**Übersicht Strukturen der Programmierung**

**Sequenz**

Die Sequenz ist die einfachste Grundstruktur eines Programms und ist eine lineare, also nicht verzweigende und sich nicht wiederholende Programm-/Anweisungsfolge. (Meist nach EVA-Prinzip).

Beispiel:

Eingabe der Größe a durch den User

Verarbeitung (Verdopplung) der Größe a und speichern in der Größe e

Ausgabe der Größe e

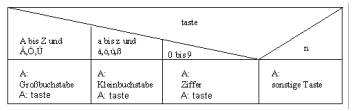
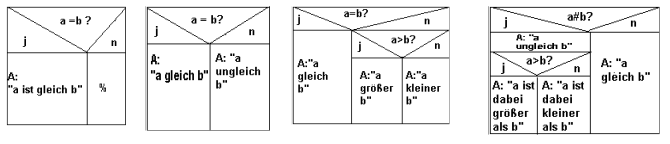
**Verzweigungen**

In Programmen müssen in Abhängigkeit von Ereignissen Fallunterscheidungen erfolgen. So müssen z.B. zur Fehlervermeidung bei mathematischen Aufgaben bestimmte Eingabefehler abgefangen werden. (z.B.: wenn Divisor = 0, dann Fehlerausgabe)

**einfache/zweiseitige Verzweigung**

Schon die einfache Verzweigung erlaubt eine sehr differenzierte Reaktion auf verschiedenste Programmsituationen.

Syntaxdiagramm der IF .. THEN .. ELSE .. - Anweisung:



**Mehrfachverzweigung**

In Abhängigkeit vom aktuellen Wert des ordinalen Selektors wird zu der Anweisungsfolge verzweigt, deren Konstante(n)(bereich) dem aktuel-

len Selektorwert entspricht. Ist der aktuelle Wert nicht in dieser Liste enthalten, wird

die Anweisungsfolge von ELSE ausgeführt.

**Schleifen**

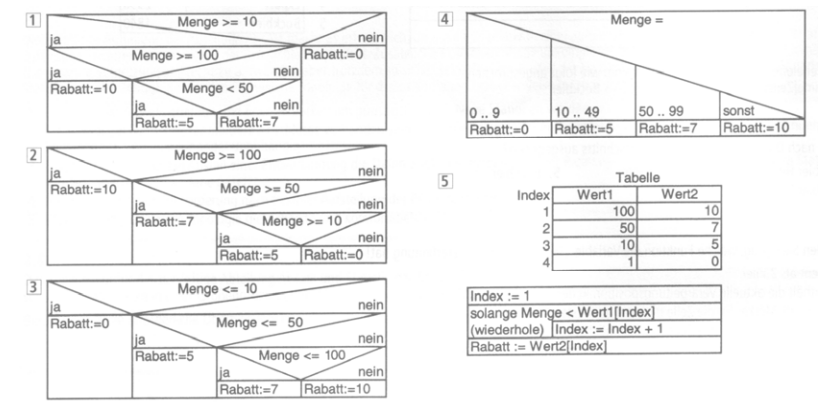
Schleifen dienen der Wiederholung von Programmteilen. Damit keine unendlichen Wiederholungen stattfinden, muss jede Schleife unter einer bestimmten Abbruchbedingung enden. Diese Abbruchbedingung muss auch tatsächlich erreicht werden, d.h. die Bedingung muss im Schleifenkörper aufgebaut werden. Die Überprüfung dieser Abbruchbedingung kann vor, nach oder innerhalb der Wiederholung der Programmsequenz erfolgen

| Schleifentypen | Beschreibung | Struktogramm |
| --- | --- | --- |
| Kopfgesteuert  while(Bedingung)  {  } | Wiederholt die Anweisungen des WHILE-Blockes bis die Abbruchbedingung wahr  (TRUE) ist.  (Prüfung am Anfang) |  |
| Fußgesteuert  do  { }  while(Bedingung) | Solange die Wiederholungsbedingung wahr (TRUE) ist, werden die Anwei- sungen zwischen DO und WHILE wiederholt.  => Abbruch wenn Bedingung = FALSE (Prüfung am Ende) |  |
| Zählergesteuert  for(i=0;i<5;i++)  {  } | Zählschleife. bei der die Anweisungen im FOR-Block wiederholt werden. Dabei wird eine Zählvariable mit einem bestimmten Anfangswert um einen bestimmten Betrag erhöht/erniedrigt. Abbruch erfolgt, wenn die Zählvariable den Endwert über-/unterschritten hat. |  |

Übungsaufgaben Struktogramme

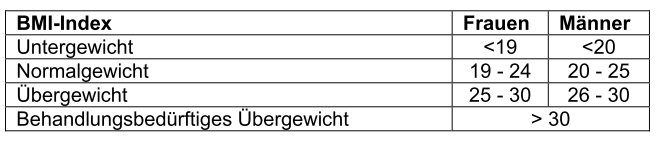
1) Struktogramm „Brauerei“

Eine Brauerei gewährt ihren Kunden bei Mindestabnahme von 10 Kästen 5%, bei Mindestabnahme von 50 Kästen 7% und bei Mindestabnahme von 100 Kästen 10% Rabatt. Diese Regelung soll durch einen Programmbaustein, z. B. bei der Fakturierung, automatisch berücksichtigt werden. Kontrollieren Sie die gegebenen Struktogramme daraufhin, ob sie die gegebene Regelung logisch richtig umsetzen. Prüfen Sie, welches Struktogramm fehlerhaft ist.



2) Struktogramm "bmi-rechner"

Entwickeln Sie ein Struktogramm, das den BMI (Body-Maß-Index) des Benutzers berechnet und "einordnet". Der BMI ist ein Maß für das Gewicht in Relation zur Körpergröße, er wird wie folgt berechnet (Gewicht in kg und Körpergröße in cm): bmi = gewicht \* 10000 / groesse²



3) Struktogramm „Anruf“

Entwickeln Sie ein Struktogramm, das den Ablauf eines Telefongesprächs wiedergibt. Stellen Sie Wiederholungsversuche (falls niemand abnimmt!!) durch unterschiedliche Schleifentypen dar und ermöglichen Sie einen Anrufbeantworter.

4) Struktogramm „Wurzel“

Entwickeln Sie ein Struktogramm, das die Quadratwurzel aus einer positiven ganzen Quadratzahl zieht.

Eingabe: ganzzahlige Quadratzahl (z.B. 36).

Ausgabe: Wurzel daraus (z.B. 6).

Anmerkung: Wenn die eingegebene Zahl keine Quadratzahl ist (z.B. 37), dann soll die Ausgabe eine Fehlermeldung sein („keine Quadratzahl“).

5) Struktogramm „Binärzahlen“

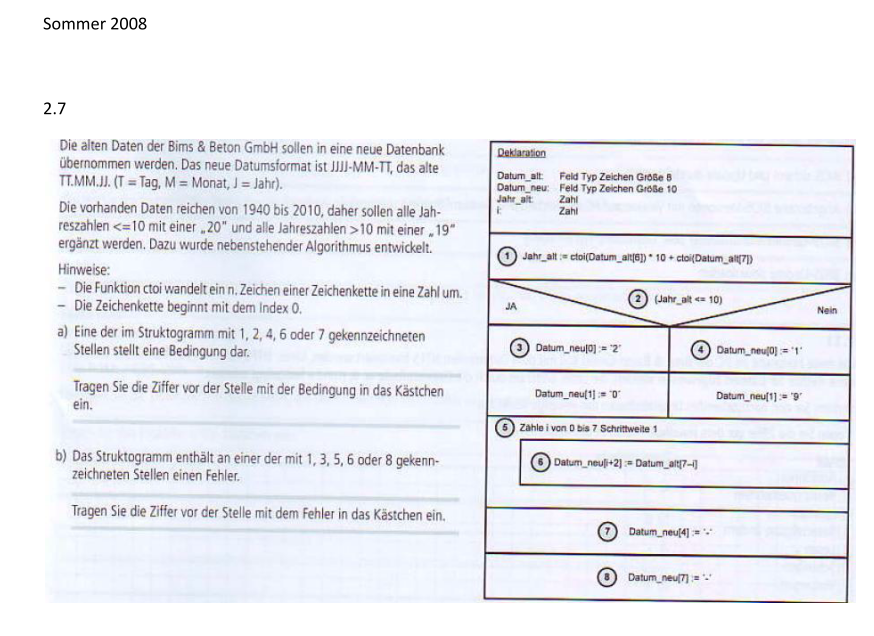
Erstellen Sie ein Struktogramm, das aus einer ganzen, positiven Zahl (0 bis 255) den Binärcode ermittelt. So ergibt die Zahl Null den Binärcode 0000 0000, die Zahl 255 den Binärcode 1111 1111.

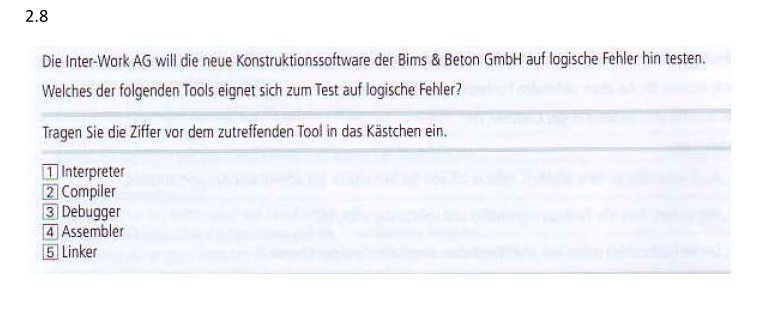
6) Struktogramm „Modula – Restwertmethode“

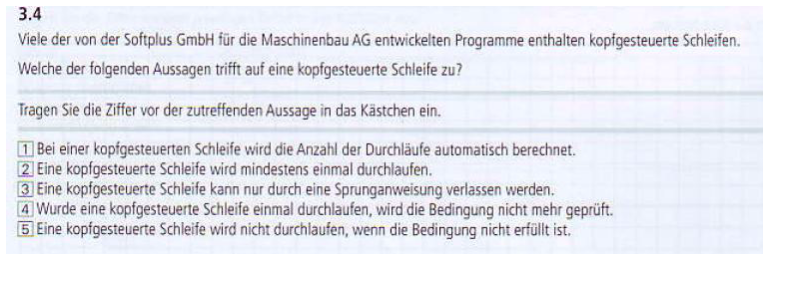
Erstellen Sie ein Struktogramm, das aus einer ganzen, positiven Zahl eine Division vornimmt, die eine positive ganze Zahl zurückgibt und je nach Modulaverfahren einen Restwert ermittelt.

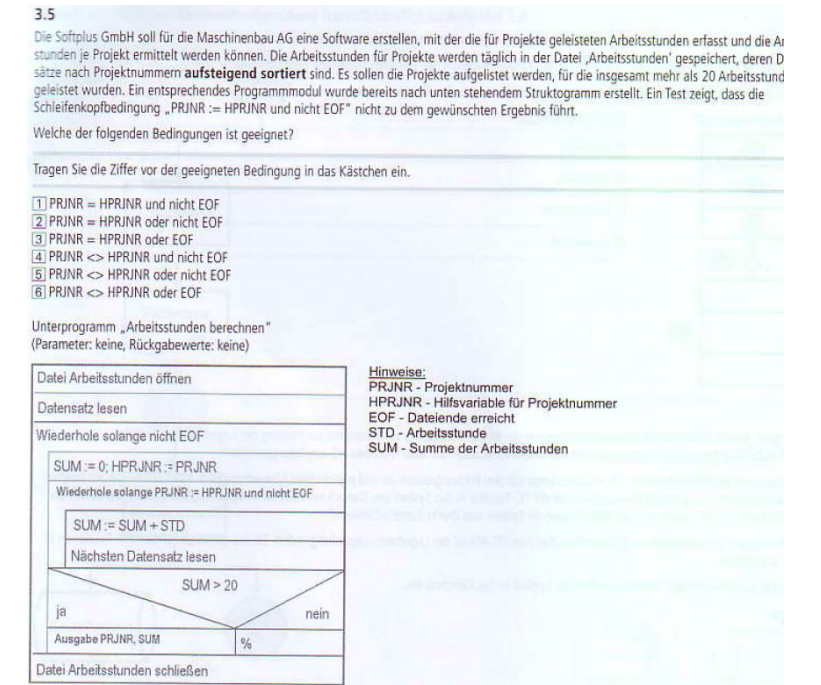
Beispiel 17 Mod 2 ergibt 8 Rest 1, 37 Mod 6 ergibt 6 Rest 1.

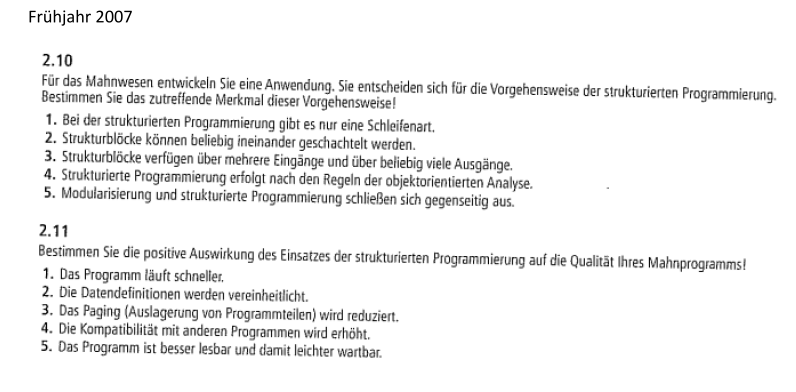
7.











Lösung

