Til Mohr, 405959
Andrés Montoya, 405409
Marc Ludevid Wulf, 405401

Aufgabe 4

```
x_3 := x_4 + 0;    // Set x_3 to 0
x_0 := x_4 + 1;    // Set x_0 to 1
// Note 1
LOOP x_2 DO
        x_3 := x_4 + 0;   // Set x_3 to 0
        // Note 2
        LOOP x_1 DO
                LOOP x_0 DO
                        x_3 := x_3 + 1;
                END
        END
        x_0 := x_3 + 0;
END
```

Unser Programm berechnet für $x_1^{x_2}$ x_2-mal $1 \cdot x_1 \cdot ... \cdot x_1$.
Hierzu müssen wir also x_2 mal eine Multiplikation $x_0 := x_0 \cdot x_1$ durchführen (`Note 1`).
Dazu initialisieren wir x_0 auf 1.
Die Multiplikation selber müssen wir nochmal in 2 Schleifen aufteilen (`Note 2`): Wir setzten unsere Hilfsvariable x_3 auf 0, addieren dann $x_1 \cdot x_0$-mal eine 1 auf x_3, und speichern dann das Ergebnis in x_0.

Aufgabe 5

a) Unser Programm benutzt Variablen zusätzlich zu den aus `P` und `Q`, die die Programme selber nicht benutzen. Diese Variablen sind eben x_l, eine Hilfsvariable x_m und x_{k+1}, welche immer 0 ist.

```
x_m := x_(k+1) + 0;  // Set x_m to 0
WHILE x_l = 0 DO
        P;
        x_l := x_(k+1) + 1; // Set x_l to 1
        x_m := x_(k+1) + 1; // Set x_m to 1
END
WHILE x_m = 0 DO
        Q;
        x_l := x_(k+1) + 1; // Set x_l to 1
        x_m := x_(k+1) + 1; // Set x_m to 1
END
```

Unser Programm verwendet die Variable x_m als eine Art Mutex. Haben wir nämlich einmal `P` ausgeführt, können wir die `Q` ausführen.

Til Mohr, 405959
Andrés Montoya, 405409
Marc Ludevid Wulf, 405401

b) Unser Programm benutzt Variablen zusätzlich zu denen aus `P_i`, die die Programme selber nicht benutzen. Diese Variablen sind eben x_{k+1}, eine Hilfsvariable x_m und x_l, welche immer 0 ist.

```
// Initialisierung
x_m := x_l + 0;    // Set x_m to 0

// Fallunterscheidung fuer x_(k+1) = 0
// Fall x_(k+1) = 0?
WHILE x_(k+1) = 0 DO
        P_0;
        x_m := x_l + 1;   // Set x_m to 1
        x_(k+1) := x_l + 1;  // Set x_(k+1) to 1
END
x_(k+1) := x_(k+1) - 1;  // Decrease x_(k+1);


// Fallunterscheidungen fuer x_(k+1) > 0
// Fall x_(k+1) = 1?
WHILE x_(k+1) = 0 DO
        WHILE x_m = 0 DO
                P_1;
                x_m := x_l + 1;  // Set x_m to 1
        END
        x_(k+1) := x_l + 1;  // Set x_(k+1) to 1
END
x_(k+1) := x_(k+1) - 1;  // Decrease x_(k+1);


...

// Fall x_(k+1) = i?
WHILE x_(k+1) = 0 DO
        WHILE x_m = 0 DO
                P_i;
                x_m := x_l + 1;  // Set x_m to 1
        END
        x_(k+1) := x_l + 1;  // Set x_(k+1) to 1
END
x_(k+1) := x_(k+1) - 1;  // Decrease x_(k+1);
```

Unser Programm verwendet die Variable x_m als eine Art Mutex. Haben wir nämlich einmal ein Programm `P_i` ausgeführt, so können wir kein anderes Programm mehr ausführen.

Übungsblatt 07

Til Mohr, 405959
Andrés Montoya, 405409
Marc Ludevid Wulf, 405401

Aufgabe 6