#### 1. Aufgabe

Der String Test ist 5 Bytes lang (4 Zeichen + " $\0$ ")

### 2. Aufgabe

```
int:
int getAt(int *a, int i) {
  return *(a+i);
}
short:
short getAt(short *a, int i) {
  return *(a+i);
}
```

Die Berechnung des i-ten Elements eines Arrays a erfolgt "dynamisch", nach Speichertyp, d.h. bei einem double wird bei \*(a+i) für i in 8-Byte Schritten gesprungen, bei int/short-Typen um 4 Byte (je nach Systemarchitektur, vgl. sizeof()).

### 3. Aufgabe

- Zeile 2: bffff844 (Adresse von b)
- Zeile 3: 3ade68b1 (Wert von b als long)
- Zeile 4: 68 (p wurde um 1 im Speicher "verschoben", d.h. zeigt nun auf bffff845)
- Zeile 5: de (p ist nun bffff46)
- Zeile 6: bffff47

## 4. Aufgabe

- Programm 1: Bei increment wurde die Adresse von i übergeben. Beide Variablen haben nach Programmduchlauf den Wert 1338.
- Es wird 1337 zurückgegeben, also j = 1337. Danach wird an der Speicheradresse eins addiert, d.h. i = 1338.

# 5. Aufgabe

Programmstück 1:

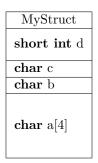
- Zeile 3: Ausgabe: 1 1. Es wird zwei Mal das erste Element von x ausgegeben.
- Zeile 5: Ausgabe: 1 2. Der Pointer px zeigt seit Zeile 4 auf das zweite Element.

Programmstück 2:

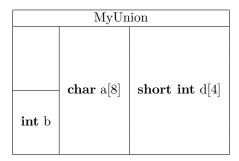
• Zeile 5: Ausgabe: 10 11. Es wird der Speicherbereich an der Adresse vor x überschrieben, welcher vorher nicht für das Programm reserviert wurde. Falls dort andere Variablen oder geschützter Speicher ist, wird versucht, diesen zu überschreiben, was unter Umständen zu Fehlern führen kann.

### 6. Aufgabe

- $\bullet$  Zeile 14: 8: Alle Elemente zusammengezählt (4 + 1 + 1 + 2) ergibt eine Speicherbelegung von 8 Byte.
- Zeile 15: 8: Das grösste Element der Union ist der char, welcher 8 Byte belegt.



MyStruct: Die Elemente werden nacheinander im Speicher platziert. Die Gesamtgrösse setzt sich aus den addierten Grössen der einzelnen Variablen zusammen.



MyUnion: Die Elemente belegen den gleichen Speicherplatz. Das grösste Element bestimmt den Speicherplatz der union.

# 7. Aufgabe

Mit #define wird eine vorgegebene Zeichenfolge durch eine andere ersetzt. Hier wird callA mit callB(1) ersetzt, was zum Aufruf der Funktion callB führt, mit Parameter 1. Dies hat den Output (1 + 2 =) 3 zur Folge.