

Potenzberechnung

*(Potenzberechnung)***Stichwörter:** Mehr-Adress-Befehl-Assembler

Potenzberechnung

Erstelle ein rekursives Assemblerprogramm, das seine beiden Parameter über zwei Variablen a und n aus dem Speicher übernimmt und den Wert $\text{power}(a, n)$ berechnet. Das Ergebnis soll in $R0$ liegen. Dabei soll die Rekursion gelten:

$$\text{power}(a, n) = a \cdot \text{power}(a, n-1)$$

Die Lösung der Berechnung soll zum Schluss in $R5$ liegen.

```
-- Potenzberechnung a^n
```

```
-- Erstelle ein rekursives Assemblerprogramm, das seine beiden Parameter ueber zwei
-- Variablen a und n aus dem Speicher uebernimmt und den Wert power(a, n) berechnet.
-- Das Ergebnis soll in R0 liegen. Dabei soll die Rekursion gelten:
-- power(a, n) == a * power(a, n - 1)
-- Die Loesung der Berechnung soll zum Schluss in R5 liegen.
```

```
-- public static int power(int a, int n) {
--     if (n == 0) {
--         return 1;
--     } else {
--         return a * power(a, n - 1);
--     }
-- }
```

```
potenz:
```

```
SEG
```

```
MOVE W I H'0000FFFF', SP
JUMP einstieg
```

```
power:
```

```
PUSHR
MOVE W 64+!SP, R1
-- if (n == 0)
CMP W R1, I 0
JEQ istNull
MOVE W I -1, -!SP
-- n - 1
SUB W I 1, R1, -!SP
CALL power
ADD W I 4, SP
-- a * power(a, n - 1);
MULT W !SP+, R0
JUMP rueckgabe
```

```
-- return 1;
```

```
istNull: MOVE W I 1, R0
```

```

rueckgabe:    MOVE W R0, 68+!SP
              POPR
              RET

einstieg:     MOVE W I -1, -!SP
              MOVE W n, -!SP
              MOVE W a, R0
              CALL power
              ADD W I 4, SP
              -- 3^4 = 81
              MOVE W !SP+, R5
              HALT

-- int a = 3;
a:            DD W 3
-- int n = 4;
n:            DD W 4

-- Tests

-- a:         DD W 3

-- n:         DD W 0 -- 1
-- n:         DD W 1 -- 3
-- n:         DD W 2 -- 9
-- n:         DD W 3 -- 27
-- n:         DD W 4 -- 81
-- n:         DD W 6 -- 729
-- n:         DD W 7 -- 2187
END

public class Power {

    public static int power(int a, int n) {
        if (n == 0) {
            return 1;
        } else {
            return a * power(a, n - 1);
        }
    }

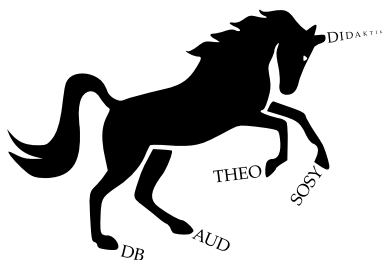
    public static void main(String args[]) {
        int a = 3;
        int n = 4;
        System.out.println(power(a, n));

        System.out.println(power(3, 0)); // 1
        System.out.println(power(3, 1)); // 3
        System.out.println(power(3, 2)); // 9
        System.out.println(power(3, 3)); // 27
        System.out.println(power(3, 4)); // 81
        System.out.println(power(3, 6)); // 729
        System.out.println(power(3, 7)); // 2187
    }
}

```

```
}
```

Code-Beispiel auf Github ansehen: [src/main/java/org/bschlangaul/aufgaben/tech_info/assembler/mehr_adress/Power.java](https://github.com/bschlangaul/aufgaben/tech_info/assembler/mehr_adress/Power.java)



Die Bschlangaul-Sammlung

Hermine Bschlangauland Friends

Eine freie Aufgabensammlung mit Lösungen von Studierenden für Studierende zur Vorbereitung auf die 1. Staatsexamensprüfungen des Lehramts Informatik in Bayern.



Diese Materialsammlung unterliegt den Bestimmungen der Creative Commons Namensnennung-Nicht kommerziell-Share Alike 4.0 International-Lizenz.

Hilf mit! Die Hermine schafft das nicht allein! Das ist ein Community-Projekt! Verbesserungsvorschläge, Fehlerkorrekturen, weitere Lösungen sind herzlich willkommen - egal wie - per Pull-Request oder per E-Mail an hermine.bschlangaul@gmx.net. Der \LaTeX -Quelltext dieses Dokuments kann unter folgender URL aufgerufen werden: https://github.com/bschlangaul-sammlung/examens-aufgaben/blob/main/Module/50_TECH/20_Mehr-Adress/Aufgabe_12-Potenz.tex