

## U2 2. Übung

### U2-1 Aufgabe 2

- Daten in char-Feld einlesen

```
wort1
w2
langeswort
a
bb1
Berg
Hamburg
```

```
char feld[10000];
```

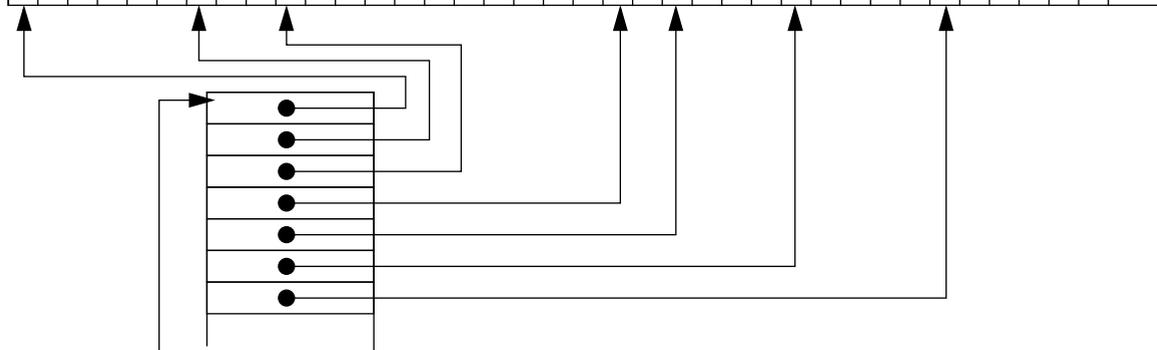
w	o	r	t	1	\n	w	2	\n	l	a	n	g	e	s	w	o	r	t	\n	a	\n	b	b	1	\n	B	e	r	g	\n	H	a	m	b	u	r	...
---	---	---	---	---	----	---	---	----	---	---	---	---	---	---	---	---	---	---	----	---	----	---	---	---	----	---	---	---	---	----	---	---	---	---	---	---	-----

↑  
feld

### U2-1 Aufgabe 2 (2)

- Zeigerfeld allokkieren
  - ◆ beim Einlesen Worte mitzählen
  - ◆ Zeigerfeld in der benötigten Größe mit malloc besorgen
  - ◆ eingelesene Worte durch \0-Zeichen terminieren
  - ◆ Zeiger im Zeigerfeld auf Wortanfänge setzen (→ jeder Zeiger  $\hat{=}$  einem Wort)

w	o	r	t	1	\0	w	2	\0	l	a	n	g	e	s	w	o	r	t	\0	a	\0	b	b	1	\0	B	e	r	g	\0	H	a	m	b	u	r	...
---	---	---	---	---	----	---	---	----	---	---	---	---	---	---	---	---	---	---	----	---	----	---	---	---	----	---	---	---	---	----	---	---	---	---	---	---	-----



```
char **woerter = (char **) malloc ...
```

## U2-1 Aufgabe 2 (3)

- Wörter sortieren
  - ◆ Funktion qsort aufrufen
  - ◆ Schnittstelle aus `stdlib.h`:

```
void qsort(void *base,
           size_t nel,
           size_t width,
           int (*compare) (const void *, const void *));
```

- ◆ Bedeutung der Parameter:
  - **base** : Zeiger auf das erste Element des Feldes, dessen Elemente sortiert werden sollen (*Zeiger auf erstes Wort*)
  - **nel** : Anzahl der Elemente im zu sortierenden Feld (*Zahl der Wörter*)
  - **width**: Größe eines Elements (*Größe eines char-Zeigers - sizeof!*)
  - **compare**: Zeiger auf Vergleichsfunktion

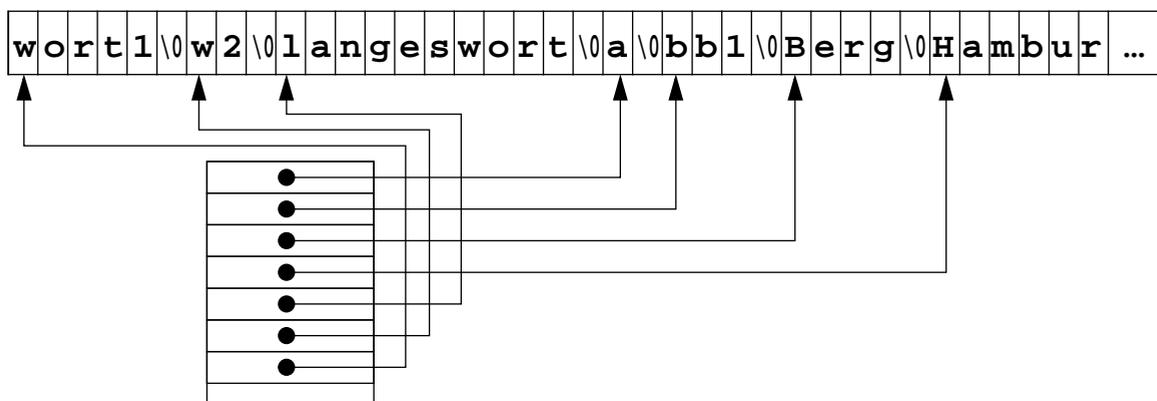
- **ACHTUNG**: nicht die Buchstaben, sondern die Wörter sortieren!

## U2-1 Aufgabe 2 (4)

- Lösung für die compare-Funktion:

```
int compare(const void *a, const void *b) {
    return strcmp(*(char **)a, *(char **)b);
}
```

- Resultat



- die Buchstaben stehen unverändert, nur die Zeiger wurden sortiert!