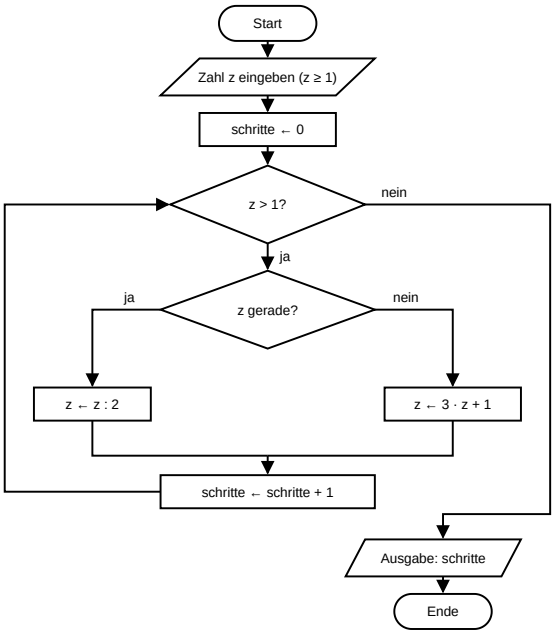
Expertenaufgaben: Algorithmen und Ablaufdiagramme

Zusatzaufgaben für Schnelle: knobeln, prüfen, begründen und selbst entwerfen

Dies sind Zusatzaufgaben für alle, die mit dem Arbeitsblatt schon fertig sind. Sie sind deutlich schwieriger: Du musst Abläufe genau verfolgen, Fehler aufspüren, Pläne verändern und eigene Algorithmen entwerfen. Du darfst Tabellen und Skizzen auf Schmierpapier anlegen. Wähle selbst, in welcher Reihenfolge du die Aufgaben bearbeitest.



Ablaufplan D: Ein geheimnisvoller Zähler (z : 2 bedeutet: z wird durch 2 geteilt; z gerade? bedeutet: Bei der Division durch 2 bleibt kein Rest.)

1 Führe Ablaufplan D für z = 3 und für z = 1 aus. a) Notiere für z = 3 nach jedem Durchlauf den neuen Wert von z und die Ausgabe. b) Was wird für z = 1 ausgegeben? Begründe, warum.

2 Forscheraufgabe: Lass Ablaufplan D für alle Startzahlen z = 1 bis 9 laufen und lege eine Tabelle an (Startzahl → Ausgabe). a) Welche Startzahl unter 10 braucht die meisten Schritte? b) Was fällt dir auf, wenn du die Ausgaben für z = 2, 4 und 8 vergleichst? Erkläre die Auffälligkeit.

3 Bisher hat noch niemand bewiesen, dass Ablaufplan D für jede Startzahl irgendwann bei „nein“ ankommt (diese Frage ist als Collatz-Vermutung bekannt). Computer haben sehr viele Startzahlen geprüft und immer ein Ende gefunden. Erkläre, warum man bei Plan D trotzdem nicht sicher sagen kann, dass er ein Algorithmus im Sinne der Definition (eindeutig und nach endlich vielen Schritten endend) ist. Warum reicht Ausprobieren hier nicht aus?

4 Ablaufplan B vom Arbeitsblatt (summe ← 0, zähler ← 1, Schleife mit „zähler ≤ 4?“, in der Schleife summe ← summe + zähler und zähler ← zähler + 1, danach Ausgabe von summe) soll umgebaut werden: Er soll die Summe aller geraden Zahlen von 2 bis 10 berechnen (2 + 4 + 6 + 8 + 10). a) Beschreibe, welche drei Stellen du im Plan änderst. b) Gib die Werte von summe nach jedem Durchlauf an und nenne die Ausgabe. c) Welche Änderung würdest du vornehmen, um die Summe aller Zahlen von 1 bis 100 zu berechnen? Was wird dann ausgegeben?

5 Fehlersuche an Ablaufplan B: a) Was passiert, wenn man das Rechteck „zähler ← zähler + 1“ einfach weglässt (der Pfeil führt dann direkt von „summe ← summe + zähler“ zurück zur Raute)? Begründe mit den Werten der Variablen. b) Was passiert, wenn die Anweisung „summe ← 0“ nicht am Anfang, sondern innerhalb der Schleife vor „summe ← summe + zähler“ steht? Welche Ausgabe entsteht dann? c) Welche der beiden Änderungen verletzt eine Grundeigenschaft eines Algorithmus? Welche?

6 Ordne jedem Fachbegriff die passende Erklärung zu.

1. Variable	____	A) Anweisung, die einen berechneten Wert in einer Variablen speichert (z. B. zähler ← zähler + 1)
2. Zuweisung	____	B) Wiederholung, deren Bedingung nie falsch wird, sodass der Ablauf nicht endet
3. Schleife	____	C) Stelle, an der abhängig von einer Bedingung zwischen zwei Wegen gewählt wird
4. Endlosschleife	____	D) Teil des Ablaufs, der so lange wiederholt wird, wie eine Bedingung erfüllt ist
5. Verzweigung	____	E) benannter Speicherplatz, dessen Wert sich im Ablauf ändern kann

7 Entscheide, ob die Aussagen wahr oder falsch sind. Überlege bei Aussagen zu Plan D genau mit eigenen Rechnungen.

	richtig	falsch
Für z = 4 gibt Ablaufplan D die Zahl 2 aus.	☐	☐
Für z = 5 gibt Ablaufplan D die Zahl 6 aus.	☐	☐
Ein Ablauf, der für eine erlaubte Eingabe in einer Endlosschleife landet, erfüllt nicht die Definition eines Algorithmus.	☐	☐
Ein Problem kann nur durch genau einen richtigen Ablaufplan gelöst werden.	☐	☐

8 Transferaufgabe „Zahlenraten“: Ein Mitschüler denkt sich eine ganze Zahl von 1 bis 100 aus. Du darfst nur Fragen der Form „Ist deine Zahl größer als x?“ stellen, die mit ja oder nein beantwortet werden. a) Entwirf einen Algorithmus (Handlungsvorschrift oder Beschreibung eines Programmablaufplans mit Schleife und Raute), der die Zahl sicher findet. Nenne die Variablen, die du brauchst. b) Wie viele Fragen brauchst du mit deiner Strategie höchstens für den Bereich 1 bis 100? Begründe. c) Wie viele wären es höchstens für den Bereich 1 bis 1000?

Lösungen

Expertenaufgaben: Algorithmen und Ablaufdiagramme · Informatik · Klasse 7

Für dich als Lehrkraft: leichtere Fassung in einer Minute

QR scannen oder lehreule.de/anpassen/algorithmen-ablaufdiagramme-klasse-7 öffnen: kostenloses Konto, dann schreibt Lehreule die leichtere Fassung mit Hilfen — oder Zusatzaufgaben für Schnelle, eine Word-Datei, einen QR-Code für die Klasse.

Aufgabe 1: a) Start: z = 3, schritte = 0.
1. Durchlauf: 3 ist ungerade, z = 3 · 3 + 1 = 10, schritte = 1.
2. Durchlauf: 10 ist gerade, z = 5, schritte = 2.
3. Durchlauf: 5 ist ungerade, z = 16, schritte = 3.
4. Durchlauf: z = 8, schritte = 4.
5. Durchlauf: z = 4, schritte = 5.
6. Durchlauf: z = 2, schritte = 6.
7. Durchlauf: z = 1, schritte = 7.

Dann ist z > 1 falsch, der Weg „nein“ führt zur Ausgabe. Ausgabe: 7.
b) Für z = 1 ist die Bedingung „z > 1?“ sofort falsch. Die Schleife wird kein einziges Mal durchlaufen, schritte bleibt 0. Ausgabe: 0.

Aufgabe 2: a) Tabelle (Startzahl → Ausgabe): 1 → 0, 2 → 1, 3 → 7, 4 → 2, 5 → 5, 6 → 8, 7 → 16, 8 → 3, 9 → 19. Die meisten Schritte braucht z = 9 mit 19 Schritten (9 → 28 → 14 → 7, danach 16 weitere Schritte wie bei z = 7).
b) Für 2, 4 und 8 ergeben sich 1, 2 und 3 Schritte. Bei Zweierpotenzen ist jede Zahl gerade, es wird immer nur halbiert. Man braucht so viele Schritte, wie man halbieren muss, bis 1 erreicht ist: 8 → 4 → 2 → 1 sind 3 Schritte. Die Zahl 3 · z + 1 wird nie benutzt.

Aufgabe 3: Zu einem Algorithmus gehört, dass er für jede erlaubte Eingabe nach endlich vielen Schritten endet. Bei Plan D kann man das nur für einzelne Startzahlen durch Ausprobieren nachweisen. Es gibt unendlich viele Startzahlen, und man kann nie alle testen. Solange niemand beweist, dass jede Folge bei 1 landet, kann es eine Startzahl geben, bei der die Schleife nie endet (z. B. weil sich die Werte immer weiter vergrößern oder im Kreis laufen). Ausprobieren liefert nur Beispiele, keinen Beweis für alle Fälle. Deshalb ist unklar, ob Plan D die Forderung „endet immer“ erfüllt.

Aufgabe 4: a) 1. Der Startwert wird geändert: zähler ← 2 statt zähler ← 1. 2. Der Zähler wird in jedem Durchlauf um 2 erhöht: zähler ← zähler + 2. 3. Die Bedingung wird geändert: zähler ≤ 10? statt zähler ≤ 4?.
b) summe nach den Durchläufen: 2, 6, 12, 20, 30. Es gibt 5 Durchläufe. Ausgabe: 30.
c) Man behält zähler ← 1 und zähler ← zähler + 1 bei und ändert nur die Bedingung in „zähler ≤ 100?“. Ausgabe: 1 + 2 + … + 100 = 5050.

Aufgabe 5: a) Der Zähler bleibt immer 1. Die Bedingung „zähler ≤ 4?“ ist immer wahr, der Weg „nein“ wird nie genommen. summe wächst ständig weiter (1, 2, 3, …). Das Programm läuft endlos und gibt nie etwas aus.
b) Bei jedem Durchlauf wird summe zuerst auf 0 gesetzt, dann wird nur der aktuelle zähler addiert. Die vorherigen Summanden gehen verloren. Im letzten Durchlauf ist zähler = 4, also ist summe = 4. Ausgabe: 4 statt 10. Der Plan endet zwar, liefert aber ein falsches Ergebnis.
c) Die Änderung aus a) verletzt die Grundeigenschaft, dass ein Algorithmus nach endlich vielen Schritten endet (Endlosschleife). Die Änderung aus b) verletzt diese Eigenschaft nicht, der Plan ist aber für die Aufgabe „Summe bilden“ falsch.

Aufgabe 6: 1 → E, 2 → A, 3 → D, 4 → B, 5 → C

Aufgabe 7: Für z = 4 gibt Ablaufplan D die Zahl 2 aus. → **richtig**
Für z = 5 gibt Ablaufplan D die Zahl 6 aus. → **falsch**
Ein Ablauf, der für eine erlaubte Eingabe in einer Endlosschleife landet, erfüllt nicht die Definition eines Algorithmus. → **richtig**
Ein Problem kann nur durch genau einen richtigen Ablaufplan gelöst werden. → **falsch**

Aufgabe 8: a) Variablen: unten ← 1, oben ← 100. Schleife: Solange unten < oben gilt, berechne mitte als ganzzahlig abgerundeten Wert von (unten + oben) : 2 und frage „Ist die Zahl größer als mitte?“. Bei ja: unten ← mitte + 1. Bei nein: oben ← mitte. Wenn unten = oben ist, endet die Schleife (Raute „unten < oben?“ mit nein), und die Ausgabe lautet: Die Zahl ist unten. Im Plan stehen: Oval Start, Zuweisungen für unten und oben, Raute „unten < oben?“, Rechteck für mitte, Raute „Zahl größer als mitte?“ mit zwei Zuweisungen, Rückführungspfeil zur ersten Raute, Ausgabe, Oval Ende.
b) Jede Frage halbiert den Bereich ungefähr. 100 → höchstens 50 → 25 → 13 → 7 → 4 → 2 → 1 Zahl. Das sind höchstens 7 Fragen. Zur Begründung: 2^7 = 128 ≥ 100, aber 2^6 = 64 < 100.
c) Es sind höchstens 10 Fragen, denn 2^10 = 1024 ≥ 1000, aber 2^9 = 512 < 1000. Der Bereich wird zehnfach größer, aber es kommen nur etwa 3 Fragen hinzu.