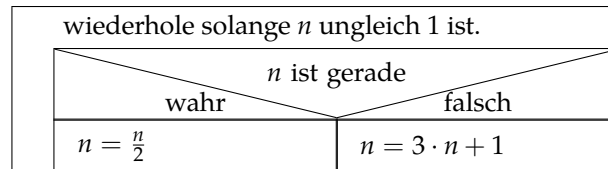


## Abitur 2019 IV

Das Collatz-Problem ist ein immer noch ungelöstes Problem der Mathematik. Dabei geht es um Zahlenfolgen, die nach folgendem Algorithmus gebildet werden, wobei der Eingabewert  $n$  eine natürliche Zahl größer 0 ist:

collatzfolge( $n$ )



Obwohl der Algorithmus sehr einfach ist, ist bis heute ungeklärt, ob er tatsächlich bei jedem beliebigen Startwert von  $n$  nach endlich vielen Durchläufen der Wiederholung terminiert.

- (a) Geben Sie die Zahlenfolge an, die man mit dem Startwert 7 erhält, wenn  $n$  nach jedem Durchlauf der Wiederholung ausgegeben wird.

7 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1

- (b) Beschreiben Sie, wie man mithilfe der ganzzahligen Division ohne Rest prüfen kann, ob eine Zahl  $a$  durch eine andere Zahl  $b$  teilbar ist.

Wenn man das Ergebnis der Division der beiden Zahlen  $a$  und  $b$  mit  $b$  multipliziert und nach der Multiplikation als Ergebnis wieder die Zahl  $a$  feststeht, dann handelt es sich um eine Division ohne Rest, ergibt sich eine Zahl, die kleiner als  $a$  ist, so handelt es sich um eine Division mit Rest.

- (c) Geben Sie ein Programm für die Registermaschine an, das den gegebenen Algorithmus `collatzfolge(n)` umsetzt, wobei zusätzlich die Anzahl der Durchläufe der Wiederholung bestimmt werden soll. Der Startwert für  $n$  steht am Anfang bereits in Speicherzelle 100.

Ohne Modulo	Mit Modulo
<pre> 1  # n:      100 2  # anzahl: 101 3 4          LOADI 0 5          STORE 101 6 7  # WHILE n &lt;&gt; 1 8  solange:  LOAD 100 9            SUBI 1 10           JMPZ ende 11 12 # anzahl := anzahl + 1; 13 zaehler:  LOAD 101 14           ADDI 1 15           STORE 101 16 17 # Poorman's Modulo 18 # IF (n % 2) = 0 THEN 19 modulo:   LOAD 100 20           DIVI 2 21           MULI 2 22           SUB 100 23           JMPN ist_ungerade 24 25 # n := n / 2; 26 ist_gerade: LOAD 100 27            DIVI 2 28            STORE 100 29            JMP solange 30 31 # n := 3 * n + 1; 32 ist_ungerade: LOAD 100 33              MULI 3 34              ADDI 1 35              STORE 100 36              JMP solange 37 38 ende:      HOLD </pre>	<pre> 1  # n:      100 2  # anzahl: 101 3 4          LOADI 0 5          STORE 101 6 7  # WHILE n &lt;&gt; 1 8  solange:  LOAD 100 9            CMPI 1 10           JMPZ ende 11 12 # IF (n % 2) = 0 THEN 13 bedingung: LOAD 100 14            MODI 2 15            JMPNZ ist_ungerade 16 17 # n := n / 2; 18 ist_gerade: LOAD 100 19            DIVI 2 20            STORE 100 21            JMP zaehler 22 23 # n := 3 * n + 1; 24 ist_ungerade: LOADI 3 25              MUL 100 26              ADDI 1 27              STORE 100 28 29 # anzahl := anzahl + 1; 30 zaehler:   LOAD 101 31            ADDI 1 32            STORE 101 33            JMP solange 34 35 ende:      HOLD </pre>

## Minisprache

```
1 PROGRAM collatz;
2 VAR n, anzahl;
3
4 BEGIN
5   n := 7;
6   anzahl := 0;
7   WHILE n <> 1 DO
8     IF (n % 2) = 0 THEN
9       n := n / 2;
10    ELSE
11      n := 3 * n + 1;
12    END;
13    anzahl := anzahl + 1;
14  END
15 END collatz.
```

## Java

```
3 public class Collatz {
4   public static void main(String[] args) {
5     int n = 7;
6     int anzahl = 0;
7     while (n != 1) {
8       if (n % 2 == 0) {
9         n = n / 2;
10      } else {
11        n = 3 * n + 1;
12      }
13      anzahl++;
14      System.out.println(n);
15    }
16    System.out.println("Anzahl an Durchläufen " + anzahl);
17  }
18 }
```