Einführung in die komponentenorientierte Programmierung mit Borland-Delphi

Übung 10

Unterprogramme in Objekt-Pascal – Funktionen und Units

Dr. Henry Herper

Otto-von-Guericke-Universität Magdeburg

Institut für Simulation und Graphik

Funktionen

Funktionen sind wie Prozeduren entweder (vordeklarierte) **Standardfunktionen** oder sie sind vom **Programmierer definiert**.

Die Parameterübergabe verhält sich analog zu Prozeduren.

Der Name einer Funktion sollte mit dem Zeichen F beginnen.

Funktionen

Die wesentlichen Unterschiede zu Prozeduren sind:

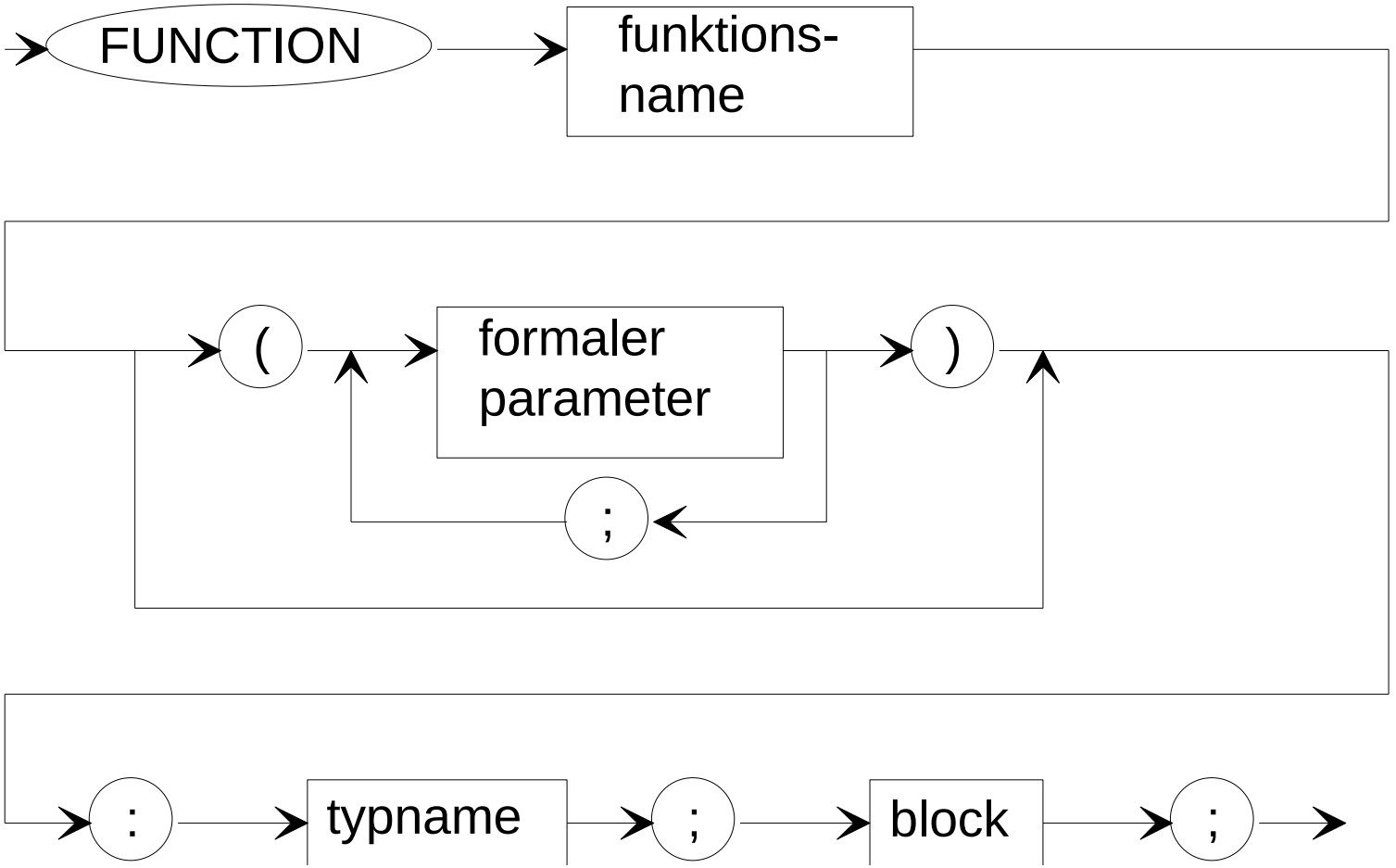
1. Innerhalb der Funktionsdefinition muss ein Typ der Funktion festgelegt werden, der den Ergebnistyp beschreibt. Diese Festlegung ist auch notwendig, wenn der Wert der Funktion nicht angegeben bzw. ausgewertet wird.
2. Dem Funktionsnamen (bzw. der result-Variable) muss ein Wert zugewiesen werden sonst ist der Rückgabewert undefiniert
3. Der Funktionsaufruf ist ein Ausdruck und kann auch in einem Ausdruck verwendet werden. Er besteht auch die Möglichkeit, den Wert der Funktion nicht auszuwerten und den Funktionsaufruf analog zum Prozeduraufruf zu verwenden.

Funktionsvereinbarung

Der Funktionskopf ist mit dem Prozedurkopf äquivalent, außer dass der Funktionskopf mit dem reservierten Wort **function** eröffnet wird und auch den Typ des Ergebnisses mit definieren muss.

Dies wird durch Anfügung eines Doppelpunktes und eines Typs an den Funktionskopf erreicht.

Funktionsvereinbarung



Funktionsvereinbarung

Der Anweisungsteil einer Funktion ist eine Verbundanweisung.

Soll die Funktion einen Rückgabewert liefern, so muss innerhalb des Anweisungsteiles wenigstens eine Ergibt-anweisung auftreten, die dem Funktionsbezeichner einen Wert zuweist.

Es ist üblich, genau eine Anweisung, die letzte Anweisung, der Funktion zu verwenden.

UNIT-Konzept

Zur Unterstützung des Softwarekonzeptes des modularen Programmierens wurde in TURBO-PASCAL (ab Version 4.0) das UNIT-Konzept integriert. Im PASCAL ist eine **UNIT** ist ein getrennt erstelltes, editiertes und kompiliertes Modul, das mit anderen Modulen des Programms nach Bedarf zusammengebunden werden kann. Es wurde verwendet, um Quelltexte erstellen zu können, die über die 64kByte Segmentgröße hinausgehen.

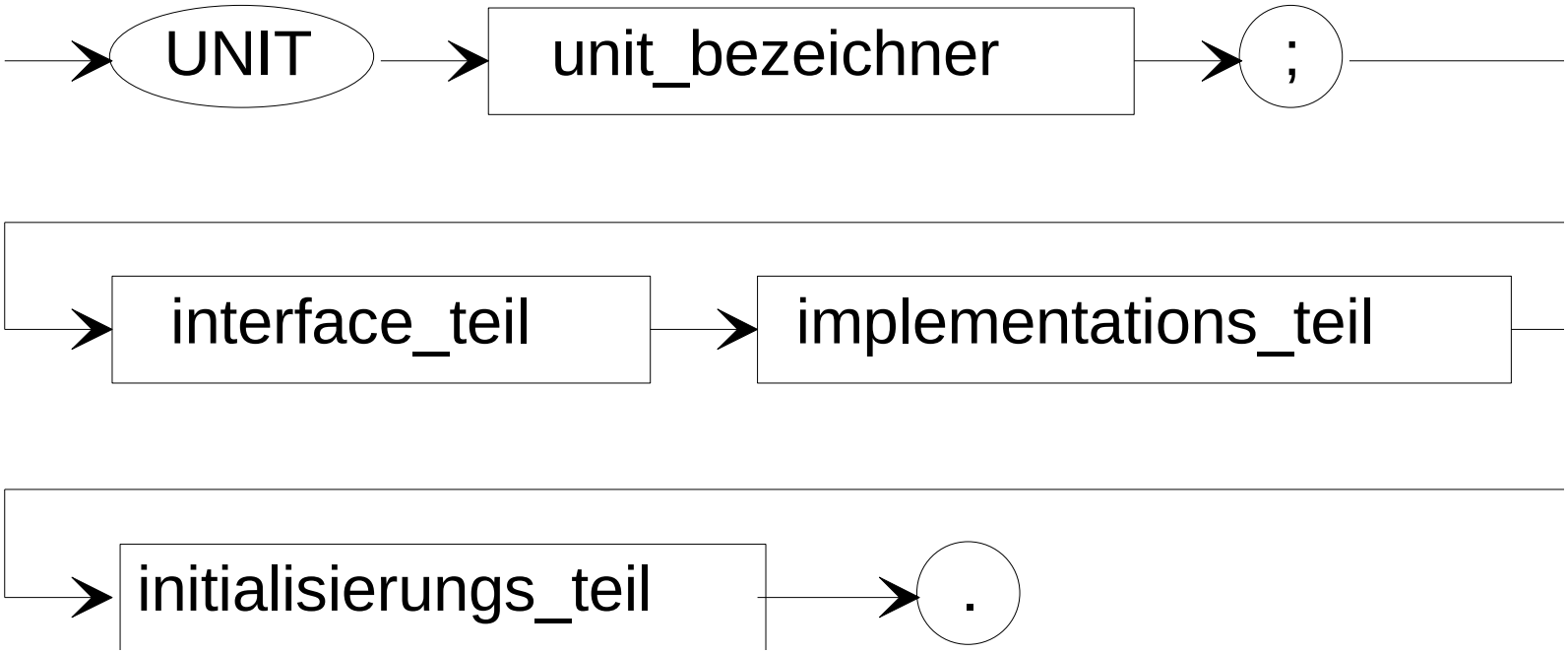
UNIT-Konzept

In Units können andere Units eingebunden werden. Eine **Kreuzverkettung (zirkuläre Referenzen)**, Units rufen sich gegenseitig auf, ist im Interfaceteil **nicht zulässig. Sind zirkuläre Referenzen erforderlich, so müssen die entsprechenden Units im Implementationsteil eingebunden werden.**

Wird eine neue Unit erstellt, so wird folgende Hülle generiert:

```
unit Unit2;
interface
implementation
end.
```

UNIT-Konzept



UNIT-Konzept - Interface

Mit dem Bezeichner **INTERFACE** werden die Deklarationen eingeleitet, die in den Units bzw. Programmen Gültigkeit haben sollen, in die die Unit eingebunden wird. Die so deklarierten

- Kostanten,
- Typen,
- Variablen und
- Prozeduren und Funktionen

sind global für alle Units oder Programme, in die die deklarierte Unit eingebunden wird.

UNIT-Konzept - Implementation

Der **IMPLEMENTATION**-Teil einer UNIT wird als "nichtöffentlicher" Teil bezeichnet. Er enthält die Quelltexte der Prozeduren und Funktionen. Weiterhin können lokale Variablen und Konstanten deklariert werden.
Für Prozeduren und Funktionen, deren Kopf sich im INTERFACE-Teil befindet, braucht der Programmierer im IMPLEMENTATION-Teil keine Parameterliste angeben. Wird eine Parameterliste angegeben, so muss sie mit der im INTERFACE-Teil identisch sein.
Es besteht die Möglichkeit, am Ende des IMPLEMENTATION-Teils einer UNIT Anweisungen anzugeben, die zu Beginn des Programms, das die UNIT ruft, automatisch abgearbeitet werden können.

UNIT-Konzept - Initialisation

Der Initialisierungsteil wird für jede Unit einmal vor dem eigentlichen Programmbeginn ausgeführt. Es ist zu beachten, dass nur auf Elemente zugegriffen werden kann, die zu diesem Zeitpunkt schon erzeugt wurden.
In Delphi-Units wird der Initialisierungsteil mit BEGIN eingeleitet. Alle hier aufgeführten Anweisungen werden vom Programm, in welches die entsprechende UNIT eingebunden ist, beim Start ausgeführt.
In Borland-Delphi wurde ein gesonderter Abschnitt geschaffen, dieser wird mit Initialization eingeleitet. Der Initialisierungsteil steht immer unmittelbar vor dem abschließenden *end* der Unit.

Aufgabe 10.1: Abstandsberechnung

Schreiben Sie eine Object-PASCAL-Funktion zur Bestimmung des Abstandes zweier Punkte im Raum! Erproben Sie diese Funktion an einer geeigneten Delphi-Applikation. Die Funktion ist in einer gesonderten Tool-Unit zu verwalten.

Folgende Daten sind der Funktion zu übergeben (Typ REAL):

- Punkt1 X1,Y1,Z1

- Punkt2 X2,Y2,Z2

Der ermittelte Abstand ist als Funktionswert zurückzugeben.

$$\left|\overrightarrow{P_1P_2}\right| = \sqrt{(X_2-X_1)^2 + (Y_2-Y_1)^2 + (Z_2-Z_1)^2}$$

Aufgabe 10.2: Schaltjahr

Erstellen Sie eine Object-Pascal-Funktion einer Delphi-Applikationzur Feststellung, ob ein einzugebendes Jahr ein Schaltjahr ist. Die Sonderregelungen für Jahrhunderte sind zu beachten. Die Jahreszahl wird als ganzzahliger, positiver Wert übergeben. Als Funktionswert ist wahr (Schaltjahr) oder falsch (kein Schaltjahr) zurückzugeben.

HA 10.1: Körperberechnung

Erstellen Sie eine Unit „koerperberechnung“ zur Unterstützung von Körperberechnungen.

Es sind Funktionen zur Berechnung

- von Volumen und Oberflächeninhalt einer Kugel,
- von Volumen und Oberflächeninhalt eines Kreiszylinders,
- von Volumen, Mantelfläche und Oberflächeninhalt eines Kegelstumpfes

zu erstellen. Die notwendigen Parameter sind zu übergeben.

HA 10.2 – Korrelation

Schreiben Sie eine Delphi-Applikation, die für eine Stichprobe von Wertepaaren das Minimum, das Maximum, den Mittelwert, die Standardabweichung für den x-Wert und den Korrelationskoeffizienten ermittelt.

Die Eingabewerte sind in einem Stringrid zu erfassen. Das Stringgrid enthält maximal 30 Wertepaare.

Als Beispiel ist der Zusammenhang zwischen Körpergröße und Geburtsmonat zu bestimmen.

Kontrollfragen

1. Erläutern Sie die beiden in Object-Pascal verfügbaren Möglichkeiten zur Realisierung von Unterprogrammen. Gehen Sie dabei auf Unterschiede und Gemeinsamkeiten bei der Kommunikation mit dem rufenden Programm ein.
2. Erläutern Sie die Funktionsweise der Parameterübergabe zwischen einem rufenden Programm und einer Prozedur am Beispiel von Object-Pascal. Gehen Sie dabei auf die Gültigkeitsbereiche und Initialisierung der Variablen ein. Welche Anforderungen werden an die Datentypen der Parameter gestellt?
3. Erläutern Sie das Variablenkonzept von Object-Pascal. Gehen Sie dabei auf Gültigkeitsbereiche und Initialisierungen von Variablen ein. Betrachten Sie dabei die Ebenen Projekt, Unit und Prozedur.

Kontrollfragen

4. Erklären Sie die Begriffe „Defaultparameter“ und „untypisierter Parameter“. Welche Besonderheiten sind bei ihrer Verwendung in Unterprogrammen zu beachten?
5. Beschreiben Sie das Konzept zur Dateneingabe und Datenausgabe von Borland Delphi. Was ist bei der Schreibweise und Verarbeitung von Daten zu beachten? Gehen Sie auf die notwendige Typenkonvertierung ein.
6. Erläutern Sie das Unit-Konzept von Objekt-Pascal. Beschreiben Sie den prinzipiellen Aufbau einer Unit. Geben Sie Gültigkeitsbereiche von Variablen und Prozeduren an. Erläutern Sie in diesem Zusammenhang das Prinzip der Forwarddeklaration.