# Aufgaben

**Gegeben ist folgendes Programmfragment (für Aufgabe 1-3)**

```
.section    .data
VarA:       .word     30,   65,   0xA,  'A'
VarB:       .byte     33,   -1,   0x10, 0x20, 0x30, 'a',  '1'
```

**Aufgabe 1**
Mit welcher Befehlsfolge (4 Befehle) addiert man die ersten beiden Werte von VarA?

**Aufgabe 2**
Wie lautet das Ergebnis?
VarB - VarA

**Aufgabe 3**
Das .data-Feld VarA beginnet bei Adresse 0x2000. Geben Sie die Speicherinhalte (Hex.) an den folgenden Adressen aus.

| 0x200C | 0x200D | 0x200E | 0x200F | 0x2010 | 0x2011 |
|---|---|---|---|---|---|
| _____ | _____ | _____ | _____ | _____ | _____ |

**Aufgabe 4**
Mit welchem Befehl legt man den Inhalt von Register r0 auf den Stack ab (push)?

**Aufgabe 5**
Mit welchem Befehl springt man zu einem Unterprogramm „MyProg“?

**Aufgabe 6**
Wozu dient das Linkregister?

**Aufgabe 7**
Wozu dient der Framepointer?

**Aufgabe 8**
Geben Sie einen Befehl an, mit dem folgendes berechnet wird:
[r0] <- [r0]+4*[r1].

**Aufgabe 9**
Geben Sie eine Befehlssequenz an, die zum Label „TestOK“ springt, wenn die Bits 0 und 3 in r0 gesetzt sind.

# Lösungen

**Aufgabe 1**

```
ldr     r0, =VarA
ldr     r1, [r0]
ldr     r2, [r0, #4]
add     r3, r1, r2
```

***alternative***

```
ldr r1, = VarA
ldr r0, [r1]
ldr r2, = VarA
ldr r1, [r2]
add r2, r1, r0
```

**Aufgabe 2**

16d = 4*words = 4 * 4 byte

**Aufgabe 3**

| **0x200C** | **0x200D** | **0x200E** | **0x200F** | **0x2010** | **0x2011** |
|---|---|---|---|---|---|
| 0x41 | 0x00 | 0x00 | 0x00 | 0x21 | 0xFF |

**Aufgabe 4**

```
str r0, [sp, #-4]!
```

***alternative***

```
Stmfd sp!, {r0}
```

**Aufgabe 5**

```
bl MyProg
```

*Unterschied B, BL:* `B` Branch | `BL` Branch with link.
bl rettet die Programmadresse nach dem Sprungbefehl (pc) im Linkregister (lr = r14)
Danach wird die Programmausführung an der Unterprogrammadresse fortgesetzt!
**„bx lr“** kopiert die in lr gespeicherte Rücksprungadresse wieder in den pc.

```
……
bl MySubroutine
…..
```

----------->

```
MySubroutine:
stmfd sp!, {r4-r11, lr}
…..
ldmfd sp!, {r4-r11, lr}
bx lr
```

**Aufgabe 6**

Das **Linkregister** dient zur Sicherung der **Rücksprungadresse** beim springen in eine Subroutine.

**Aufgabe 7**

Der **Framepointer** ermöglicht bequemen Zugriff auf den lokalen Speicher (lokale Stack Daten)

**Aufgabe 8**

**add ZIEL, OP1, OP2, (LSL | LSR) #X**

```
add    r0, r0, r1, LSL #2
```

**Aufgabe 9**

```
mov    r1, #00001001b
and    r0, r1
cmp    r0, #00001001b
beq    TestOK
```