

# Erste Übung im Sommersemester 2021

Nadine Schärmann & Thomas Klütz

PR10

19. April 2021

1. Organisatorisches
  - HA und Übungen
  - Ausblick über Themen
2. Hausaufgabe - Vier Gewinnt
  - Felder als UP-Parameter
3. Pointer - Zeiger
  - Was sind Pointer?
  - Erstes Handwerkszeug
    - Deklaration
    - NULLIFY und NULL
    - ASSOCIATED
4. Hausaufgabe - Kinderreim

# Organisatorisches

- Übung: Montag 3.DS über BBB
- Sonderübung: Dienstag 4.DS über BBB
- DRINGEND in Übungsgruppen eintragen für die Korrektur!
- Hausaufgaben: Abgabe in 2er-Teams im OPAL
- Es werden 50% der HA-Punkte benötigt für die Zulassung zur Prüfung.
- Fragen zu Vorlesung und HA bitte im öffentlichen Matrix-Chatraum stellen!
- Folien und Programme der Übung / Sonderübung findet ihr im gitlab.

# Themen im kommenden Semester

- **Pointer bzw. Zeiger**
- ADTs mit Pointern
- Algorithmen, insbesondere Sortieralgorithmen
- rekursive und iterative Algorithmen
- ...

# HA - Vier Gewinn

- Hinweise beachten:
  - Feld für den Füllstand
  - Funktion für den Gewinn-Test
- Häufige Fehlerquelle: Felder in Unterprogrammen
- Gab es sonst noch Probleme?

# Felder in Unterprogrammen

## Felder als UP-Parameter

- statisches Feld oder
- Feld übernommener Gestalt: Gestalt zum Aufrufzeitpunkt des UPs durch a. A. festgelegt

```
FUNCTION mult (A, v)
  INTEGER, DIMENSION ( : , : ) :: A
  INTEGER; DIMENSION (0 : ) :: v
  ...
END FUNCTION
```

Nicht mit ALLOCATABLE arbeiten!

# Was sind Pointer?

Ein Pointer ist eine Variable, die keinen Wert sondern eine Speicheradresse zwischenspeichert.

Der Zeiger *referenziert* (verweist, zeigt auf) einen Ort im Hauptspeicher des Computers. Hier können Variablen oder andere Zeiger gespeichert sein.

Pointer dienen unter anderem der dynamischen Speicherverwaltung.

Pointer können auf folgende Objekte zeigen:

- andere Pointer
- allokierte Objekte (z.B. dynamische Arrays)
- Variablen mit Target-Attribut

# Erstes Handwerkszeug - Deklaration

## Deklaration:

### ... das **POINTER-Attribut**:

```
INTEGER, POINTER :: p, q
```

### ... das **TARGET-Attribut**:

```
INTEGER, TARGET :: a, b
```

```
a = 3
```

```
p => a
```

## Pointerzuweisung mit =>



# Erstes Handwerkszeug - NULLIFY und NULL

Der Nullzeiger ist ein spezieller Wert.

Dieser Wert gibt an, dass die Variable (der Pointer mit diesem Wert) auf „nichts“ zeigt.

## **NULLIFY und NULL:**

```
INTEGER, POINTER :: p => NULL()
```

```
INTEGER, POINTER :: p  
NULLIFY(p)
```

# Erstes Handwerkszeug - ASSOCIATED

## ASSOCIATED:

```
IF (ASSOCIATED(p)) THEN  
  do something ...
```

Seien p1 und p2 Pointer und t eine Variable mit Targetattribut.

- ASSOCIATED(p1) ist .TRUE., wenn p auf ein Objekt zeigt, es ist .FALSE., falls das Pointerargument NULL ist.
- ASSOCIATED(p1,p2) ist .TRUE., wenn beide Pointer auf dasselbe Objekt zeigen, .FALSE. sonst.
- ASSOCIATED(p1,t) ist .TRUE., wenn p1 auf t zeigt, .FALSE. sonst.

! Sind p1 oder p2 in einem nicht definierten Zustand, ist der Rückgabewert von ASSOCIATED zufällig.

# Und jetzt das Ganze noch etwas anschaulicher:

# Und jetzt das Ganze noch etwas anschaulicher:

# HA - Kinderreim

- Abgabefrist: 02.05.2021
- Beachtet die Dateien kreim.dat und KREIM.DAT im OPAL.

Eine Gruppe von Kindern steht im Kreis. Mit dem Abzählreim „Eene, meene, muh und 'raus bist du ! – 'raus bist du noch lange nicht, sag mir erst, wie alt du bist !“ wird bis zu einem Kind „abgezählt“ (jede der 21 Silben entspricht einem Abzählschritt) und dann noch so viele Schritte weiter, wie dieses Kind (an Jahren) alt ist. Das dadurch ausgewählte Kind scheidet aus. Mit dem nachfolgenden Kind wird das nächste Abzählen begonnen.

Dieser Vorgang wird so oft wiederholt, bis nur noch ein Kind im Kreis übrig bleibt. Dieses hat gewonnen.

# HA - Wiederholung

Öffnen von Dateien:

```
OPEN([UNIT=]u [,IOSTAT=iostat] [,ERR=err][,FILE=file][,STATUS=status]  
[,ACTION=action])
```

Schließen von Dateien:

```
CLOSE([UNIT=]u [,IOSTAT=iostat] [,ERR=err][,FILE=file][,STATUS=status])
```

Lesen aus geöffneten Dateien:

```
READ([UNIT=]u,[FMT=]f [,IOSTAT=iostat] [,ERR=err][,ADVANCE=ad])
```

Schreiben in geöffnete Dateien:

```
WRITE([UNIT=]u,[FMT=]f [,IOSTAT=iostat] [,ERR=err][,ADVANCE=ad])
```