Kontrollfragen und Übungen Kapitel 4

1. Was versteht man im Zusammenhang mit der Programmierung unter dem Begriff Ausdruck?

Ein Ausdruck ist der kleinste ausführbare Teil eines Programms und ist eine Kombination aus Operatoren und Operanden. Er weißt Variablen und Konstanten Werte zu, berechnet einen Wert oder stellt eine logische Bedingung.

1. Woraus setzen sich Ausdrücke zusammen?

Aus Operatoren und Operanden.

1. Zählen sie unterschiedliche Operatoren auf. Unterscheiden sie dabei nach dem Typ der Operanden.

Arithmetisch: Plus, Minus, Mal, Geteilt, Restwert, Postinkrement, Präinkrement, Postdekrement, Prädekrement

Relational: Gleich, Ungleich, Größer, Kleiner, Größer Gleich, Kleiner Gleich,

Logisch: Nicht, Und, Oder, Und (mit SCE), Oder (mit SCE), Exklusiv Oder,

Zuweisend: Zuweisung, Additionszuweisung, Subtraktionszuweisung, Mulitplikationszuweisung, Divisionszuweisung, Modulozuweisung

1. Erläutern sie den Fragezeichenoperator anhand des folgenden Codings.   
   String X = (a==b) ? „ja“ : „nein“;

Der Fragezeichenoperator ist eine Art mini if-Anweisung. Ist String X = (a==b) true wird “Ja“ ausgeführt ist er false wird „nein“ ausgeführt

1. Definieren sie den Begriff Anweisung im Sinne der Programmierung.

Anweisungen sind elementare Arbeitsschritte eines Programms. Sie bestehen aus Ausdrücken.

1. Welche Möglichkeiten bietet ihnen die Programmiersprache Java um Verzweigungen zur realisieren?

If-Anweisunf, If-Else-Anweisung, If-Else-If-Anweisung, Switch-Anweisung,

1. Erläutern sie den Begriff „dangling else“.

Beschreibt das Problem, bei dem wenn die Optionale Klammerung weg gelassen wird, das else sich bei verschachtelten if-Anweisungen sich immer auf das letzte if bezieht und nicht auf das, auf dessen Level es eingerückt wurde.

1. Welche Bedeutung messen sie der Break-Anweisung im Zusammenhang mit der Switch Anweisung zu?

Das Break ist bei einer Switch-Anweisung sehr wichtig, da ohne Break, obwohl ein Case eingetreten ist, alle anderen Cases auch abgearbeitet werden.

1. Nenne sie unterschiedliche Schleifenarten in Java und beschreiben sie deren Verhalten.

Kopfgesteuerte Schleife prüft erst die Bedingung und führt dann den Schleifeninhalt durch, wenn diese true ist, prüft dann wieder und wiederholt diesen Vorgang bis die Bedingung nicht erfüllt ist.

Fußgesteuerte Schleife führt erst den Schleifeninhalt durch prüft dann die Bedingung und führt den Inhalt dann wieder durch, wenn die Bedingung true ist. Diesen Vorgang wiederholt sie bis die Bedingung false ist

Zählende Schleife Führt den Schleifeninhalt eine bestimmte Anzahl an Wiederholungen durch

1. Worin besteht der wesentliche Unterschied zwischen einer Kopf- und einer Fußgesteuerten Schleife?

Die Kopfgesteuerte Schleife prüft erst die Bedingung und die Fußgesteuert führt erst den Inhalt durch. Somit hat man bei unzutreffender Bedingung bei einer Kopfgesteuerten schleife 0 Durchläufe und bei einer Fußgesteuerten 1 Durchlauf.

1. Beschreiben sie den Schleifenkopf einer For-Schleife.

Der Schleifenkopf einer For-Schleife sieht wie folgt aus: for (init; test; update)

Der init-Teil initialisiert eine oder mehrere Zählervariablen, die nur in der Schleife bekannt sind

Der test- Teil bildet die Abbruchbedingung.

Der update-Teil verändert den / die Zähler.

1. Was bewirken die Break- und Continue-Anweiung innerhalb einer Schleife?

Break Führt zum Verlassen der Schleife  
macht mit dem ersten Befehl nach der Schleife weiter

Continue Schleifendurchlauf wird abgebrochen und die Schleife wird neu gestartet

1. Implementieren sie ihr Struktogramm zur Berechnung des größten gemeinsamen Teiler nach Euklid.

public class euklid {

public static void main(String[]args){

int zahlA = 128232212;

int zahlB = 21232342;

while ((zahlA - zahlB) !=0){

if (zahlA > zahlB){

zahlA = zahlA - zahlB;

}

else{

zahlB = zahlB -zahlA;

}

}

System.out.println(zahlA);

}

}

1. Implementieren sie ihren Pseudocode zur Berechnung einer Fakultät.

public class fakulaet {

public static void main(String[]Args) {

int zahlA = 1;

int zahlB = 2;

int zahlC = 4;

if (zahlA != zahlC) {

while (zahlB <= zahlC){

zahlA = zahlA \* zahlB;

zahlB++;

}

}

System.out.println(zahlA);

}

}

1. Implementieren sie ihr Struktogramm zur Bestimmung von Primzahlen mit dem Sieb des Eratosthenes.

public static void main(String[]Args) {

String s = JOptionPane.showInputDialog("Geben Sie eine Zahl ein:");

int zahl = Integer.parseInt(s);

int zahla= zahl+1;

int zahlb = 2;

int zahlc;

int zahld = 2;

int[] array = IntStream.range(1, zahla).toArray();

while (zahlb<=zahl){

while (zahlb\*zahld<=zahl) {

zahlc = (zahlb \* zahld) - 1;

array[zahlc] = 0;

zahld++;

}

zahld=2;

zahlb++;

}

int l = zahla-1;

for(int i =0; i<l; i++){

System.out.print(array[i]);

}

}

}

1. Formulieren sie eine zur folgenden Anweisung semantisch äquivalente Anweisung ohne das Wort „if“. return a==b ? false : true;

return a!=b ? true : false

1. Gegeben sind folgende Variablen: int a= 10; int b= 5; boolean z= false;

Füllen sie folgende Wahrheitstabelle aus:

|  |  |
| --- | --- |
| !z | True |
| a<20 | True |
| a == 2\*b | True |
| a%b != 0 | False |
| (a>b)&&z | False |
| (a>b)||z | True |
| !(a<b)^!z | False |
| (a<b)||((a%3<b) && !z) | True |

1. Übung Folgende Ausgabe soll erzeugt werden: 1 3 5 7 9   
   Lösen Sie diese Aufgabe mithilfe einer   
   a) While-Schleife   
   b) Do-while-Schleife   
   c) For-Schleife und des Modulo-Operators   
   d) For-Schleife ohne den Modulo-Operator  
     
    int a = 1;  
    System.*out*.println(a);  
    while (a<9) {  
    a = a + 2;  
    System.*out*.println(a);  
    }  
     
     
    a = 1;  
    do {  
    System.*out*.println(a);  
    a = a + 2;  
    }while (a<11);  
     
     
    a = 1;  
    for (;a%9!=0; a = a+2){  
    System.*out*.println(a);  
    }  
    System.*out*.println(a);  
     
     
    a = 1;  
    for (int b = 1; b<=5; b++){  
    System.*out*.println(a);  
    a= a + 2;  
    }  
    }  
   }
2. Übung Folgende Ausgabe soll erzeugt werden: 1 2 4 7 11 16 22 29 37   
   Lösen Sie diese Aufgabe mithilfe einer a) Do-while-Schleife b) For-Schleife

a = 1;

int b = 1;

do{

System.out.println(a);

a = a + b;

b++;

}while(a<=37);

a = 1;

for(int c = 1; c + a <= 37; c++){

System.out.println(a);

a = a + c;

}

1. Entwerfen Sie eine For-Schleife, die die Buchstaben A bis Z auf dem Bildschirm ausgibt

String [] buchstaben = {"A","B","C","D","E","F","G","H","I","J","K","L","M","N","O","P","Q","R","S","T","U","V","W","X","Y","Z",};

for(int z = 0; z<26; z++){

System.out.print(buchstaben[z]);

}

1. Welche mathematischen Operationen werden durch die Methoden meth1 und meth2 beschrieben und welche Werte haben zahl1 und zahl2 am Ende der Berechnung?

public class TestAlgorithmen {

public static void main(String[] args) {

int zahl1 = meth1(-5); = 25

int zahl2 = meth2(2, 6); = 64

System.out.println("Zahl 1:\t" + zahl1 +

"\nZahl 2:\t" + zahl2);

}

static int meth1( int a ) { <- entspricht a2

if( a<0 ) a=-a;

int rc=0;

for( int i=0; i<a; i++ )

rc += a;

return rc;

}

static int meth2( int a, int b ) { <- entspricht ab

// Wenn b<0 wird eine Fehlermeldung erzeugt -> 2. Semester

// Die Fehlermeldung führt zum Abbruch der Methode

if( b<0 )

throw new RuntimeException( "unerlaubter Wert"

);

if( b==0 ) return 1;

int rc=a;

for( int i=1; i<b; i++ )

rc \*= a;

return rc;

}

}

}

1. Setzen Sie Ihre Lösung zur Bestimmung der Fibonacci-Folge aus Übung 3 im Kapitel 1.2 in ein Programm um, wobei die Zahlenfolge dauerhaft in einem Array gespeichert werden soll

public class fibonacci {

public static void main(String[]Args) {

int a = 0;

int b = 1;

int l = 20;

int [] fibo = new int[l];

fibo[0]=a;

fibo[1]=b;

System.out.println(fibo[0]);

System.out.println(fibo[1]);

for (int z = 2; z<l; z++){

b += a;

a = b - a;

fibo[z]=b;

System.out.println(fibo[z]);

}

}

}

1. Übung Beschreiben Sie einen iterativen Algorithmus zur Berechnung der Wurzel einer Zahl nach dem Heron-Verfahren (babylonisches Wurzelziehen) als Struktogramm! Implementieren Sie sie.

public class heron {

public static void main(String[]args){

int a = 9;

double b = a;

double c;

double g = 0.000001;

do {

c=b;

b = (b + (a / b)) / 2;

}while ((c-b)>g);

System.out.println("Die Wurzel aus "+ a + " ist ca. " + b);

}

}

1. Schreiben sie einen Algorithmus zur Berechnung des Pascal‘schen Dreieicks

public class PascalDreieck {

public static void main(String[]args){

int zeile = 8;

int [][] dreieck = new int [zeile][zeile];

int a;

int b;

for(int z = 0; z<zeile; z++){

dreieck[0][z]=1;

dreieck[z][z]=1;

}

for (int z = 1; z< zeile;z++){

for (int x = z+1; x<zeile; x++){

a = z - 1;

b = x - 1;

dreieck[z][x] = dreieck[a][b] + dreieck[z][b];

}

}

for (int y = 0; y < zeile; y++){

System.out.println("Zeile " + y);

for (int x = 0; x < zeile; x++){

if (dreieck[x][y]>0){

System.out.println(dreieck[x][y]);

}

}

}

}

}