Übungsblock Pascal

Kontrollstrukturen in Pascal:

Bedingte Anweisung:

if *log. Ausdruck* then *Anweisung* else *Anweisung;*

- else-Teil optional
- Anweisung kann wiederum if-Anweisung sein
- mehrere Anweisungen im then- oder else-Teil: Klammerung durch begin…end
- ohne Klammerung: else gehört zum “am nächsten davor stehenden” if, das noch kein else hat

Übungsblock Pascal

```
…
If k<l
    then
        statement1;
        statement2;
…

If k<l
    then
        begin
        statement1;
        statement2
        end
```

nicht äquivalent!

Übungsblock Pascal

```
if bedingung1
        then  if bedingung2
                        then statement1

        else statement2;
```

else-Teil gehört zu innerem if-Teil → einrücken oder

```
if bedingung1
        then begin
                if bedingung2
                then statement1
                else statement2
        end
```

Klammerung macht es lesbarer

Übungsblock Pascal

```
if bedingung1
        then begin
                if bedingung2
                then statement1
            end
        else statement2;
```

Klammerung notwendig, sollte else-Teil erstem if zugeordnet werden!

Übungsblock Pascal

Kontrollstrukturen in Pascal:

Schleifen (1):

repeat *Anweisung* until *Bedingung*

- mindestens eine Ausführung
- mehrere Anweisungen in Schleife: Anweisungen durch ; trennen
- Bedingung: logischer Ausdruck

```
repeat
        wert := wert * zaehler;
        zaehler := zaehler +1
until zaehler = n;
```

Übungsblock Pascal

Kontrollstrukturen in Pascal:

Schleifen (2):

while *Bedingung* do *Anweisung*

- Bedingung: logischer Ausdruck
- Bedingung nicht erfüllt: kein Schleifendurchlauf
- mehrere Anweisungen in Schleife: Anweisungen in begin…end-Block

```
while zaehler <= n do
    begin
        wert := wert * zaehler;
        zaehler := zaehler +1
    end
```

Gleiches Resultat wie bei repeat-Schleife?

Übungsblock Pascal

Kontrollstrukturen in Pascal:

Schleifen (3):

for *var* := startwert to endwert do *Anweisung*

- var: sog. Kontrollvariable, Typ integer, char, boolean oder Aufzählungstyp (succ-Fkt. muss darauf definiert sein)
- mehrere Anweisungen in Schleife: Anweisungen in begin…end-Block
- startwert, endwert können Ausdrücke mit passendem Typ sein
- Automatisches Heraufzählen

```
for zaehler := 1 to n do
        wert := wert * zaehler;
```

Wie sieht entsprechende while-Schleife aus?

Übungsblock Pascal

Kontrollstrukturen in Pascal:

Schleifen (4):

for *var* := startwert downto endwert do *Anweisung*

- Eigenschaften wie eben
- Automatisches Herabzählen (pred), wenn startwert >=endwert

```
for zaehler := n downto 1 do
        wert := wert * zaehler
```

Übungsblock Pascal

Case-Anweisung in Pascal:

```
case    ausdruck  of
    const1 :  Anweisung1;
    const2 :  Anweisung2;
    …
    const_n :  Anweisung_n
end;
```

- Zur Auswahl einer von mehreren Anweisungen in Abhängigkeit vom Wert des Ausdrucks
- const1, …,const_n muss vom Ergebnistyp des Ausdrucks sein
- const_i kann eine oder mehrere durch Komma getrennte Konstanten enthalten
- Anweisung_i kann (durch Klammerung) bel. komplexes Statement sein
- sehr hilfreich bei Aufzählungstypen

Übungsblock Pascal

Beispiel:

```
TYPE    wochentag=(mo,di,mi,do,fr,sa,so);
VAR     tag:wochentag;
…..
case    tag  of
    mo :  writeln(´Montag´);
    di :    writeln(´Dienstag´);
    …
    sa, so :  writeln(´Wochenende!´)
end;
```