



```
void A(int a) {
    if (a > 3) return;
    std::cout << a;
    A(a+1);
}
void B(int a) {
    if (a > 3) return;
    B(a+1);
    std::cout << a;
}
void C(int a) {
    while (a <= 3){
        std::cout << a++;
    }
}
```

Welche der Aufrufe A(1), B(1) und C(1) erzeugen dieselbe Ausgabe?

- 1 A(1) und B(1)
- 2 A(1) und C(1)
- 3 B(1) und C(1)
- 4 A(1), B(1) und C(1)
- 5 keine



```
void A(int a) {
    if (a > 3) return;
    std::cout << a; // 123
    A(a+1);
}
void B(int a) {
    if (a > 3) return;
    B(a+1);
    std::cout << a;
}
void C(int a) {
    while (a <= 3){
        std::cout << a++;
    }
}
```

Welche der Aufrufe A(1), B(1) und C(1) erzeugen dieselbe Ausgabe?

- 1 A(1) und B(1)
- 2 A(1) und C(1)
- 3 B(1) und C(1)
- 4 A(1), B(1) und C(1)
- 5 keine



```
void A(int a) {
    if (a > 3) return;
    std::cout << a; // 123
    A(a+1);
}
void B(int a) {
    if (a > 3) return;
    B(a+1);
    std::cout << a; // 321
}
void C(int a) {
    while (a <= 3){
        std::cout << a++;
    }
}
```

Welche der Aufrufe A(1), B(1) und C(1) erzeugen dieselbe Ausgabe?

- 1 A(1) und B(1)
- 2 A(1) und C(1)
- 3 B(1) und C(1)
- 4 A(1), B(1) und C(1)
- 5 keine



```
void A(int a) {
    if (a > 3) return;
    std::cout << a; // 123
    A(a+1);
}
void B(int a) {
    if (a > 3) return;
    B(a+1);
    std::cout << a; // 321
}
void C(int a) {
    while (a <= 3){
        std::cout << a++; // 123
    }
}
```

Welche der Aufrufe A(1), B(1) und C(1) erzeugen dieselbe Ausgabe?

- 1 A(1) und B(1)
- 2 A(1) und C(1)
- 3 B(1) und C(1)
- 4 A(1), B(1) und C(1)
- 5 keine



```
void A(int a) {
    if (a > 3) return;
    std::cout << a; // 123
    A(a+1);
}
void B(int a) {
    if (a > 3) return;
    B(a+1);
    std::cout << a; // 321
}
void C(int a) {
    while (a <= 3){
        std::cout << a++; // 123
    }
}
```

Welche der Aufrufe A(1), B(1) und C(1) erzeugen dieselbe Ausgabe?

- 1 A(1) und B(1)
- 2 A(1) und C(1)
- 3 B(1) und C(1)
- 4 A(1), B(1) und C(1)
- 5 keine



```
void A(int a) {  
    if (a > 3) return;  
    std::cout << a; // 123  
    A(a+1);  
}
```

Funktion A ist *endrekursiv*: sie enthält nur einen einzigen rekursiven Aufruf ganz am Ende.

Endrekursive Funktionen können besonders leicht iterativ (C!) geschrieben werden.

Welche der Aufrufe A(1), B(1) und C(1) erzeugen dieselbe Ausgabe?

- 1 A(1) und B(1)
- 2 A(1) und C(1) ●
- 3 B(1) und C(1)
- 4 A(1), B(1) und C(1)
- 5 keine



```
void B(int a) {  
    if (a > 3) return;  
    B(a+1);  
    std::cout << a; // 321  
}
```

B(1) a=1

Welche der Aufrufe A(1), B(1) und C(1) erzeugen dieselbe Ausgabe?

- 1 A(1) und B(1)
- 2 A(1) und C(1) ●
- 3 B(1) und C(1)
- 4 A(1), B(1) und C(1)
- 5 keine



```
void B(int a) {  
    if (a > 3) return;  
    B(a+1);  
    std::cout << a; // 321  
}
```

B(2) a=2



B(1) a=1

Welche der Aufrufe A(1), B(1) und C(1) erzeugen dieselbe Ausgabe?

- 1 A(1) und B(1)
- 2 A(1) und C(1) ●
- 3 B(1) und C(1)
- 4 A(1), B(1) und C(1)
- 5 keine





```
void B(int a) {  
    if (a > 3) return;  
    B(a+1);  
    std::cout << a; // 321  
}
```

B(3) a=3



B(2) a=2



B(1) a=1

Welche der Aufrufe A(1), B(1) und C(1) erzeugen dieselbe Ausgabe?

- 1 A(1) und B(1)
- 2 A(1) und C(1) ●
- 3 B(1) und C(1)
- 4 A(1), B(1) und C(1)
- 5 keine



```
void B(int a) {  
    if (a > 3) return;  
    B(a+1);  
    std::cout << a; // 321  
}
```

B(4) a=4



B(3) a=3



B(2) a=2



B(1) a=1

Welche der Aufrufe A(1), B(1) und C(1) erzeugen dieselbe Ausgabe?

- 1 A(1) und B(1)
- 2 A(1) und C(1) ●
- 3 B(1) und C(1)
- 4 A(1), B(1) und C(1)
- 5 keine



```
void B(int a) {  
    if (a > 3) return;  
    B(a+1);  
    std::cout << a; // 321  
}
```

B(4) a=4  $\longrightarrow$  return



B(3) a=3



B(2) a=2



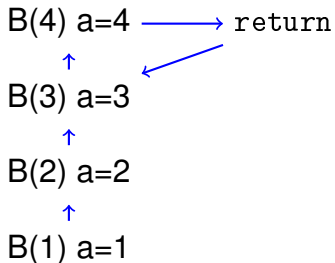
B(1) a=1

Welche der Aufrufe A(1), B(1) und C(1) erzeugen dieselbe Ausgabe?

- 1 A(1) und B(1)
- 2 A(1) und C(1) ●
- 3 B(1) und C(1)
- 4 A(1), B(1) und C(1)
- 5 keine



```
void B(int a) {  
    if (a > 3) return;  
    B(a+1);  
    std::cout << a; // 321  
}
```

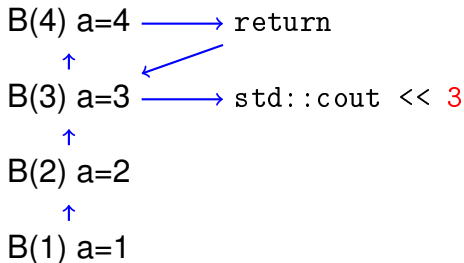


Welche der Aufrufe A(1), B(1) und C(1) erzeugen dieselbe Ausgabe?

- 1 A(1) und B(1)
- 2 A(1) und C(1) ●
- 3 B(1) und C(1)
- 4 A(1), B(1) und C(1)
- 5 keine



```
void B(int a) {  
    if (a > 3) return;  
    B(a+1);  
    std::cout << a; // 321  
}
```

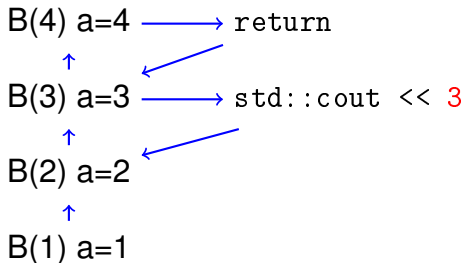


Welche der Aufrufe A(1), B(1) und C(1) erzeugen dieselbe Ausgabe?

- 1 A(1) und B(1)
- 2 A(1) und C(1) ●
- 3 B(1) und C(1)
- 4 A(1), B(1) und C(1)
- 5 keine



```
void B(int a) {  
    if (a > 3) return;  
    B(a+1);  
    std::cout << a; // 321  
}
```



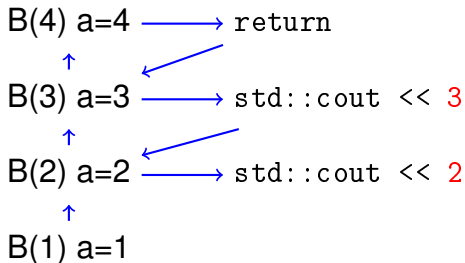
Welche der Aufrufe A(1), B(1) und C(1) erzeugen dieselbe Ausgabe?

- 1 A(1) und B(1)
- 2 A(1) und C(1) ●
- 3 B(1) und C(1)
- 4 A(1), B(1) und C(1)
- 5 keine

# Rekursion



```
void B(int a) {  
    if (a > 3) return;  
    B(a+1);  
    std::cout << a; // 321  
}
```



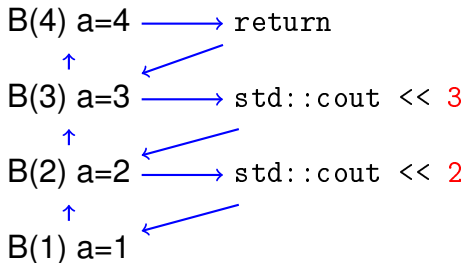
Welche der Aufrufe A(1), B(1) und C(1) erzeugen dieselbe Ausgabe?

- 1 A(1) und B(1)
- 2 A(1) und C(1) ●
- 3 B(1) und C(1)
- 4 A(1), B(1) und C(1)
- 5 keine

# Rekursion



```
void B(int a) {  
    if (a > 3) return;  
    B(a+1);  
    std::cout << a; // 321  
}
```



Welche der Aufrufe A(1), B(1) und C(1) erzeugen dieselbe Ausgabe?

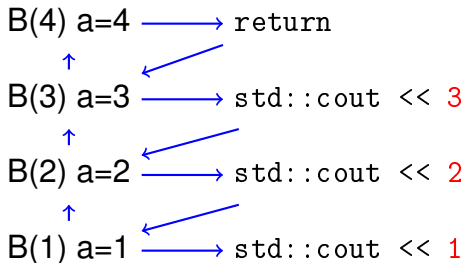
- 1 A(1) und B(1)
- 2 A(1) und C(1) ●
- 3 B(1) und C(1)
- 4 A(1), B(1) und C(1)
- 5 keine



# Rekursion



```
void B(int a) {  
    if (a > 3) return;  
    B(a+1);  
    std::cout << a; // 321  
}
```



Welche der Aufrufe A(1), B(1) und C(1) erzeugen dieselbe Ausgabe?

- 1 A(1) und B(1)
- 2 A(1) und C(1) ●
- 3 B(1) und C(1)
- 4 A(1), B(1) und C(1)
- 5 keine