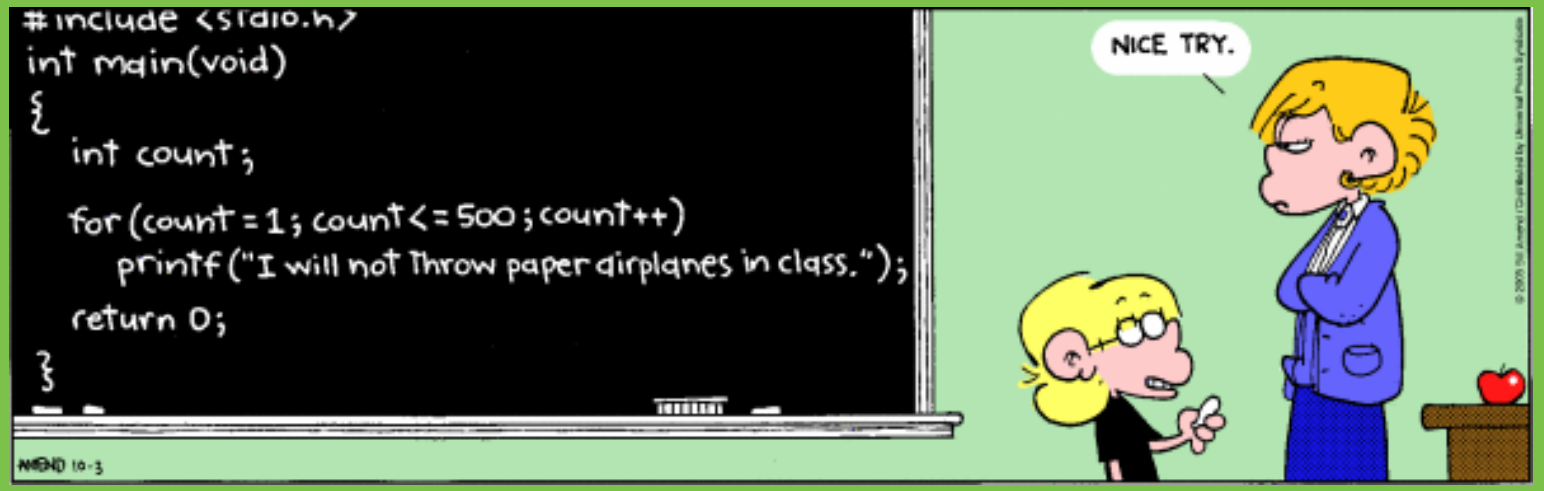
Hochschule Fulda
University of Applied Sciences

Programmierung 1 – Wiederholungsstrukturen



Zählschleifen (`for`-Schleife)

Hochschule Fulda
University of Applied Sciences

```
for (int count=1; count<=500; count++) {
    printf("I will not throw paper airplanes in class.\n");
}
```

- Schlüsselwort **`for`**
- Danach in Klammern drei Bestandteile, je durch Semikolon getrennt
  - Initialisierung (wird zuerst ausgeführt), hier z.B.: `int count=1`
  - Abbruchbedingung (hier Ende, wenn `count>500`): tue etwas, solange `count<=500`
  - Schritt (wird direkt nach jedem Durchlauf ausgeführt), hier: `count++`
- Die Variable *count* heißt Zähler oder auch Schleifenvariable
- Danach folgt ein Block **{ … }**, der sogenannte Schleifenrumpf
  - Bei Blöcken (auch einzeilige) für bessere Lesbarkeit Einrücken nicht vergessen
  - Wenn obige Bedingung nicht mehr erfüllt ist, mache nach dem Rumpf weiter!

Zählschleifen (`for`-Schleife)

Hochschule Fulda
University of Applied Sciences

```c
int akku = 0;
for (int i=2; i<=100; i+=2) {
    akku += i;
}
printf("Ergebnis: %d\n", akku);
```

- Typische Anwendung: Akkumulieren (Ansammeln) von Werten
- Verwende eine sog. Akkumulatorvariable zum Sammeln (*akku*)
- Beispiele
  - Addiere alle geraden Zahlen von 2 bis 100
    - Achtung: Initialisiere Zählvariable *i* mit 2 und nimm Schrittweite 2!
  - Addiere alle Quadratzahlen von 1^2 bis 20^2

Zählschleifen (`for`-Schleife)

Hochschule Fulda
University of Applied Sciences

```c
for (int i = 1; i <= 1000; i++) {
    if (i % 12 == 0 && i % 27 == 0 && i % 44 == 0) {
        printf("Gefunden: %d\n", i);
    }
}
```

- Typische Anwendung: Aus einer Reihe von Zahlen eine oder mehrere mit einer bestimmten Eigenschaft herausfinden
- Häufig ist im Schleifenrumpf eine `if`-Anweisung, die auf die gesuchte Eigenschaft prüft
- Beispiele
  - Kleinste Zahl, die durch 12, 27 und 44 teilbar ist
  - Lösung der Gleichung $x^2 - 34x + 289 = 0$ finden

Übung 1

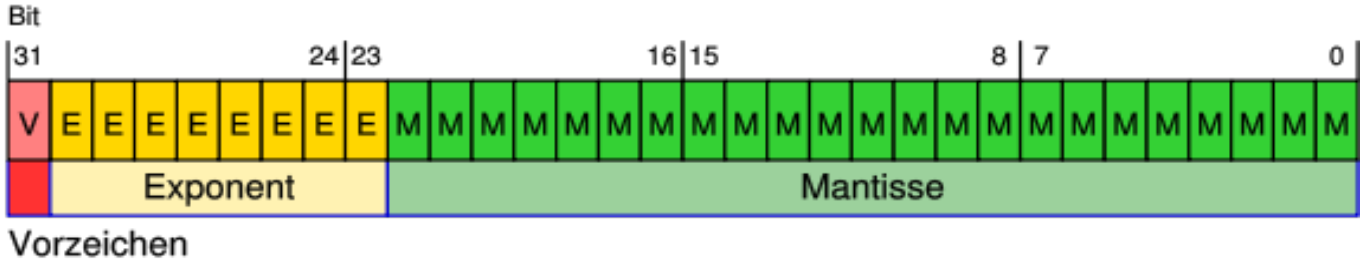
Hochschule Fulda
University of Applied Sciences

- Lesen Sie einen Integerwert *x* ein, berechnen Sie die Teiler von *x* und geben Sie dann die Summe aller Teiler von *x* aus
- Finden Sie die Lösung(en) der Gleichung $x^2 - 34x + 289 = 0$
  - Hinweis: die Lösung(en) liegt/liegen zwischen 1 und 50
  - Aber Achtung, meistens haben Gleichungen keine ganzzahligen Lösungen…
- Geben Sie folgendes Muster auf der Konsole aus (Breite 20, Höhe 30)

Datentyp "float"

Hochschule Fulda
University of Applied Sciences

- Für Gleitpunktzahlen gibt es den Datentyp `float`
  - Die Größe eines `float`'s beträgt 4 Byte
    - Wertebereich des Datentyps geht grob von $\pm 1.2 * 10^{-38}$ bis $\pm 3.4 * 10^{+38}$
  - Interne Darstellung basiert auf Zerteilung in Vorzeichen, Mantisse u. Exponent



- Für Ein- und Ausgabe von Floating-Point-Zahlen gibt es Platzhalter **%f**
  - Beispiel:

```
float pi = 3.1415926f, eps = 1E-6f;
printf("Pi=%f, Epsilon=%f\n", pi, eps);
```

Kopfgesteuerte Schleifen

Hochschule Fulda
University of Applied Sciences

```c
int i = 0;
while (i < 20) {
    printf("%d\n", i);
    i = i + 2;
}
```

- Schlüsselwort `while`
- Dann folgt (in Klammern) die Abbruchbedingung (logischer Ausdruck)
- Danach folgt ein Block { … }, der sog. Schleifenrumpf
  - Geschweifte Klammern und Einrücken nicht vergessen
  - Wenn Bedingung erfüllt ist, wird Rumpf ausgeführt und Abbruchbedingung erneut ausgewertet

Verwendung

Hochschule Fulda
University of Applied Sciences

- Kopfgesteuerte Schleifen und Zählschleifen sind gleich ausdrucksstark
- `while`-Schleifen verwendet man, wenn die Anzahl der Durchläufe (sog. Iterationen) nicht im Voraus bekannt ist
  - Bsp.: Gebe solange Zahlen ein, bis 0 eingegeben wird; berechne dann Summe
- Achtung: bei allen Schleifenarten kann es passieren, dass Rumpf niemals ausgeführt wird oder aber endlos
  - Letzteres heißt Endlosschleife und ist i.d.R. ein Fehler
    - Bsp.:

```
while (true) {
    /* Endlosschleife */
}
```

Fußgesteuerte Schleifen

Hochschule Fulda
University of Applied Sciences

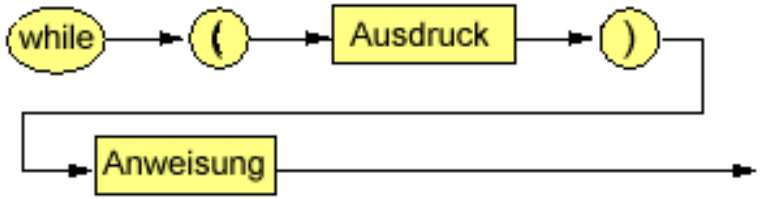
```c
int i = 0;
do {
    printf("%d\n", i);
    i = i + 2;
}
while (i < 20);
```

- Schlüsselwort `do`
- Danach folgt ein Block { … }, der sog. Schleifenrumpf
  - Geschweifte Klammern und Einrücken nicht vergessen
- Dann folgt Schlüsselwort `while` und die Abbruchbedingung
  - Im Unterschied zur `while`-Schleife wird Bedingung erst am Ende geprüft
  - Deshalb wird `do-while`-Schleife immer mindestens einmal durchlaufen

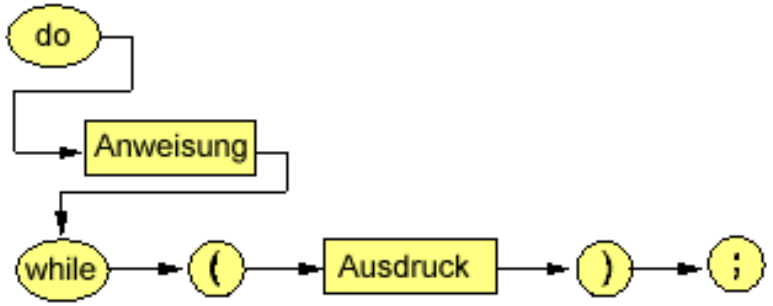
Übung 2

Hochschule Fulda
University of Applied Sciences

- Schreiben Sie ein Programm, das eine Integervariable *x* einliest und die Quersumme von *x* mit Hilfe einer `while` Schleife berechnet.



- Schreiben Sie das Programm so um, dass eine `do-while` Schleife verwendet wird. Was sind Vor- und Nachteile?



Weitere Kontrollmöglichkeiten

Hochschule Fulda
University of Applied Sciences

- Schleifen abbrechen mit `break`
  - Anweisung `break` beendet Schleife sofort, Programm wird nach Schleife weiter ausgeführt
  - Typischer Einsatz: unbestimmtes Warten auf Ereignisse (ungültige Eingaben, spezielle Werte usw.)

```
while (true) {
    /* some code */
    if (found)
        break;
    /* more code */
}
```

- Einzelne Iterationen abbrechen mit `continue`
  - Anweisung `continue` erzwingt direkt nächsten Schleifendurchlauf
  - Typischer Einsatz: man will bestimmte Werte in der Schleife "überspringen"
- Funktioniert beides bei Zählschleifen, kopf- und fußgesteuerten Schleifen
- Vorsicht, Verwendung kann Lesbarkeit von Programmen beeinträchtigen ☹

Fazit: Kontrollstrukturen

Hochschule Fulda
University of Applied Sciences

- Fallunterscheidungen
  - if (*Bedingung_erfuellt*) { ... }
    else if (*andere_Bedingung*) { ... }
    ...
    else { ... }
  - switch(*Ausdruck*) { //char *oder* int
    case *Const_Term_1*: ... break;
    case *Const_Term_2*: ... break;
    ...
    default: ...
    }
- Schleifen
  - while (*Bedingung_erfuellt*)
    { ... }
  - do
    { ... }
    while (*Bedingung_erfuellt*);
  - for (*Init.*; *Bedingung*; *Reinitialisierung*)
    { ... }
  - Zusätzliche Kontrollmöglichkeiten
    - Mit break und continue

Vielen Dank!

Noch Fragen?