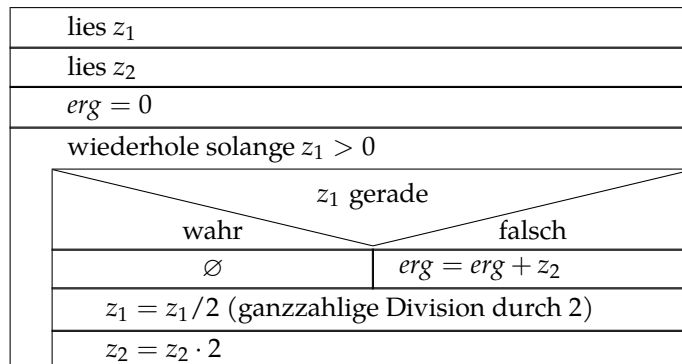


Abitur 2015 IV

Auf dem ägyptischen *Papyrus Rhind*, der etwa auf das Jahr 1550 v. Chr. datiert wird, ist eine Möglichkeit zur Multiplikation zweier natürlicher Zahlen z_1 und z_2 beschrieben. Als Struktogramm lässt sich dieser Algorithmus folgendermaßen darstellen:



Das berechnete Produkt steht nach Abarbeitung des Algorithmus in der Variablen *erg*.

- (a) Berechnen Sie mithilfe der beschriebenen ägyptischen Multiplikation schrittweise das Produkt aus $z_1 = 13$ und $z_2 = 5$

z_1	z_2	erg
13	5	0
		5
6	10	
3	20	
		25
1	40	
		65
0	80	

- (b) Nennen Sie die wesentliche Idee des Speichermodells eines Rechners, der nach dem von-Neumann-Prinzip aufgebaut ist. Geben Sie einen Vor- und einen Nachteil dieses Speichermodells an.

Wesentliche Idee des Speichermodells eines Rechners, der nach dem von-Neumann-Prinzip gebaut ist: Programme und Daten sind im selben Speicher, wobei der Hauptspeicher aus Zellen gleicher Größe besteht.

Vorteil:

Streng sequentieller Ablauf von Befehlen ist ein Vorteil, weil zu jedem Zeitpunkt klar ist, welcher Schritt durchgeführt wird.

Nachteil:

Der von-Neumann-Flaschenhals, weil alle Daten über denselben Bus

weitergeleitet werden müssen und der Ablauf deshalb eine gewisse Zeit benötigt.

- (c) Bestätigen Sie anhand zweier Beispiele, dass mithilfe des folgenden Programmausschnitts entschieden werden kann, ob Speicherzelle 101 eine gerade oder ungerade Zahl enthält.

	Beispiel für gerade Zahl	Beispiel für ungerade Zahl
LOAD 101	6=110	5=101
SHRI 1	3=011	2=010
SHLI 1	6=110	4=100
SUB 101	0	not zero für ungerade

	Beispiel für gerade Zahl	Beispiel für ungerade Zahl
LOAD 101	14=1110	17=10001
SHRI 1	7=0111	8=01000
SHLI 1	14=1110	16=10000
SUB 101	0	not zero für ungerade

	Beispiel für gerade Zahl	Beispiel für ungerade Zahl
LOAD 101	44=101100	25=11001
SHRI 1	22=010110	12=01100
SHLI 1	44=101100	24=11000
SUB 101	0	not zero für ungerade

- (d) Schreiben Sie ein Programm für die angegebene Registermaschine, das den Algorithmus des *Papyrus Rhind* umsetzt. Gehen Sie dabei davon aus, dass die beiden positiven ganzzahligen Faktoren z_1 und z_2 bereits in den Speicherzellen 101 und 102 stehen und dass alle weiteren nicht vom Programm belegten Speicherzellen mit dem Wert 0 vorbelegt sind.

```

1  Start:  LOADI 13
2          STORE 101
3          LOADI 5
4          STORE 102
5
6  Wdh:    LOAD 101
7          JMPZ end
8
9  Bed:    SHRI 1
10         SHLI 1
11         SUB 101
12         CMPI 0
13         JMPZ true
14
15  false:  LOAD erg
16         ADD 102
17         STORE erg
18
19  true:   LOAD 101
20         DIVI 2
21         STORE 101
22         LOAD 102
23         MULI 2
24         STORE 102
25         JMP Wdh
26

```

```

27  end:  LOAD erg
28      HOLD
29
30  erg:  WORD 0

3  public class AegyptischeMultiplikation {
4
5
6      public static int multipliziereIterativ(int z1, int z2) {
7          int ergebnis = 0;
8          while (z1 > 0) {
9              if (z1 % 2 == 1) {
10                 ergebnis = ergebnis + z2;
11             }
12             z1 = z1 / 2;
13             z2 = z2 * 2;
14         }
15         return ergebnis;
16     }
17
18     public static int multipliziereRekursiv(int z1, int z2) {
19         if (z1 == 1)
20             return z2;
21         int z1Alt = z1;
22         int z2Alt = z2;
23
24         z1 = z1 / 2;
25         z2 = z2 * 2;
26
27         if (z1Alt % 2 == 1)
28             return z2Alt + multipliziereRekursiv(z1, z2);
29         else
30             return multipliziereRekursiv(z1, z2);
31     }
32
33     public static void main(String[] args) {
34         System.out.println(multipliziereIterativ(9, 9));
35         System.out.println(multipliziereRekursiv(9, 9));
36     }
37 }

```