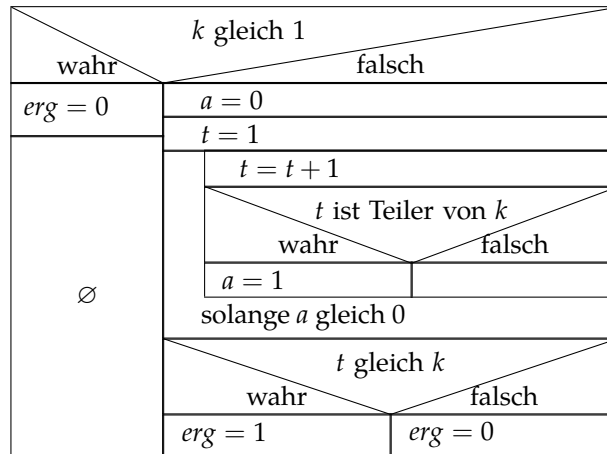


## Abitur 2017 IV

In dem folgenden Struktogramm wird ein Algorithmus dargestellt, der erkennt, ob eine natürliche Zahl  $k$  eine Primzahl ist. In diesem Fall wird in die Speicherzelle  $erg$  die Zahl 1 abgelegt, sonst 0.



- (a) Stellen Sie die Veränderung der Variablenwerte bei Ablauf dieses Algorithmus jeweils für die Startwerte  $k = 5$  und  $k = 15$  durch zwei Speicherbelegungstabellen wie nachfolgend gezeigt dar.

Anweisung	k	a	t	erg
	5			
$a = 0$		0		
$t = 1$			1	
$t = t + 1$	5		2	

$k = 5$

Anweisung	k	a	t	erg
	5			
$a = 0$		0		
$t = 1$			1	
$t = t + 1$			2	
$t = t + 1$			3	
$t = t + 1$			4	
$t = t + 1$			5	
$a = 1$		1		
$erg = 1$				1

$k = 15$

Anweisung	k	a	t	erg
	15			
$a = 0$		0		
$t = 1$			1	
$t = t + 1$			2	
$t = t + 1$			3	
$t = t + 1$			4	
$t = t + 1$		1	5	
$a = 1$		1		
$erg = 0$				0

Im Folgenden soll ein Programm für diese Maschine erstellt werden, das den dargestellten Algorithmus umsetzt. Der Wert von  $k$  soll in Speicherzelle 101, der von  $a$  in 102, der von  $t$  in 103 und der von  $erg$  in 104 gespeichert werden.

- (b) Betrachten Sie die folgende kurze Sequenz; xx steht dabei für ein geeignetes Sprungziel.

```
1  LOAD 101
2  MOD 103
3  JMPP xx
4  LOADI 1
5  STORE 102
```

Geben Sie an, welcher Teil des Algorithmus damit umgesetzt wird.

Die Bedingung „ $t$  ist Teiler von  $k$ “.

- (c) Setzen Sie unter Verwendung der Sequenz aus Teilaufgabe 2b den gesamten Algorithmus in ein Programm für die gegebene Registermaschine um.

**Assembler**

```
1  # k: 101
2  # a: 102
3  # t: 103
4  # erg: 104
5
6  # IF k = 1 THEN
7      LOADI 101
8      CMPI 1
9      JMPZ ist_nicht_prim
10
11 # a := 1;
12     LOADI 0
13     STORE 102
14
15 # t := 1;
16     LOADI 1
17     STORE 103
18
19 # t := t + 1;
20 erhoehe_t:  LOAD 103
21             ADDI 1
22             STORE 103
23
24 # IF (k % t) = 0 THEN
25     LOAD 101
26     MOD 103
27     JMPP solange_bed
28
29 # a := 1;
30     LOADI 1
31     STORE 102
32
33 # UNTIL a = 0;
34 solange_bed:  LOAD 102
35             JMPZ erhoehe_t
36
37 # IF t = k THEN;
38     LOAD 103
39     CMP 101
```

```

40                                JMPZ ist_prim
41
42 # erg := 0;
43 ist_nicht_prim: LOADI 0
44                STORE 104
45                JMP ende
46
47 # erg := 1;
48 ist_prim:      LOADI 1
49                STORE 104
50
51 ende:          HOLD

```

## Minisprache

```

1 PROGRAM primzahl;
2 VAR k, a, t, erg;
3
4 BEGIN
5   k := 5;
6   IF k = 1 THEN
7     erg := 0;
8   ELSE
9     a := 0;
10    t := 1;
11    REPEAT
12      t := t + 1;
13      IF (k % t) = 0 THEN
14        a := 1;
15      END
16    UNTIL a = 0;
17
18    IF t = k THEN
19      erg := 1;
20    ELSE
21      erg := 0;
22    END
23  END;
24 END primzahl.

```

## Java

```

5 public static boolean istPrimzahl(int k) {
6   if (k == 1)
7     return false;
8   int a = 0;
9   int t = 1;
10
11   do {
12     t++;
13     if (k % t == 0) {
14       a = 1;
15     }
16   } while (a == 0);
17
18   if (t == k) {
19     return true;
20   } else {

```

```
21     return false;  
22 }  
23 }
```