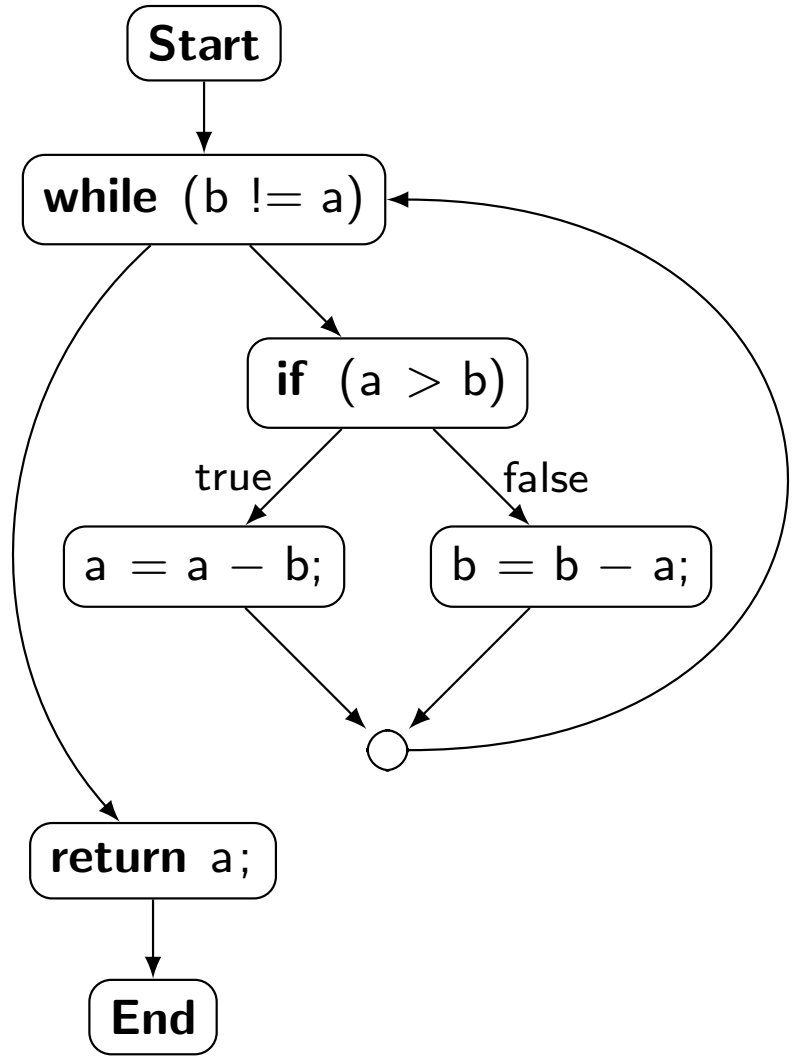
# Dominanz

# Wiederholung: Ablaufgraph (CFG)

- Jeder Grundblock besitzt einen entsprechenden Knoten im Ablaufgraph.
- Die Kanten zwischen den Knoten repräsentieren mögliche Sprünge zwischen den Grundblöcken.
- Es existieren 2 zusätzliche Knoten **Start** und **End** (auch Startblock und Endblock genannt).
- Es gibt eine Kante von **Start** zu jedem Grundblock mit dem das Programm betreten werden kann.
- Es gibt eine Kante von jedem Grundblock mit dem das Programm verlassen werden kann zu **End**.

# Beispiel Ablaufgraph

```
int gcd(int a, int b)
{
    while(b != a) {
        if (a > b)
            a = a - b;
        else
            b = b - a;
    }
    return a;
}
```



# Korrektheitsbedingung Ablaufgraph

$\forall$ Eingabe $z$: Programm durchläuft bei Eingabe $z$ dynamisch die Folge von Grundblöcken

$$W = (\textbf{Start}, b_1, b_2, \ldots, b_n, \textbf{End})$$

$$\Longrightarrow \; W \text{ ist ein Pfad im CFG.}$$

Beachte: Die umgekehrte Implikation muss nicht gelten. Dh CFG darf zuviele Pfade enthalten (zB toter Code), aber niemals zuwenig! CFG ist *konservative Approximation*. Natürlich will man, dass CFG möglichst wenig nicht dynamisch realisierbare Pfade enthält (*Präzision*). CFGs für strukturierten Steuerfluss (nur 1 Ein-Ausgang je Konstrukt, zB IF, WHILE, ...) sind i.a. präzise.

# Dominanz

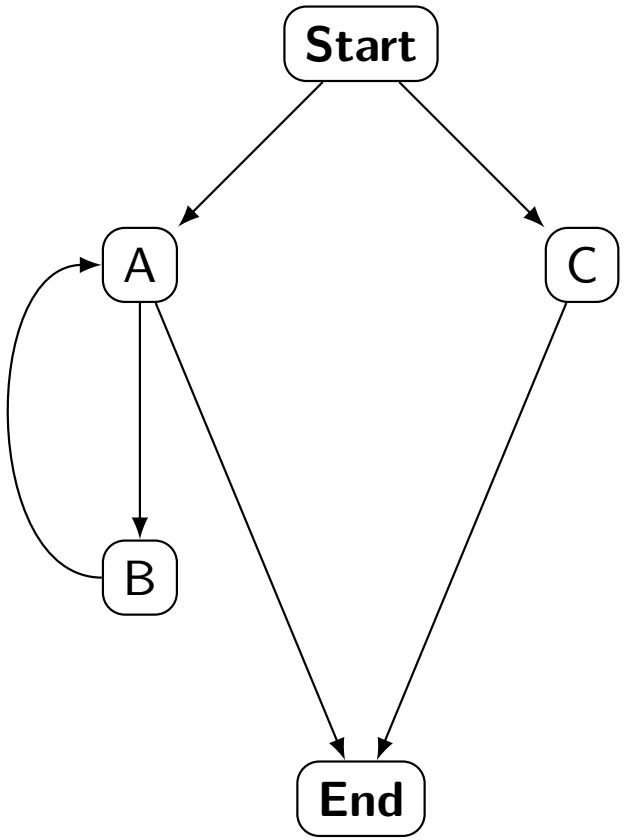
## Definition Dominanz

$X$ *dominiert* $Y$, wenn jeder Pfad im Ablaufgraph von **Start** zu $Y$ auch $X$ enthält. Wir schreiben $X \preceq Y$ (oft auch $X$ dom $Y$)

## Definition Postdominanz

$Y$ *postdominiert* $X$, wenn jeder Pfad im Ablaufgraph von $X$ zu **End** auch $Y$ enthält. Wir schreiben $Y$ postdom $X$.

# Dominanz Beispiel




- **Start** $\preceq B$, $A \preceq B$
- $\neg(A \preceq C)$, $\neg(A \preceq$ **End**$)$
- **End** postdom $B$,
  **End** postdom $C$,
  $A$ postdom $B$

# Dominanz Anwendungen

- Schleifeneintrittspunkt $\preceq$ Schleifenrumpf
- Schleifenaustrittspunkt postdom Schleifenrumpf
- $X \preceq Y \wedge Y$ postdom $X \implies X, Y$ werden stets zusammen ausgeführt
- Steuerflussregion: Bereich zwischen $X$ und $Y$
- strukturierte Programme: Konstrukte (z.B. Schleifen) haben genau einen Eintritts- und einen Austrittspunkt
- Dominatorbaum: liefert Schachtelung von Schleifen, If-Anweisungen, u.ä. $\leftrightarrow$ "*Intervallanalyse*" [Tarjan]

# Dominanz Eigenschaften

- Dominanz ist reflexiv: $X \preceq X$
- Dominanz ist transitiv: $X \preceq Y \wedge Y \preceq Z \Rightarrow X \preceq Z$.
- *strikte Dominanz*: $X \prec Y := X \preceq Y \wedge X \neq Y$.
- *direkte Dominanz* (englisch immediate dominator):

$$X = \text{idom}(Y) := X \prec Y \wedge \neg\exists Z : X \prec Z \prec Y$$

- Jeder Block außer dem Startblock hat genau einen direkten Dominator $\Rightarrow$ direkte Dominatoren bilden einen Baum, den *Dominatorbaum*.
  Dominatorbaum ist spezieller aufspannender Baum.
- Postdominanz ist ebenfalls reflexiv und transitiv. Definition von *strikter Postdominanz*, *direkter Postdominanz* und dem *Postdominanzbaum* analog.

# GOTO-Elimination

Satz(Böhm, Jacopini 1964): Jedes Programm kann nur durch While / If-Then-Else / Begin-End dargestellt werden. [1]
$\Rightarrow$ Ersatz von GOTO durch strukturierte Schleifen.

Die so entstehenden Programme sind jedoch softwaretechnisch pervers

[1] Beweisidee: Durchnummerieren der Grundblöcke im CFG, "Program Counter"

# Beispiel

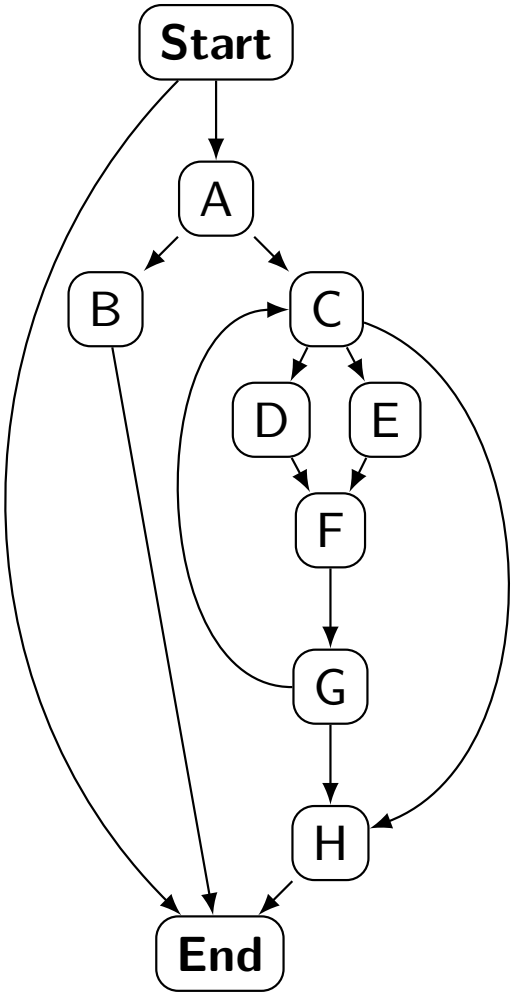



Abbildung: Ablaufgraph

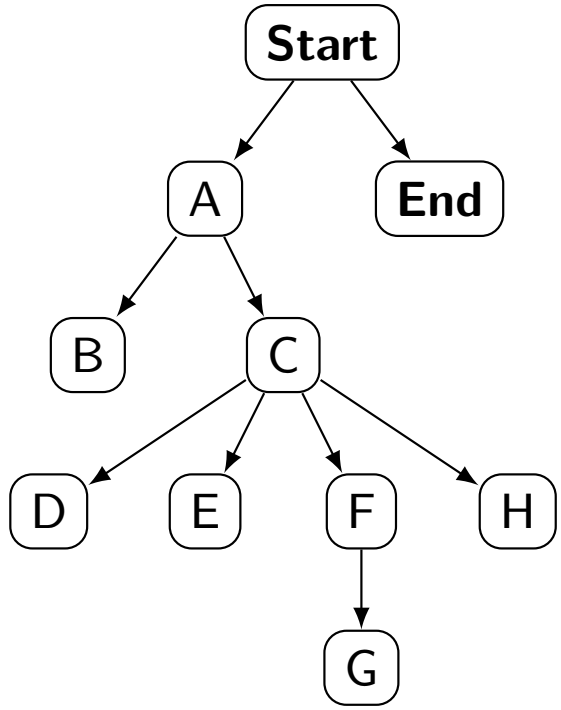



Abbildung: Dominanzbaum

# Berechnung mit Fixpunktiteration

Die Menge aller Dominatoren von $X$ lässt sich darstellen als

$$\mathrm{dom}(X) = \{X\} \cup \bigcap_{Y \in \mathrm{pred}(X)} \mathrm{dom}(Y)$$

$\Longrightarrow$ Berechnung mit Fixpunktiteration möglich (vgl Kapitel Datenflussanalyse). Worst-Case Laufzeit quadratisch bis kubisch.

Zugrundeliegender Verband: Potenzmengenverband der Grundblöcke (oder Anweisungsnummern $x_1, x_2, \ldots, x_n$)
Mit $z_i = dom(x_i)$ ist

$$f_i(z_1, z_2, \ldots, z_n) = \{x_i\} \cup \bigcap_{x_j \in \mathrm{pred}(x_i)} z_j$$

wobei $f_i$ nicht von allen $z_j$ abhängt, sondern nur von Vorgängern von $x_i$
$f_i$ sind monoton (wieso?), $F = (f_1, f_2, \ldots, f_n)$ hat deshalb Fixpunkt