[meinArray.length] liefert einen gültigen Wert.

☐ ■ Eine foreach-Schleife gibt die Elemente eines String-Arrays in alphabetischer Reihenfolge aus.

Antwort

Die zweite Aussage trifft zu.

Aufgabe 1.2 Funktionen (Stufe 2)

w f

■ ☐ Eine Funktion mit Rückgabewert muss auch einen Wert zurückgeben.

☐ ■ Eine void-Funktion kann Werte zurückgeben.

☐ ■ Eine Funktion kann auch mehrere Werte zurückgeben.

Antwort

Die erste Aussage trifft zu.

Aufgabe 2 Theoriefragen

Aufgabe 2.1 Funktionsköpfe (Stufe 1)

Welche der folgende Funktionsköpfe sind gültig.

a) public static void f1()

b) public static f2()

c) public static boolean f3(x, y)

d) public static int f4(int a, int b)

e) public static String f5(String, char)

f) public static String[] f6(String r, String... c)

Antwort

Die gültigen Funktionsköpfe sind f1, f4 und f6

Aufgabe 3 Arrays

Aufgabe 3.1 Ein Array anlegen (Stufe 1)

a) Schreibe ein Programm, das zuerst ein Array für 10 Elemente vom Typ char allokiert. Anschließend sollen diese 10 Elemente mit Benutzereingaben gefüllt werden. D.h. der Benutzer soll in der Lage sein nach und nach Zeichen einzugeben, welche in dem Array abgelegt werden. Achte dabei darauf, das die Größe des Arrays nicht überschritten wird. Gib die gespeicherten Zeichen anschließend nacheinander aus.

b) Teste dein Programm und gib der Reihe nach die folgenden Zeichen ein:
I, n, f, o, r, m, a, t, i, k.

Antwort

```
1  public class Array {
2      public static void main(String[] args) {
3          char[] array = new char[10];
4          for(int i=0;i<10;i++){
5              System.out.print("Gib␣Zeichen␣" + (i + 1) + "␣ein:␣");
6              array[i] = Input.readChar();
7          }
8          for (char c:array)
9              System.out.print(c);
10         System.out.println();
11     }
12 }
```

Aufgabe 4 Funktionen

Aufgabe 4.1 Taschenrechner Reloaded (Stufe 1)

In der vorigen Übung sollte ein simpler Taschenrechner programmiert werden. Nun wollen wir diesen Taschenrechner erweitern. Schreibe zuerst jeweils eine Funktion für jede der angebotenen Grundrechenarten (`public static int plus(3, 5)` soll beispielsweise 8 liefern). Es ist dir überlassen, ob du bei der Division eine Ganzzahl- oder Fließkommadivision machst. Im Anschluss sollen die Rechenanweisungen in der switch-case-Anweisung oder der if-Struktur durch Funktionsaufrufe ersetzt werden. Vergiss nicht, das Ergebnis auszugeben.

Antwort

siehe nächste Teilaufgabe

Aufgabe 4.2 Taschenrechner Reloaded II (Stufe 1)

Erweitere den Taschenrechner nun noch um einen weiteren Operator. Dabei soll auch das Potenzieren möglich sein. Dabei gilt $a^n = a \cdot a \cdot a \cdot \ldots \cdot a$ (d.h. a wird n-mal mit sich selbst malgenommen). Zum Aufruf der Potenzier-Funktion soll der Benutzer ^ (das kleine „Dach“ links oben auf der Tastatur) eingeben.

Antwort

```
public class Taschenrechner {
    public static void main(String[] args) {
        System.out.print("Wie lautet der erste Operand: ");
        int op1 = Input.readInt();
        System.out.print("Wie lautet der zweite Operand: ");
        int op2 = Input.readInt();
        System.out.print("Wie lautet die Operator: ");
        char op = Input.readChar();
        switch(op){
        case '+':
            System.out.println("Das Ergebnis lautet: " + plus(op1, op2));
            break;
        case '-':
            System.out.println("Das Ergebnis lautet: " + minus(op1, op2));
            break;
        case '*':
            System.out.println("Das Ergebnis lautet: " + mal(op1, op2));
            break;
        case '/':
            System.out.println("Das Ergebnis lautet: " + teilen(op1, op2));
            break;
        case '^':
            System.out.println("Das Ergebnis lautet: " + potenz(op1, op2));
            break;
        default:
            System.out.println("Keine gueltige Eingabe!");
        }
    }
    public static int plus(int op1, int op2) {
        return op1 + op2;
    }
    public static int minus(int op1, int op2) {
        return op1 - op2;
    }
    public static int mal(int op1, int op2) {
        return op1 * op2;
    }
    public static int teilen(int op1, int op2) {
        return op1 / op2;
    }
    public static int potenz(int op1, int op2) {
        int potenz = 1;
        for (int i = 1; i <= op2; i ++)
            potenz *= op1;
        return potenz;
    }
}
```

Aufgabe 4.3 Tabellen Ausgabe (Stufe 2)

Schreibt ein Programm, welches eine Tabelle von Potenzen ausgibt.

n	n^2	n^3
1	1	1
2	4	8
3	9	27

...

⋮

Dabei soll die Größe (also die höchste Potenz und die höchste Basis) variabel, also von einer Benutzereingabe abhängig sein. Zum Potenzieren kann die Funktion des Taschenrechners wiederverwendet werden. Wenn beide Klassen im selben Ordner liegen, kann die Potenz-Funktion ähnlich der Input-Funktionen direkt aus dem neuen Tabellen-Programm genutzt werden. Je nach vergebenen Namen kann der Funktionsaufruf etwa Taschenrechner.potenz(basis, exponent) lauten.

Antwort

```
1  public class Tabelle {
2      public static void main(String[] args) {
3          System.out.print("Gib die maximale Basis ein: ");
4          int basis = Input.readInt();
5          System.out.print("Gib den maximalen Exponent ein: ");
6          int exponent = Input.readInt();
7          System.out.print("n");
8          for (int i = 2; i <= exponent; i ++)
9              System.out.print("\tn^" + i);
10         System.out.println();
11         for (int b = 1; b <= basis; b ++) {
12             System.out.print(b);
13             for (int e = 2; e <= exponent; e ++)
14                 System.out.print("\t" + Taschenrechner.potenz(b,e));
15             System.out.println();
16         }
17     }
18 }
```

Aufgabe 4.4 Funktionen mit beliebig vielen Parameters (Stufe 2)

Schreibe eine Funktion die beliebig viele Parameter vom Type double übergeben bekommt und diese einfach nacheinander ausgibt.

Antwort

```
1 public class Parameter {
2     public static void main(String[] args) {
3         ausgabe(6.5f, 3.7f, 4.2f);
4     }
5     public static void ausgabe(double... werte) {
6         for (double d: werte)
7             System.out.println(d);
8     }
9 }
```

Aufgabe 5 Arrays und Funktionen

Aufgabe 5.1 Summe und Durchschnitt eines Arrays (Stufe 3)

Ergänze im folgende Code die fehlenden Codestellen damit die beiden Funktionen die Summe und den Durchschnitt errechnen.

```
1  public class Arrayfunktionen {
2      public static void main(String[] args) {
3          int array[] = {2,4,6,8,10,12,14,16,18};
4          System.out.println(summe(array));
5          System.out.println(durchschnitt(array));
6      }
7
8      public static int summe(int[] werte) {
9          //To-Do Rechne die Summe des Arrays aus
10     }
11
12     public static int durchschnitt(int[] werte) {
13         //To-Do Berechne den Durchschnitt mit Hilfe der Funktion summe
14     }
15 }
```

Antwort

```
1  public class Arrayfunktionen {
2      public static void main(String[] args) {
3          int array[] = {2,4,6,8,10,12,14,16,18};
4          System.out.println(summe(array));
5          System.out.println(durchschnitt(array));
6      }
7
8      public static int summe(int[] werte) {
9          int summe = 0;
10         for(int n:werte)
11             summe += n;
12         return summe;
13     }
14
15     public static int durchschnitt(int[] werte) {
16         return summe(werte) / werte.length;
17     }
18 }
```

Aufgabe 5.2 Summe und Durchschnitt eines Arrays II (Stufe 2)

Schreibe deine Funktionen aus dem vorigen Aufgabenteil nun so um, dass sie kein Array entgegennehmen sondern beliebig viele Integerwerte. Was fällt im Bezug auf die übergebenen Zahlen auf.

Antwort

```java
public class Arrayfunktionen2 {
    public static void main(String[] args) {
        int array[] = {2,4,6,8,10,12,14,16,18};
        System.out.println(summe(array));
        System.out.println(durchschnitt(array));
    }

    public static int summe(int... werte) {
        int summe = 0;
        for(int n:werte)
            summe += n;
        return summe;
    }

    public static int durchschnitt(int... werte) {
        return summe(werte) / werte.length;
    }
}
```

Es fällt auf, dass man einer Funktion mit beliebig vielen Parametern auch ein Array übergeben kann. Dieses Verhalten ist nur all zu verständlich, wenn man sich ansieht, wie innerhalb der Funktion auf die Parameter zugegriffen wird. Daher kann eine Funktion mit beliebig vielen Parametern eine gute Alternative sein, da dieses Konstrukt mächtiger ist (kann sowohl ein Array als auch eine Menge von Werten entgegennehmen). Allerdings gibt es dabei ein paar Einschränkungen. Beispielsweise ist ein Funktionskopf mit mehreren „beliebig-viele“-Parametern (function(int ... x, int ... y)) nicht möglich.