Professor Dr. F. Schweiggert  Abteilung Angewandte Informationsverarbeitung  6. Dezember 2000
Johannes Mayer


# Allg. Informatik für WiWi (WS 2000)

**Abgabetermin: 14. Dezember 2000**

## Beispiellösung

## 1 Wo haben sich die Fehler versteckt? (7 Punkte)

Im Verzeichnis **/www/thales/ftp/pub/vorlesungen/ws00/prog/7** auf dem FTP-Server befinden sich die Dateien *GGT.od* und *GGT.om*. Leider haben sich bei beiden Dateien massiv Fehler eingeschlichen. Versuchen Sie, das Programm GGT zu compilieren, und bessern Sie dann jeweils die Fehler aus, die der Compiler meldet. Wiederholen Sie dies solange, bis das Programm compiliert ist. Erklären Sie Ihrem Tutor, was für Fehler in beiden Dateien vorkommen und wie sie behoben werden können.
**Lösung:**
Verbesserung der Datei *GGT.od*:

- `DEFINITION GGT;` statt `DEFINITION GGT.` in Zeile 1
- `END GGT.` statt `END.` in Zeile 5 (am Schluss muss der Modulname nach dem `END` stehen)

Verbesserung der Datei *GGT.om*:

- `IMPORT Read, Write, Streams;` statt `IMPORT Read; Write Streams;` in Zeile 2 (Liste der importierten Module durch Kommata getrennt)
- `VAR x,y: INTEGER;` statt `VAR x,y,: INT` in Zeile 10 (es gibt keinen Typ `INT` und nach der letzten Variable darf kein Komma stehen)
- `Write.String(...)` statt `String(...)` in Zeile 13 (bei der Verwendung importierter Funktionen muss der Name des Moduls angegeben werden)
- `"Bestimmung des GGT:"` statt `"Bestimmung des GGT:""` in Zeile 13 (String ist nicht ordentlich terminiert)
- `Read.Int(y);` statt `Read.Int(yy);` in Zeile 17 (Variable `yy` existiert nicht)
- `&` statt `AND` in Zeile 19 (das logische Und ist der Operator `&`; `AND` ist kein Schlüsselwort)
- `#` statt `!=` in Zeile 20 (der Ungleich-Operator ist `#`; `!=` ist kein Operator in Oberon)

- `:=` statt `=` in Zeile 22 (Wertzuweisung statt Vergleich)
- `END` statt `ENDIF` in Zeile 26 (jede Schleife wird mit `END` beendet)
- `END` statt `END WHILE` in Zeile 27
- `Write.String("GGT = ");` statt `Write.String("GGT = );` in Zeile 29 (String ist nicht ordentlich terminiert)
- `Write.Int(x, 4);` statt `Int(x, 4, 2)` in Zeile 30 (importierte Funktionen müssen unter Angabe des Modulnames verwendet werden; außerdem muss die Anzahl der Parameter stimmen)
- `Streams.stderr` statt `Streams.stder` in Zeile 33 (Bezeichner `stder` nicht vorhanden im Modul `Streams`; wohl Tippfehler)
- `Write.LnS(Streams.stderr);` statt `LnS(Stream.stderr);` in Zeile 35 (Funktionsaufruf mit Angabe des Moduls; Modul `Stream` wurde nicht importiert - Tippfehler)
- `END` statt `END IF` in Zeile 36
- `END GGT.` statt `END GGT;` in Zeile 37

## 2 Berechnung des KGV (7 Punkte)

Das kleinste gemeinsame Vielfache zweier positiver Zahlen erhält man, wenn man das Produkt der beiden Zahlen durch den ggT teilt. D. h.

$$\mathrm{kgV}(x, y) = x \cdot y / \mathrm{ggT}(x, y)$$

wobei $x$ und $y$ positive ganze Zahlen seien. Seien also beispielsweise $x = 12$ und $y = 15$. Dann ist $\mathrm{ggT}(x, y) = \mathrm{ggT}(12, 15) = 3$ und somit ist $\mathrm{kgV}(x, y) = \mathrm{kgV}(12, 15) = 12 \cdot 15/3 = 12 \cdot 5 = 60$. Schreiben sie nun das Programm GGT, das Sie in der vorigen Aufgabe verbessert haben, so um, dass der KGV berechnet wird. (Falls es Ihnen nicht gelungen ist, alle Fehler in der vorigen Aufgabe zu verbessern, so können sie auch die beiden Dateien aus dem Skript abtippen.)
**Lösung:**
In der Definitionsdatei muss nur der Name GGT in KGV geändert werden.

```
DEFINITION KGV;
(* Bestimmung des kleinsten gemeinsamen
 * Vielfachen zweier positiver ganzer Zahlen
 *)
END KGV.
```

In der Moduldatei muss ebenfalls der Name von GGT in KGV geändert werden. Zudem speichern wir die Eingabe vor der Berechnung des ggT in den Variablen *x0* und *y0*. Nach der Berechnung des ggT berechnen wir den kgV nach der angegebenen Formel.

```
MODULE KGV;
   IMPORT Read, Write, Streams;
   (* Read: Modul mit Funktionen zum Lesen von der
    * Standardeingabe
    * Write: Modul mit Funktionen zum schreiben auf
    * Standardausgabe / Diagnoseausgabe
    * Streams: Definition von stdin, stdout, stderr
    *)

   VAR x,y,x0,y0,ggt,kgv: INTEGER;

BEGIN
   Write.String("Bestimmung des KGV:"); Write.Ln;
   Write.String("Bitte erste pos. ganze Zahl: ");
   Read.Int(x); Write.Ln;
   Write.String("Bitte zweite pos. ganze Zahl: ");
   Read.Int(y); Write.Ln;

   (* urspruengliche Zahlen speichern *)
   x0 := x; y0 := y;

   IF (x>0) & (y>0) THEN

      (* ggT berechnen *)

      WHILE x # y DO
         IF x > y THEN
            x := x-y
         ELSE
            (*hier: y>x*)
            y := y-x
         END
      END;

      (* GGT der urspruenglichen Zahlen ist nun
       * gleich x bzw. y
       *)

      ggt := x; (* gleichbedeutend mit ggt := y; *)

      kgv := x0 * (y0 DIV ggt);

      Write.String("kgV = ");
      Write.Int(kgv, 4);
      Write.Ln;
   ELSE
```

```
      Write.StringS(Streams.stderr,
                    "fehlerhafte Eingabe!!!");
      Write.LnS(Streams.stderr);
   END;
END KGV.
```