U4 4. Übung

U4-1 Aufgabe 3

- Teilaufgabe a)
  - ➤ Argumente aus der Kommandozeile ausgeben
  - ➤ Vorlesung F.36 ff
- Teilaufgabe b)
  - ➤ Directory öffnen (*opendir(3)*, Vorlesung G.8)
  - ➤ Schleife: Einträge lesen (*readdir(3)*, Vorlesung G.9)
  - ➤ Dateinamen ausgeben (*printf(3)*, `d_name` aus `dirent`-Struktur)

U4-2 Aufgabe 3c

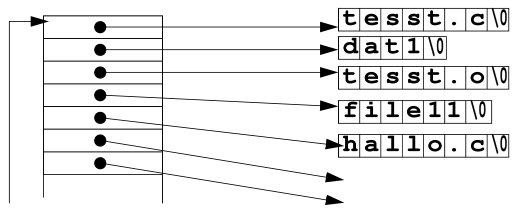
- Problem
  - ➤ Rückgabewert von readdir nur bis zum nächsten Aufruf von readdir gültig
  - ➤ Daten müssen aus dirent-Struktur wegkopiert werden
- Lösung
  - ➤ Speicher für Dateinamen mit *malloc(3)* besorgen
  - ➤ benötigten Speicherplatz mir *strlen(3)* ermitteln
  - ➤ Platz für abschließendes `'\0'`-Zeichen nicht vergessen!
  - ➤ Dateiname in neuen Speicher umkopieren (*strcpy(3)*)
- Problem 2
  - ➤ Einträge in einem Directory sind nicht sortiert
- Lösung
  - ➤ Feld mit Zeigern auf die Dateinamen mit der Funktion *qsort(3)* sortieren

U4-2 Aufgabe 3c (2)

- Zeigerfeld anlegen
  - ◆ Annahme: maximal 1000 Einträge in einem Directory (Aufgabenstellung!)
    - ➤ Feld fester Größe möglich
```
char dateinamen[1000];
```
  - ◆ funktioniert nicht, wenn Feld vergrößert werden soll!
    - ➤ Feld dynamisch allokieren
```
int eintraege = 1000;
char **dateinamen = malloc (eintraege * sizeof(char *));
```
    - ➤ und wenn es dann nicht groß genug ist? — Feld vergößern!
```
eintraege += 1000;
dateinamen = realloc(dateinamen, eintraege * sizeof(char *));
```


U4-2 Aufgabe 3c (3)

- ◆ Zeiger im Zeigerfeld auf Dateinamen setzen
  (➔ jeder Zeiger $\hat{=}$ einem Dateinamen)




```
char **dateinamen = malloc(...
```

U4-2 Aufgabe 3c (4)

- Dateinamen (=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 *));
```

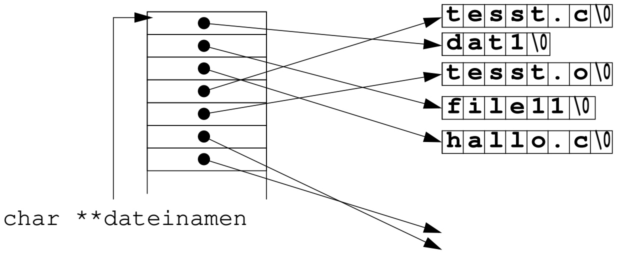
  - Bedeutung der Parameter:
    - **base** : Zeiger auf das erste Element des Feldes, dessen Elemente sortiert werden sollen *(Zeiger auf erstes Wort)*
    - **nel** : Anzahl der Elemente im zu sortierenden Feld *(Zahl der Wörter)*
    - **width**: Größe eines Elements *(Größe eines char-Zeigers - sizeof!)*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

U4-2 Aufgabe 2 (4)

- Lösung für die compare-Funktion:

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

- Resultat



  - die Wörter (Dateinamen) stehen unverändert, nur die Zeiger auf die Wörter wurden sortiert!

U4-2 Aufgabe 2 (4)

- nach dem Sortieren
  - mit `printf("%s\n", ...)` alle Einträge des Feldes ausgebenfc