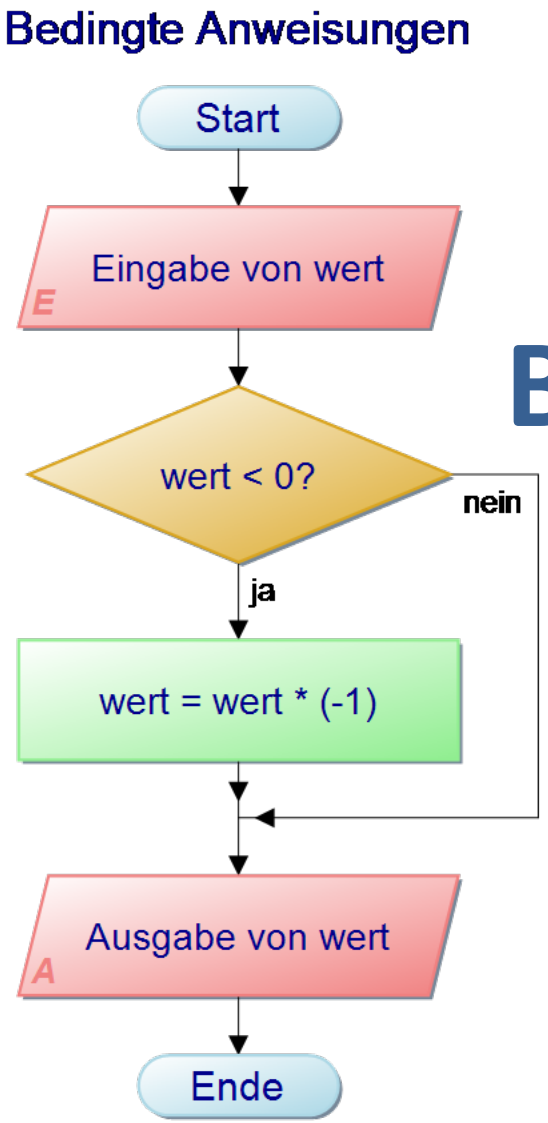


Bedingte Anweisungen in C

Wie berechnet man den Betrag einer Zahl?……

***wenn** sie kleiner als 1 ist ….*

***wenn** sie größer als 1 ist ….*

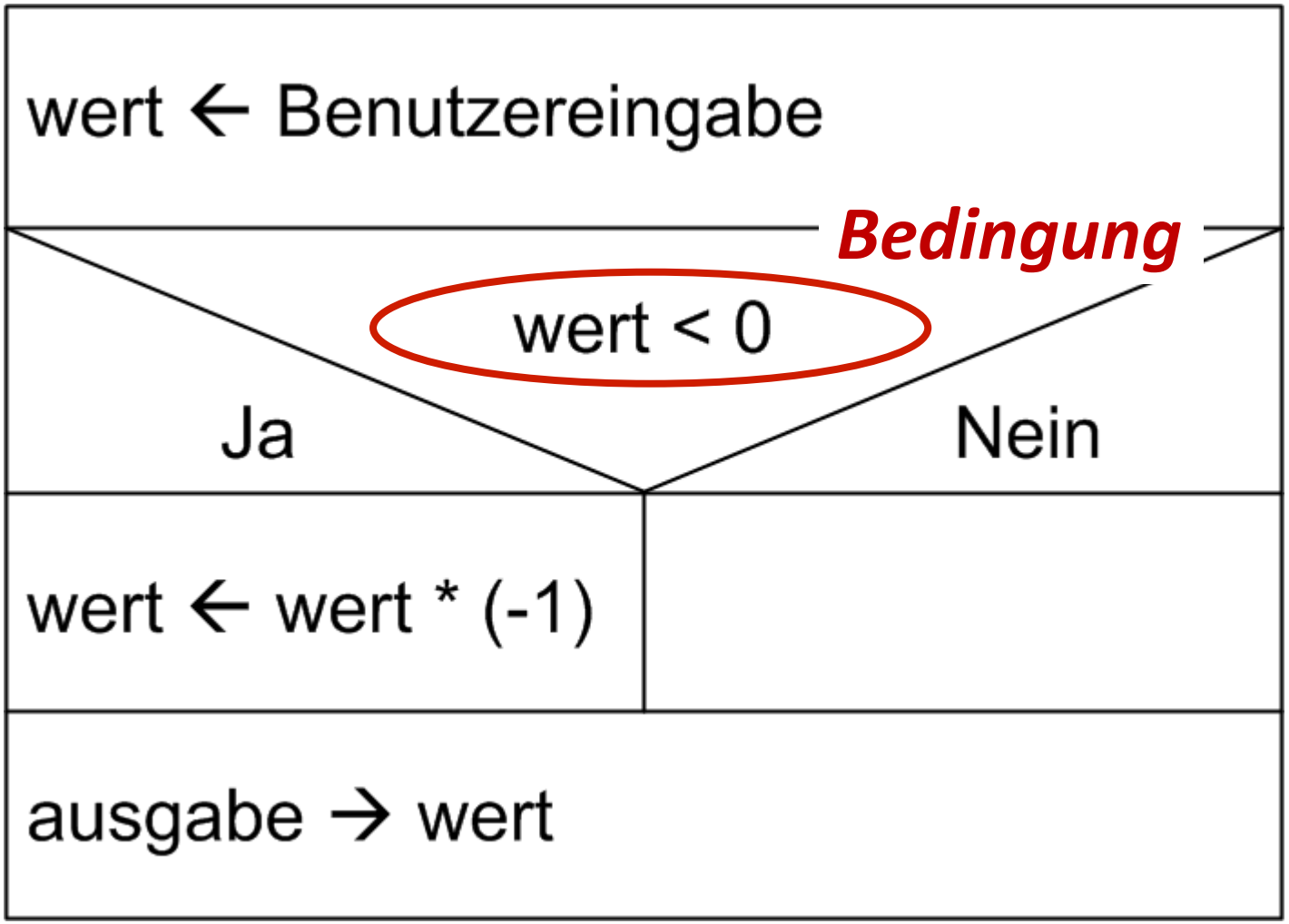
Der Ablauf eines Algorithmus hängt von einer Bedingung ab.

Ablauf der Auswertung …

Eingabe
E

Verarbeitung
V

Ausgabe
A



Umgesetzt in C

Ausgabe:

```
Eingabe: -121
Betrag: 121
```

Eingabebereich

```c
int wert = 0;

printf("Eingabe: ");
scanf("%d",&wert);
```

Verarbeitungsbereich

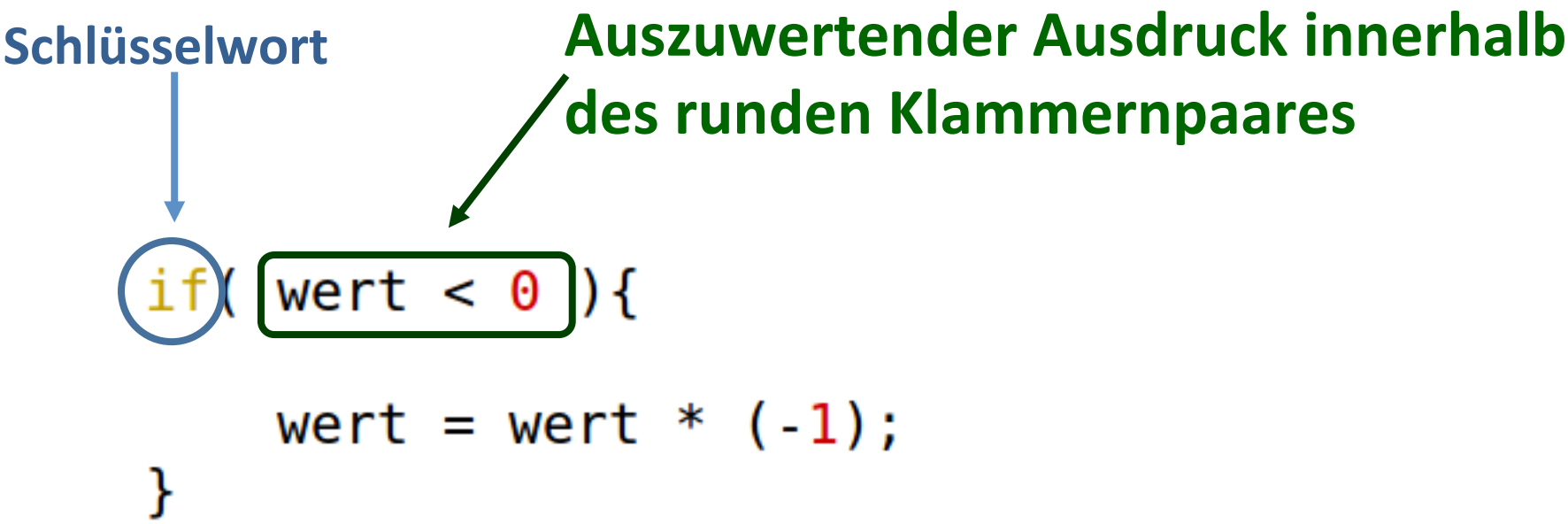
```c
if( wert < 0 ){

    wert = wert * (-1);
}
```

Ausgabebereich

```c
printf("Betrag: %d\n",wert);
```

Die einfache if-Anweisung



1. **Auswertung des Ausdruckes**
2. **Ist das Ergebnis der Auswertung true so wird der folgende Anweisungsblock im geschweiften Klammernblock ausgeführt.**

Die if-else Anweisung

```
int zahl = 0;

printf("Eingabe: ");

if(scanf("%d",&zahl)){
    printf("OK, Eingabe: %d\n", zahl);
}
else{
    printf("Falscheingabe ....\n");
}
```

Auszuwertender Ausdruck innerhalb des runden Klammernpaares

scanf() liefert bei Falscheingabe den Rückgabewert 0.

Anweisung für true

Schlüsselwort für Alternativanweisung

Anweisung für false

Ausgabe:

```
Eingabe: 123
OK, Eingabe: 123
```

Die if-else Anweisung

Achtung!

```
int x = 27, y = 22, temp;

if( x < y)
    temp = x;
    x    = y;
    y    = temp;

printf("%d ist groesser als %d\n", x, y);
```

bei fehlendem geschweiften Klammernblock bezieht sich nur die Folgezeile auf die if-Anweisung

Hauptprogramm

Ausgabe:

```
22 ist groesser als 2543604
```

Wann liefert ein Ausdruck true?

Kurz formuliert:

Bei Ausdrücken ungleich 0

```c
if( wert ){
    // do something
}
```

```c
if( wert%2 ){
    // do something
}
```

```c
if( 124 ){
    // do something
}
```

Wann liefert ein Ausdruck true?

Wiederholung Operatoren

Arithmetische Operatoren

```
int x = 7, y = 10;
int z = 0;

z = x + y;
z = x - y;
z = x * y;
z = x / y;
z = x % y;
z++;
--z;
```

Vergleichsoperatoren

```
int x = 7, y = 10;
int z = 0;

x <  y    // kleiner als
x <= y    // kleiner gleich
x >  y    // groesser als
x >= y    // groesser gleich
x == y    // Vergleich aus
          // Identitaet
x != y    // Vergleich auf
          // Ungleichheit
```

Zuweisungsoperatoren `y = 10;`

Welches Ergebnis liefern die folgenden Ausdrücke?

```
5 >= 7
```
→ ***false***

```
1.7 < 1.8
```
→ ***true***

```
4 + 2 == 5
```
→ ***false***

```
2 * 4 != 7
```
→ ***true***

Vorrang der Vergleichsoperatoren

Priorität	Operator
hoch ↕ niedrig	Arithmetische Operatoren
	< <= > >=
	== !=
	Zuweisungsoperatoren

Welches Ergebnis liefern die folgenden Ausdrücke?

```
int index = 5, max = 6;

int flag = index < max - 1;

printf("Flag: %d\n", flag);
```

höchster Rang

zweiter Rang

niedrigster Rang

Ausgabe:

```
Flag: 0
```

Welches Ergebnis liefern die folgenden Ausdrücke?

```
int length = 4, limit = 5;
int flag = length + 1 == limit;

printf("Flag: %d\n", flag);
```

höchster Rang

zweiter Rang

niedrigster Rang

Ausgabe:

```
Flag: 1
```

Welches Ergebnis liefern die folgenden Ausdrücke?

```
int result = 0, length = 4, limit = 5;
(result = length + 1) == limit;
```

höchster Rang

zweiter Rang

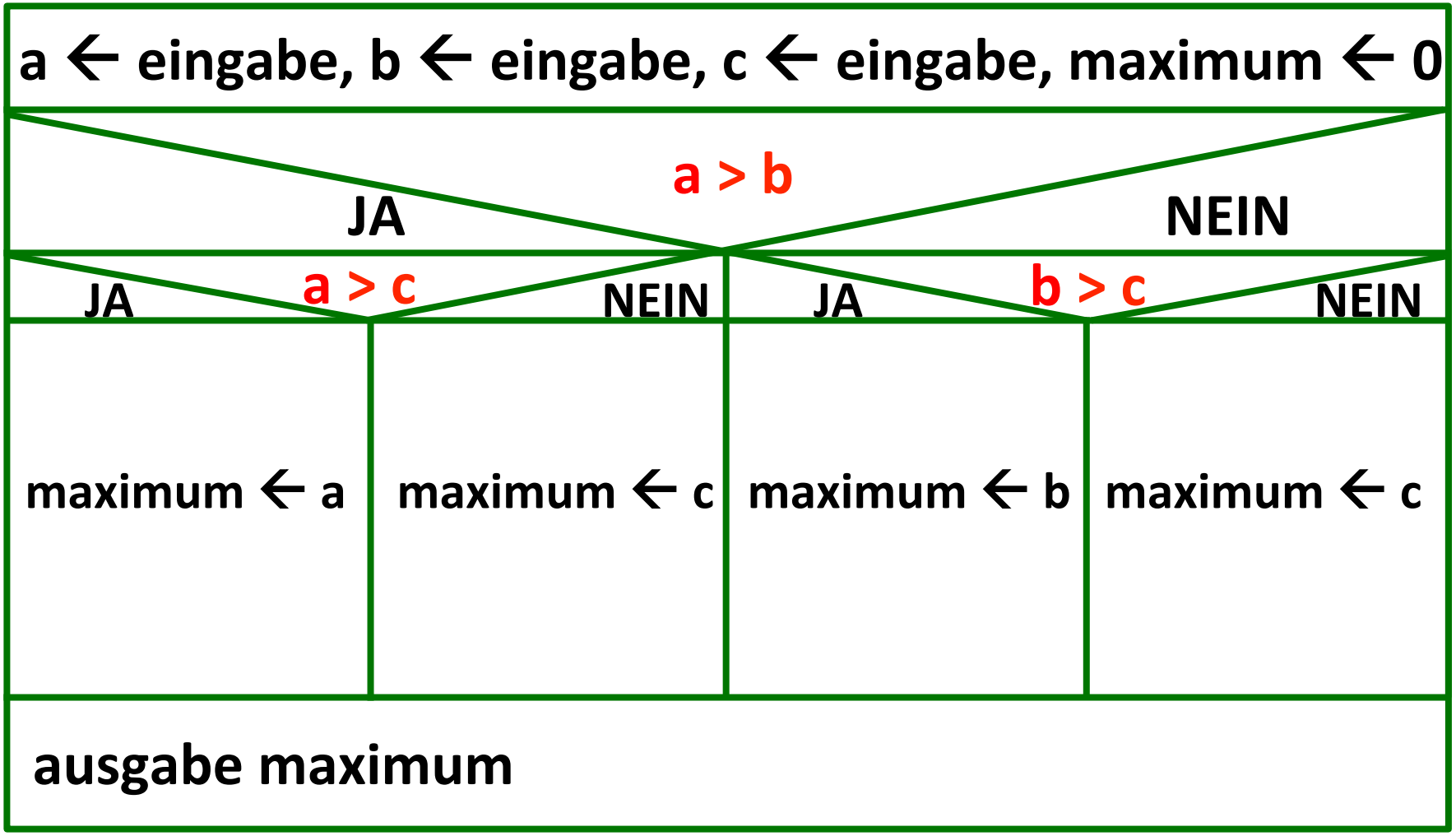
niedrigster Rang

```
printf("Result: %d\n", result);
```

Ausgabe:

```
Result: 5
```

Bestimmung eines Maximums …



… zunächst mit Hilfe eines Struktogramms

Bestimmung eines Maximums

```
int a = 0, b = 0, c = 0, maximum = 0;

printf("3 Werte [a b c]: ");
scanf("%d %d %d",&a, &b ,&c);

if( a > b ){

    if( a > c ){ maximum = a;  }
          else{ maximum = c;  }
}else{

    if( b > c ){ maximum = b;  }
          else{ maximum = c;  }
}

printf("Maximum: %d\n",maximum);
```

aufwändig, oder?

einfacher geht's mit ……

Ausgabe:

```
3 Werte [a b c]: 5 8 2
Maximum: 8
```

Logischen Operatoren

Es handelt sich um die boolschen Operatoren UND, ODER und NICHT

Mit diesen Operatoren ist das Erstellen von zusammengesetzten Bedingungen möglich

UND-Operator &&

liefert true, wenn beide Bedingungen erfüllt sind.

Bedingung 1 **Bedingung 2**

```
( a > b ) && ( a > c )
```

nur true, wenn a größer als b und größer als c ist

Logischen Operatoren

ODER-Operator ||

liefert true, wenn **mindestens eine** der beiden Bedingungen erfüllt sind.

Bedingung 1 **Bedingung 2**

```
( a > b ) || ( a > c )
```

nur true, wenn entweder a größer als b, oder a größer als c oder beide Bedingungen zutreffen

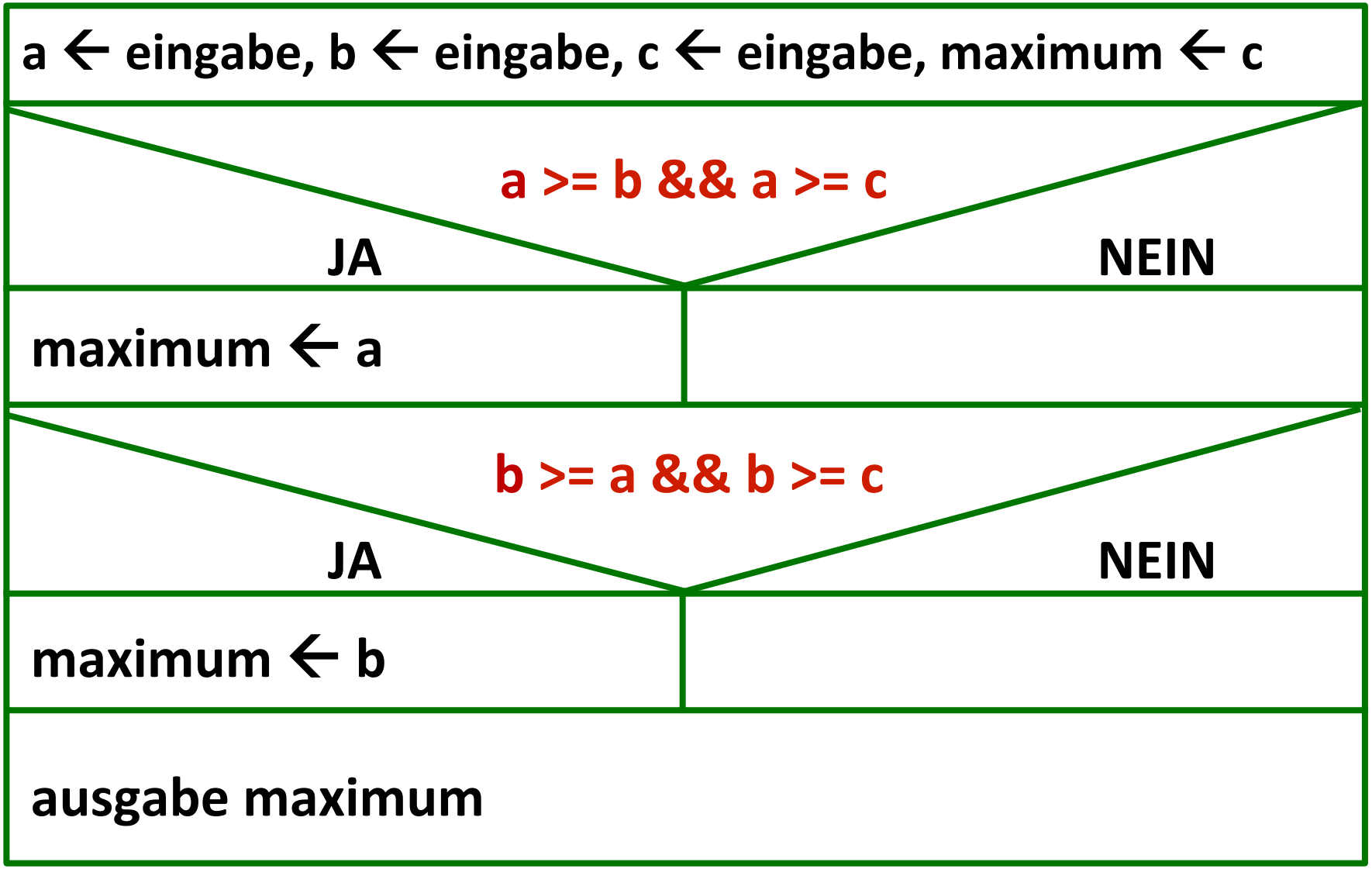
NICHT-Operator !

liefert true, wenn die Bedingung falsch ist.

Wahrheitstafel

A	B	A && B	A || B
true	*true*	*true*	*true*
true	*false*	*false*	*true*
false	*true*	*false*	*true*
false	*false*	*false*	*false*

Bestimmung eines Maximums, vereinfacht



Bestimmung eines Maximums, vereinfacht

```
int a = 0, b = 0, c = 0, maximum = 0;

printf("3 Werte [a b c]: ");
scanf("%d %d %d",&a, &b ,&c);

maximum = c;

if( a >= b && a >= c ){  maximum = a;  }
if( b >= a && b >= c ){  maximum = b;  }

printf("Maximum: %d\n",maximum);
```

Quellcode …

Ausgabe:

```
3 Werte [a b c]: 11 44 22
Maximum: 44
```

Bestimmung der Note in einer Prüfung

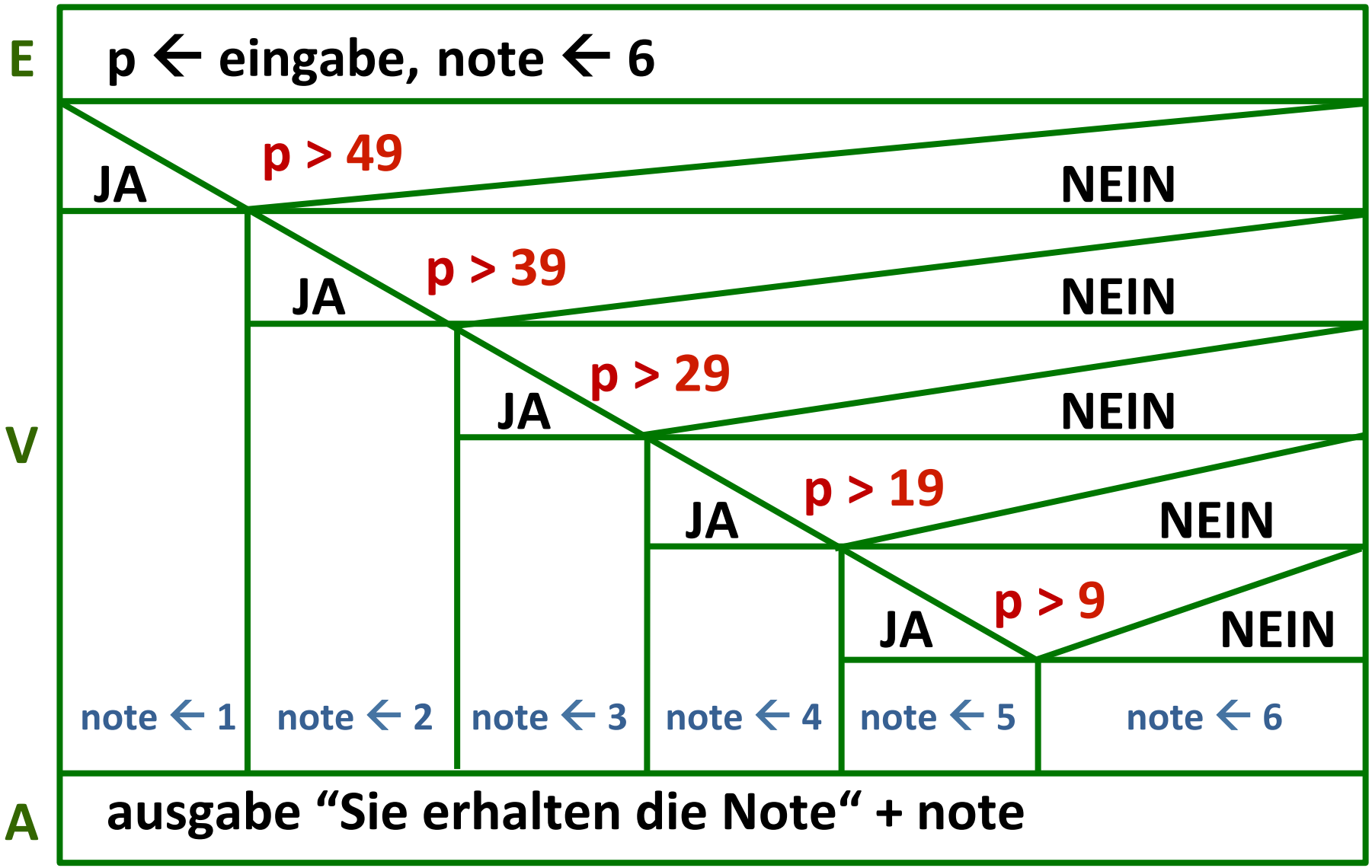
Für eine Prüfung soll sich die Note aus der erreichten Punktezahl ergeben.

Es gilt die Zuordnung:

Punkte	Note
50	1
40 - 49	2
30 - 39	3
20 - 29	4
10 - 19	5
0 - 9	6

Wie kann man dies realisieren?

Versuch 1: Über die einfache if-Anweisung



Realisierung im Quellcode

E

```c
int punkte = 0, note = 6;

printf("Punkte: ");
scanf("%d",&punkte);
```

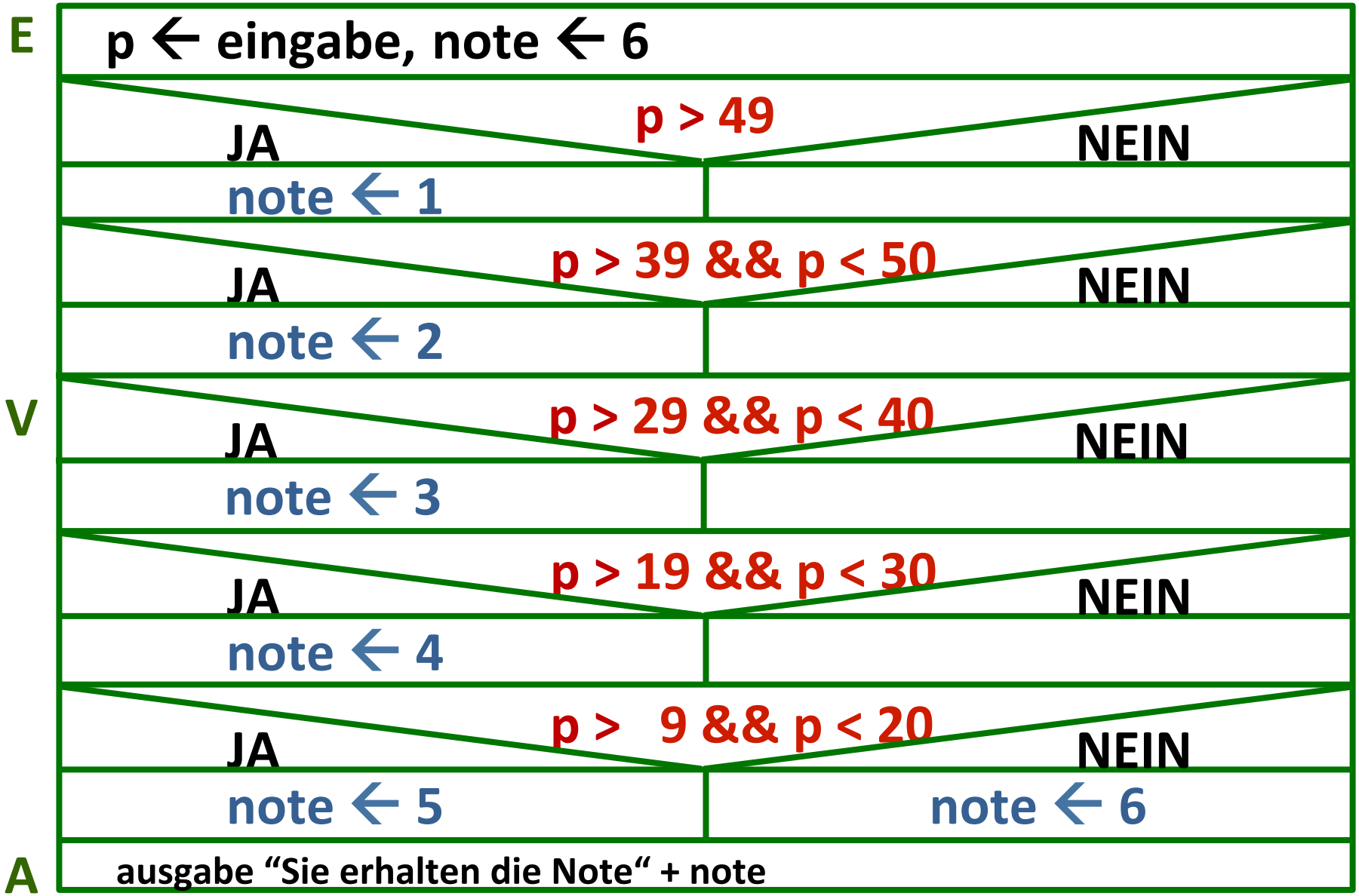
V

```c
if( punkte > 49 ){  note = 1; }
else { if( punkte > 39 ){  note = 2; }
    else { if( punkte > 29 ){  note = 3; }
        else { if( punkte > 19 ){  note = 4; }
            else { if( punkte >  9 ){  note = 5; }
            }
        }
    }
}
```

A

```c
printf("Note: %d\n", note);
```

Versuch 2: Über if-Anweisung und && - Operator



Realisierung im Quellcode

E

```c
int punkte = 0, note = 6;

printf("Punkte: ");
scanf("%d",&punkte);
```

V

```c
if( punkte > 49 ){  note = 1; }
if( punkte > 39 && punkte < 50 ){  note = 2; }
if( punkte > 29 && punkte < 40 ){  note = 3; }
if( punkte > 19 && punkte < 30 ){  note = 4; }
if( punkte >  9 && punkte < 20 ){  note = 5; }
```

A

```c
printf("Note: %d\n", note);
```

Alternative mit switch-case

E

```
int punkte = 0, note = 6;

printf("Punkte: ");
scanf("%d",&punkte);
```

Sonderfall Bedingte Anweisung nur für int-Werte

V

```
switch(punkte/10){

    case 5 : note = 1;
             break;
    case 4 : note = 2;
             break;
    case 3 : note = 3;
             break;
    case 2 : note = 4;
             break;
    case 1 : note = 5;
             break;
    default: note = 6;
}
```

Übereinstimmung?

Sofortiges Verlassen der switch-Anweisung

Falls kein Kriterium zutrifft.

A

```
printf("Note: %d\n", note);
```

Weiter geht's mit Übungen ….