

LISTER

INF100

HØST 2026

Torstein Strømme

Vikar: Illimar Rekand

RECAP LØKKER

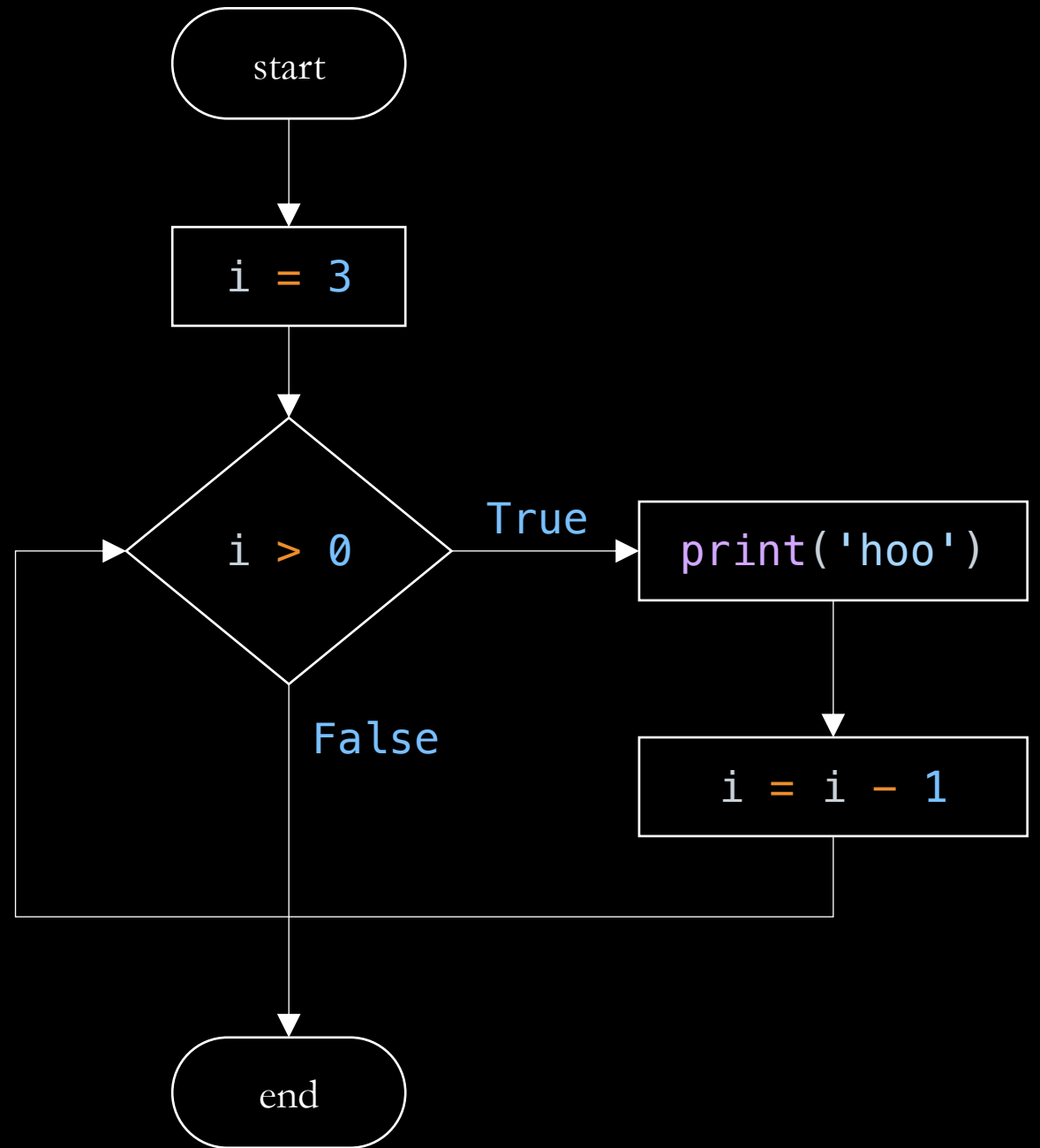
betingelse. et uttrykk som evaluerer til en boolsk verdi (True eller False)

```
i = 3
while i > 0:
    print('hoo')
    i = i - 1
```

} while-setning

løkke kropp. innrykket blokk av setninger som utføres så lenge betingelsen er True.

```
i = 3
while i > 0:
    print('hoo')
    i = i - 1
```

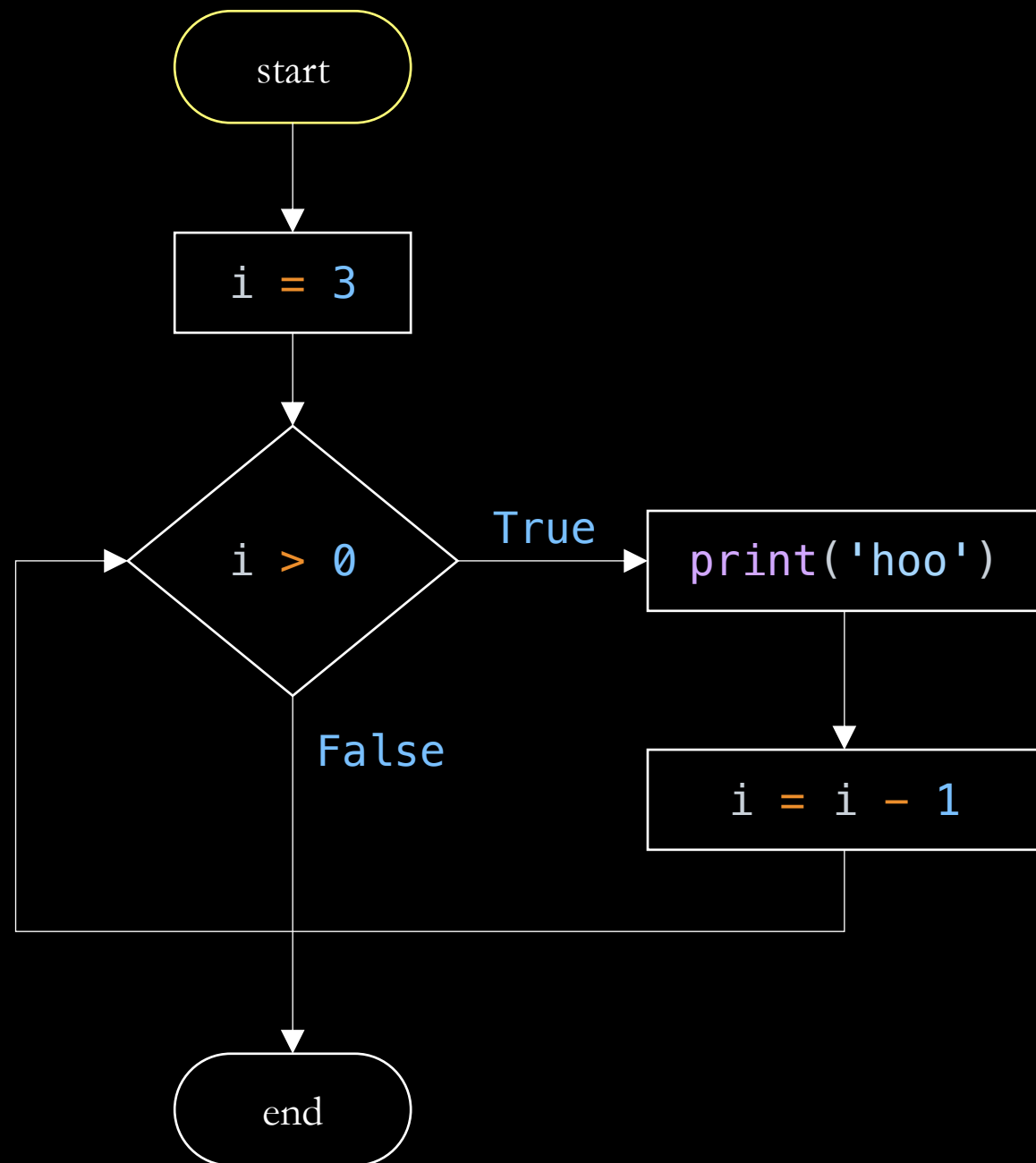
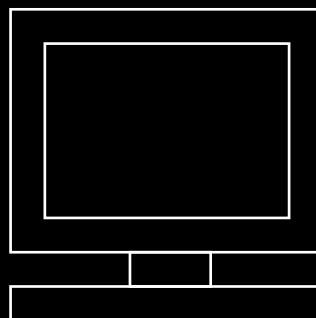




```
i = 3
while i > 0:
    print('hoo')
    i = i - 1
```

Variabler

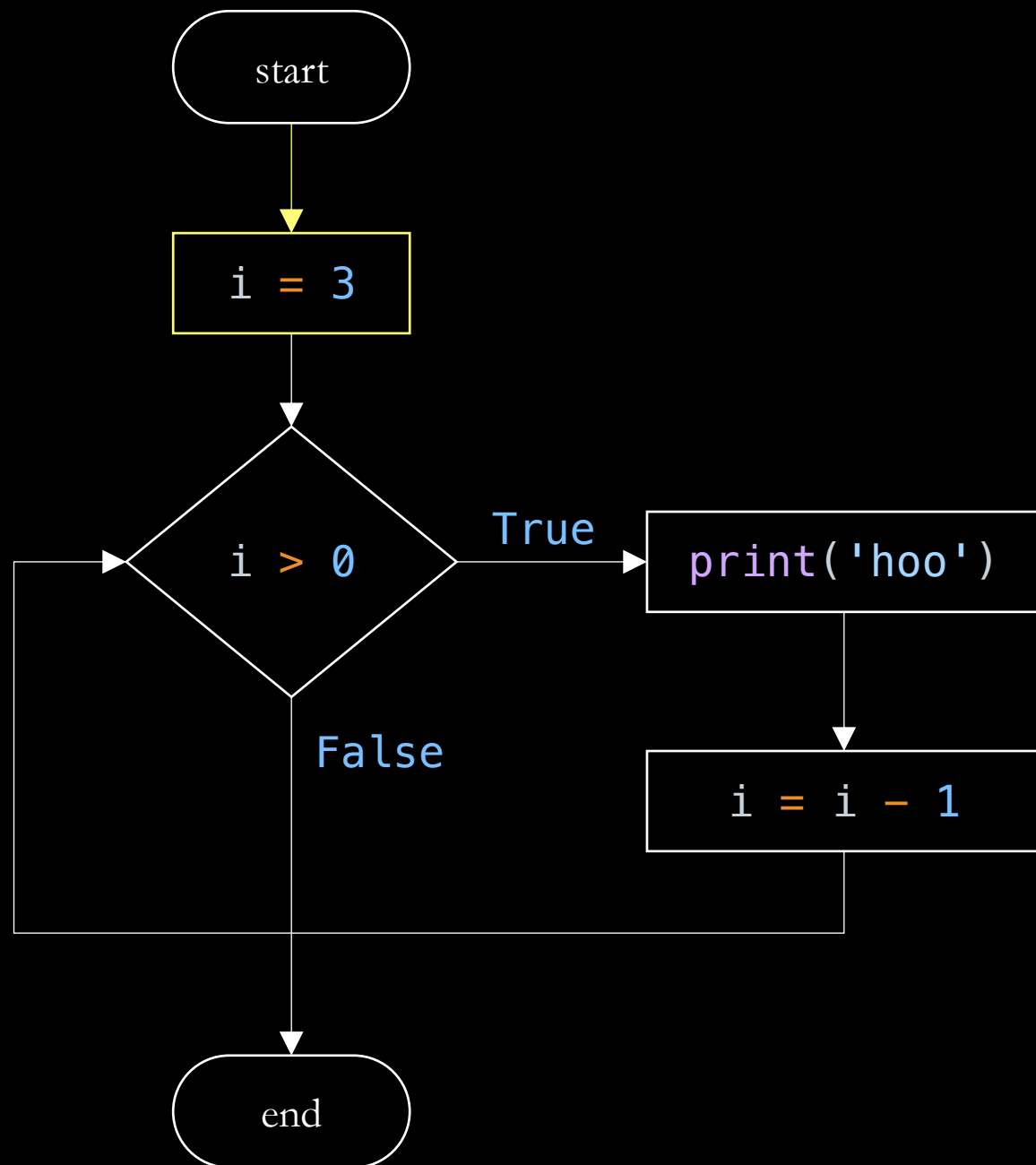
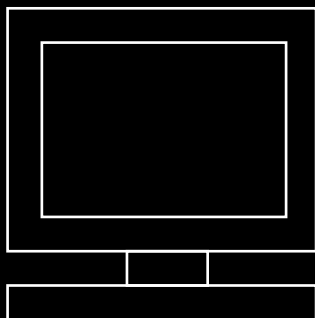
Objekter



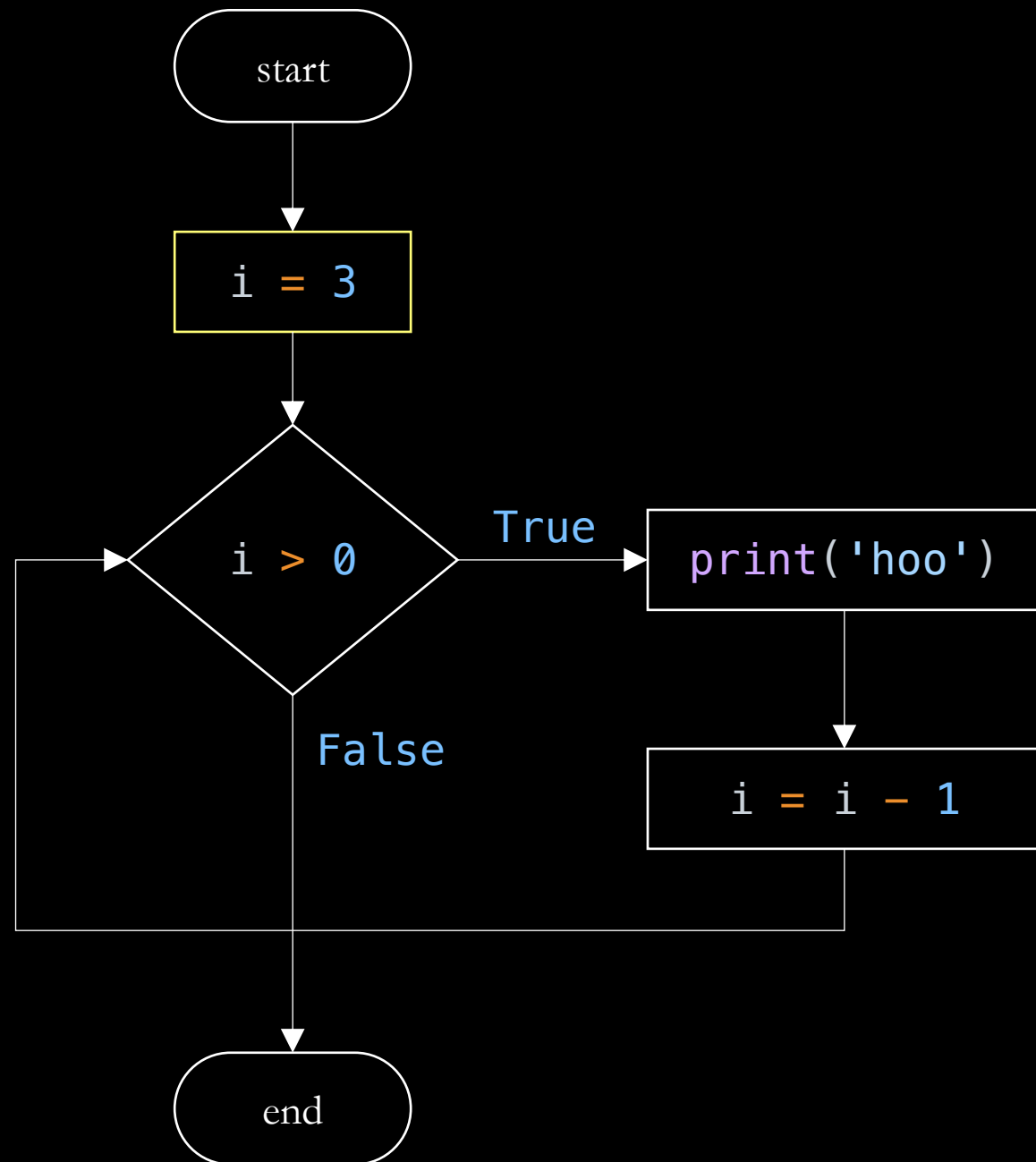
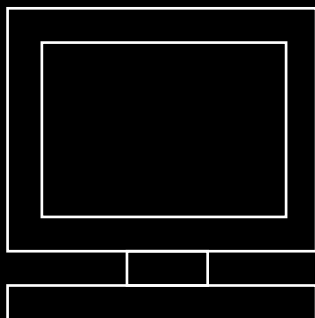
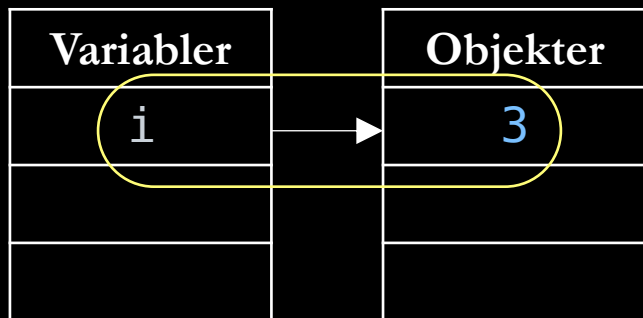
→
`i = 3`
`while i > 0:`
 `print('hoo')`
 `i = i - 1`

Variabler

Objekter



→ `i = 3`
`while i > 0:`
 `print('hoo')`
 `i = i - 1`

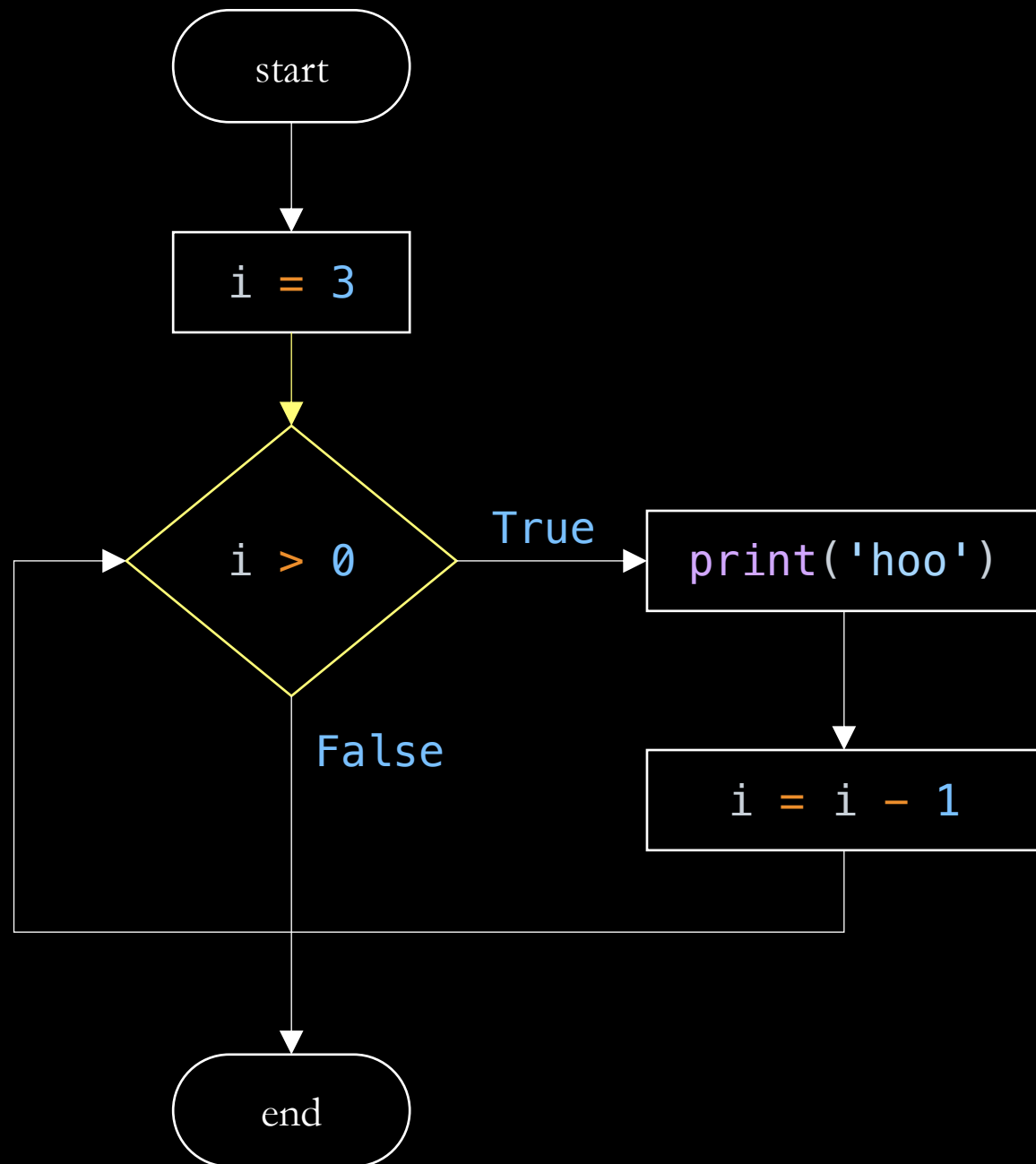
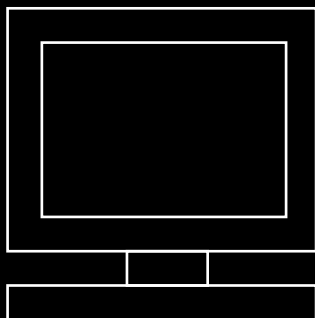


→ `i = 3`
`while i > 0:`
 `print('hoo')`
 `i = i - 1`

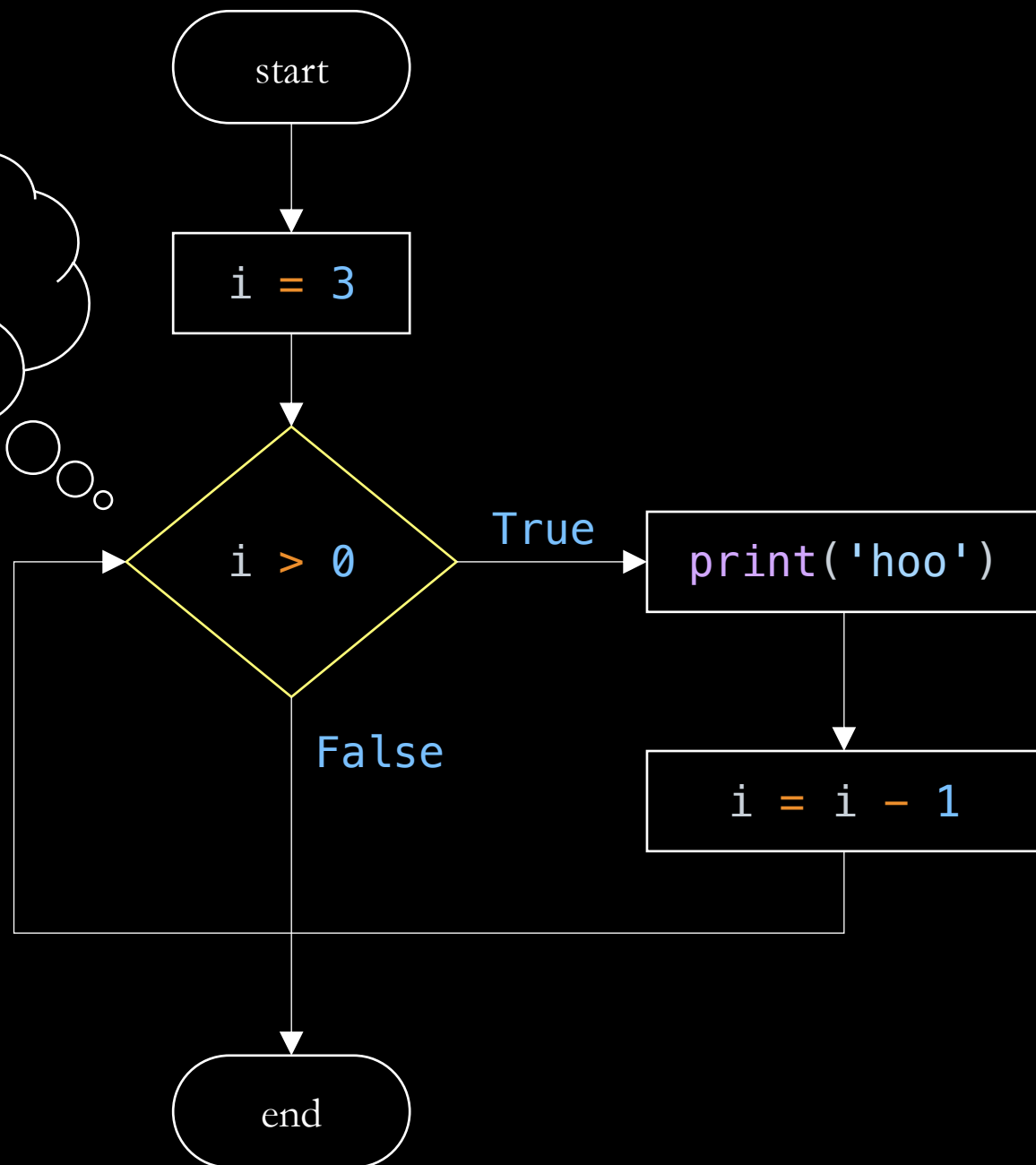
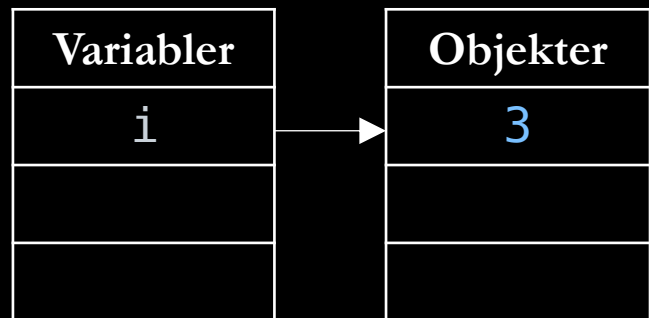
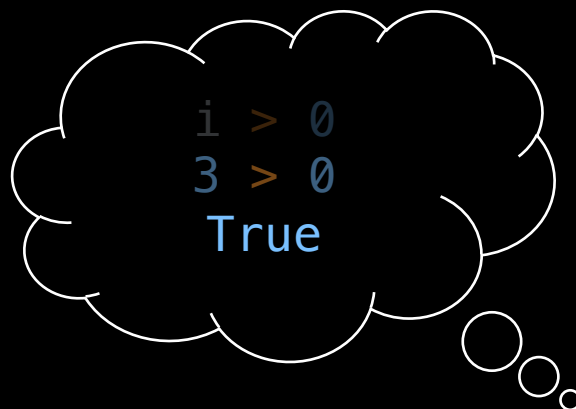
Variabler
i



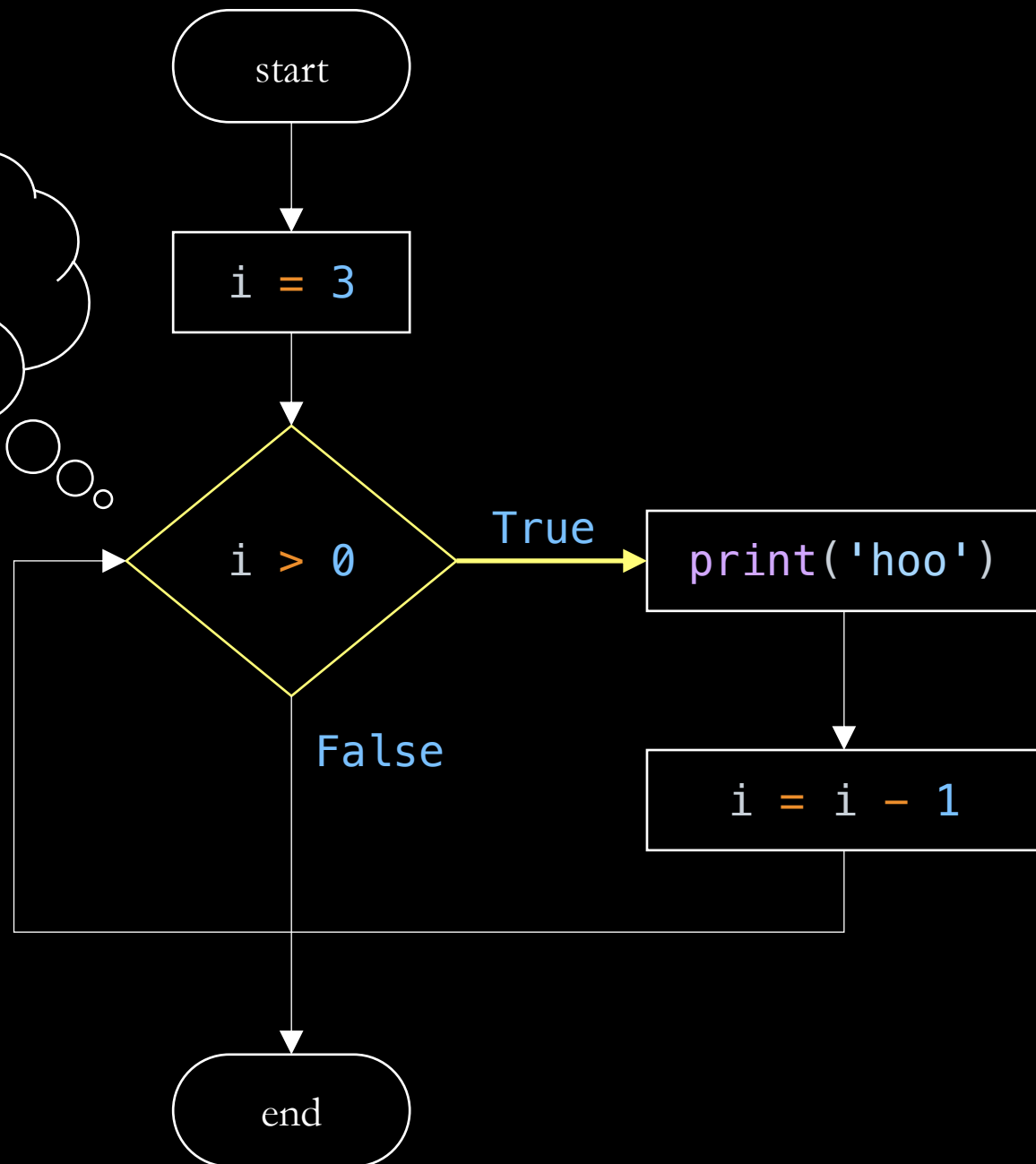
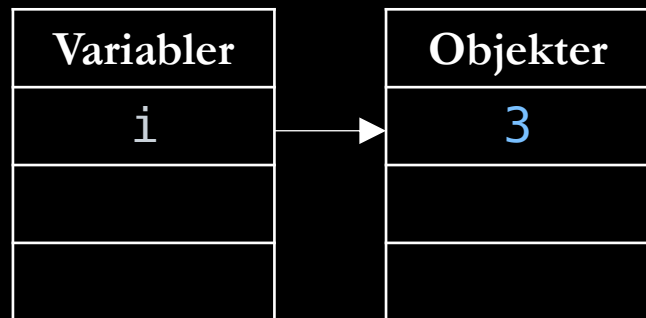
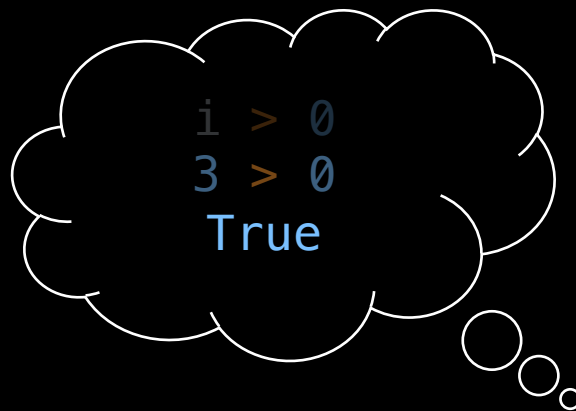
Objekter
3



→ `i = 3`
`while i > 0:`
 `print('hoo')`
 `i = i - 1`



→ `i = 3`
`while i > 0:`
 `print('hoo')`
 `i = i - 1`



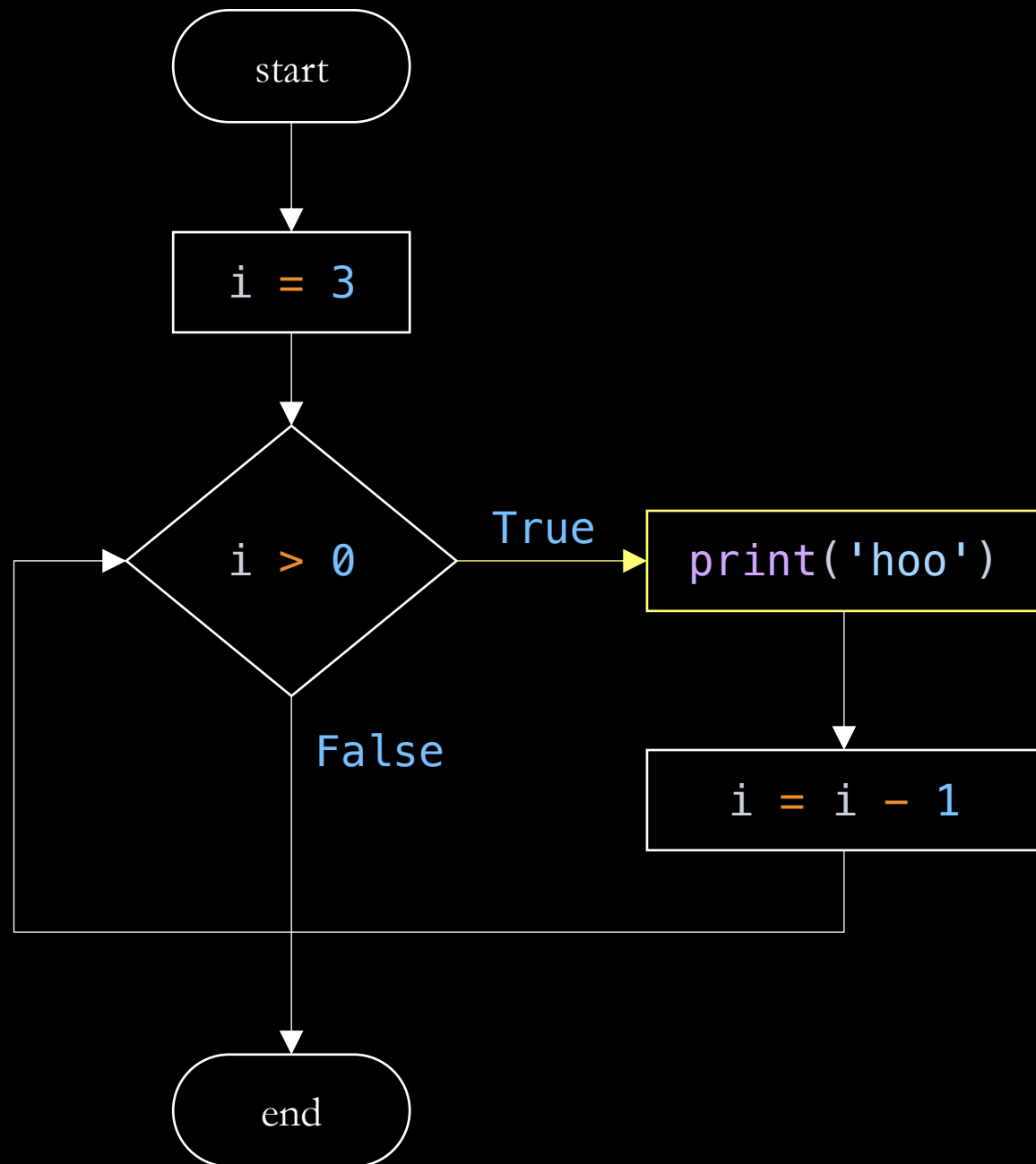
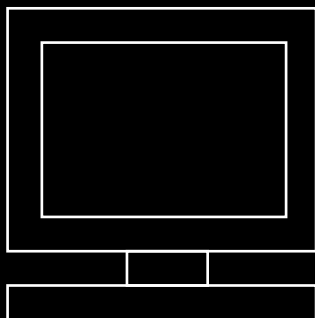
→

```
i = 3
while i > 0:
    print('hoo')
    i = i - 1
```

Variabler
i



Objekter
3

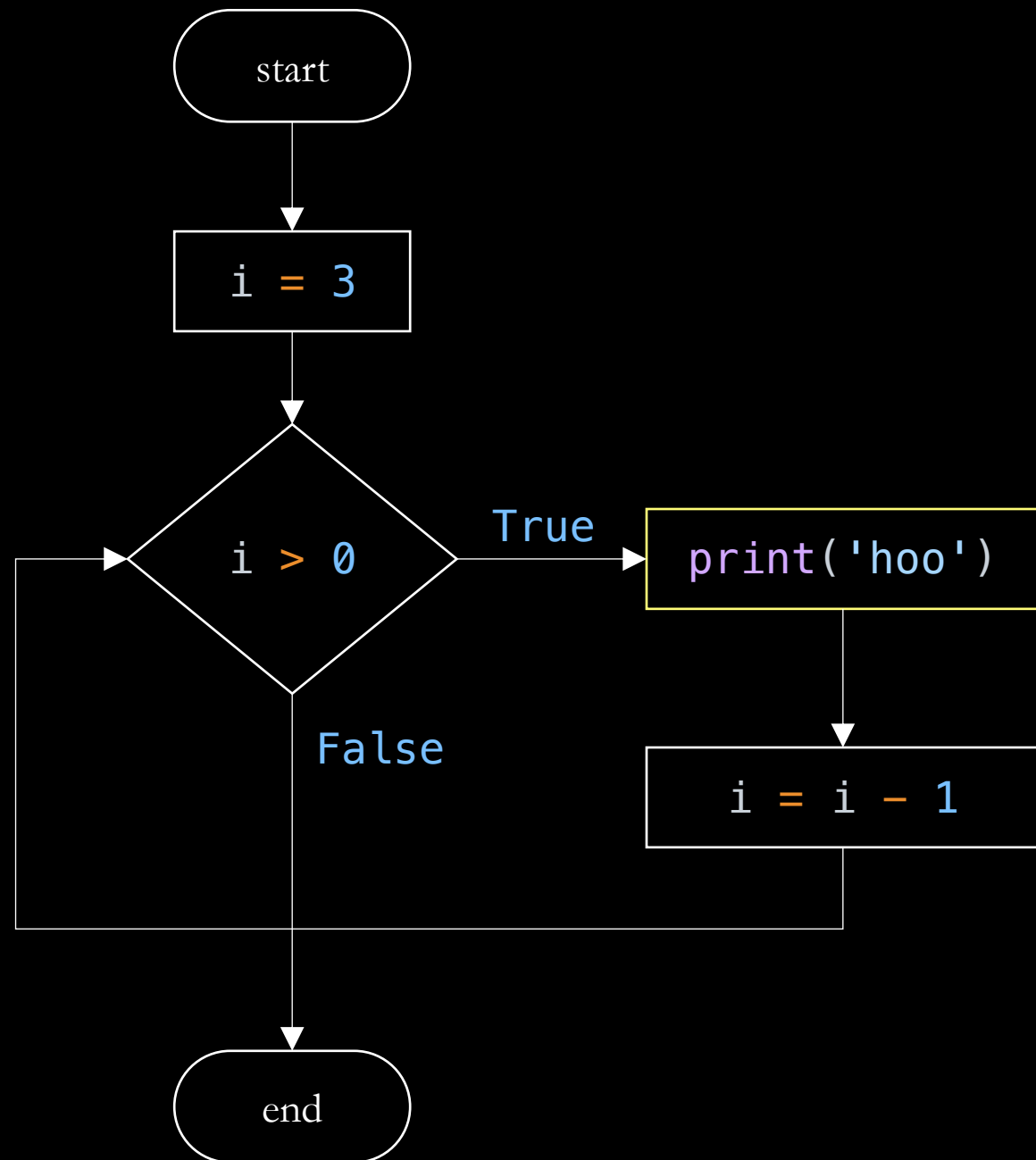
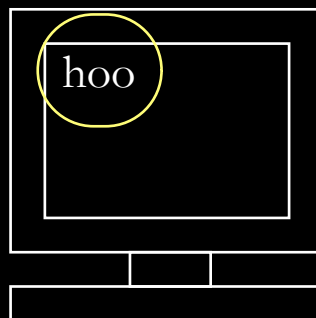


→
`i = 3`
`while i > 0:`
 `print('hoo')`
 `i = i - 1`

Variabler
i



Objekter
3

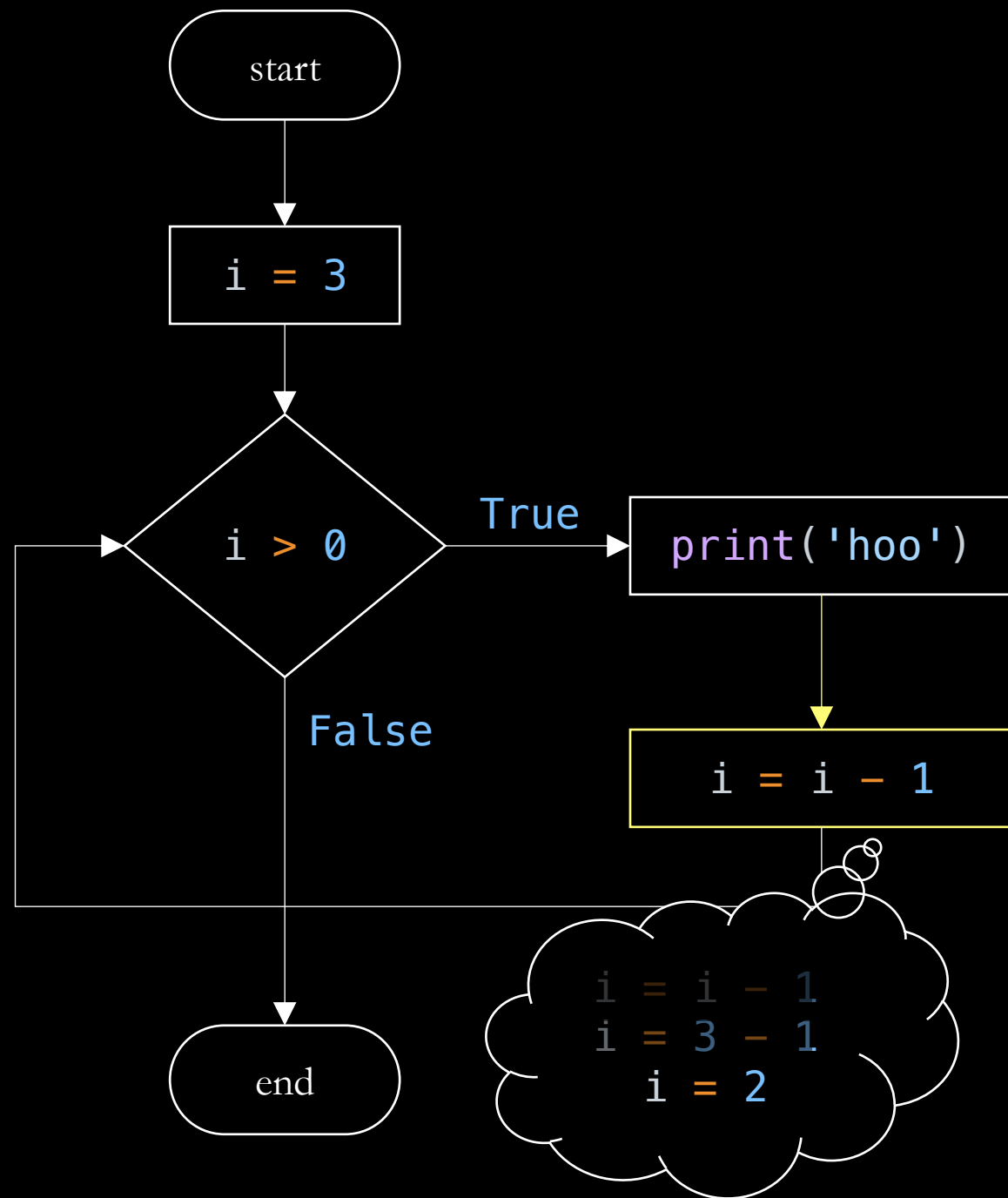
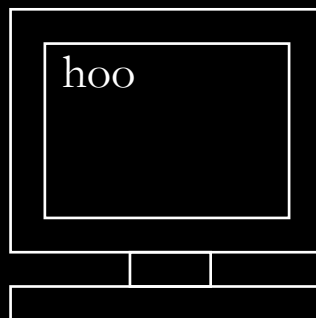


→
`i = 3`
`while i > 0:`
 `print('hoo')`
 `i = i - 1`

Variabler
i

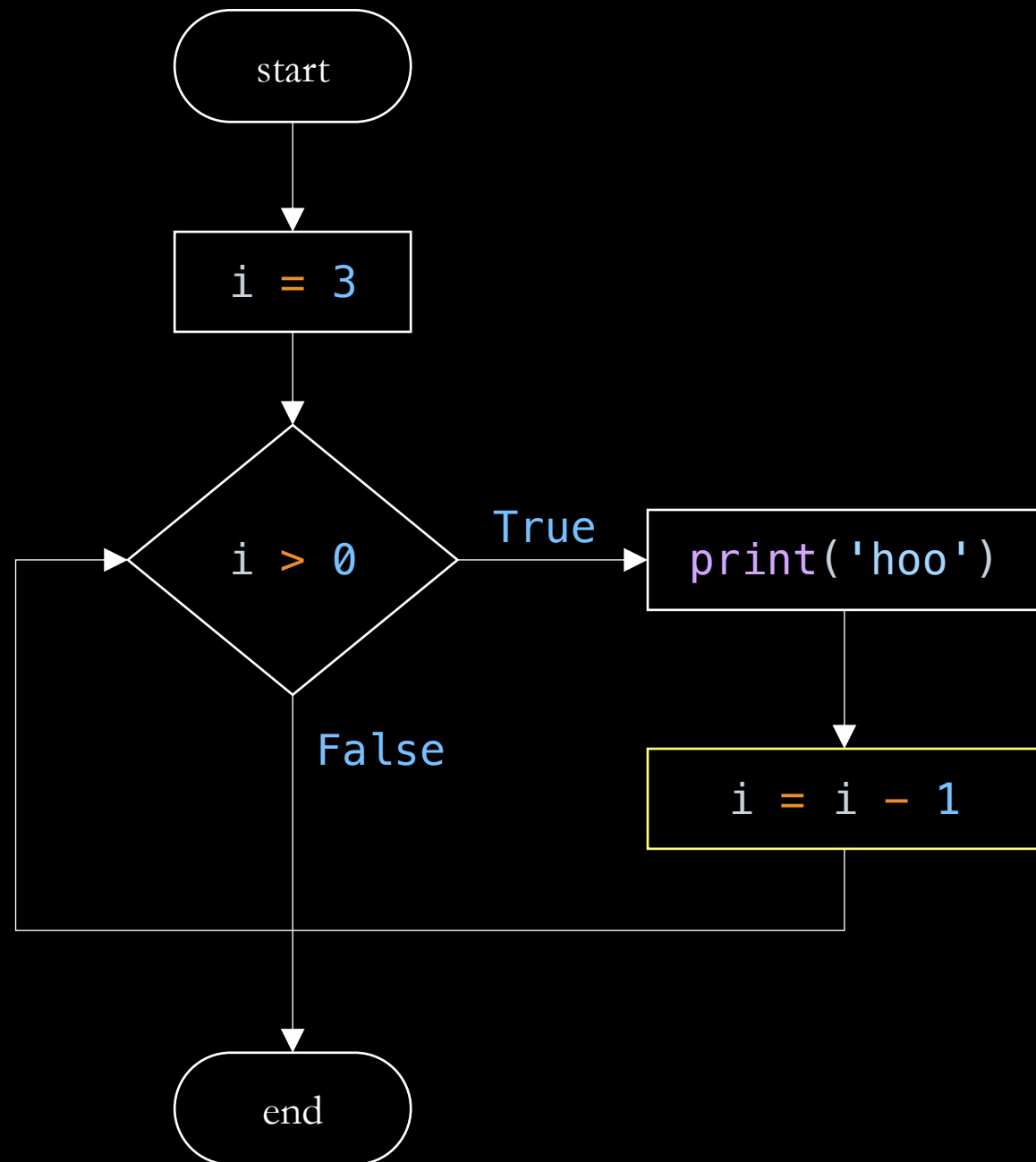
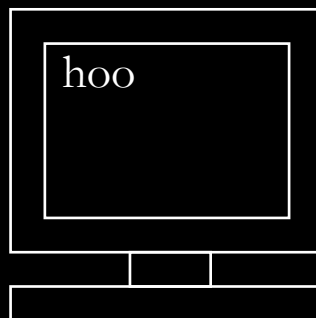


Objekter
3



→
`i = 3`
`while i > 0:`
 `print('hoo')`
 `i = i - 1`

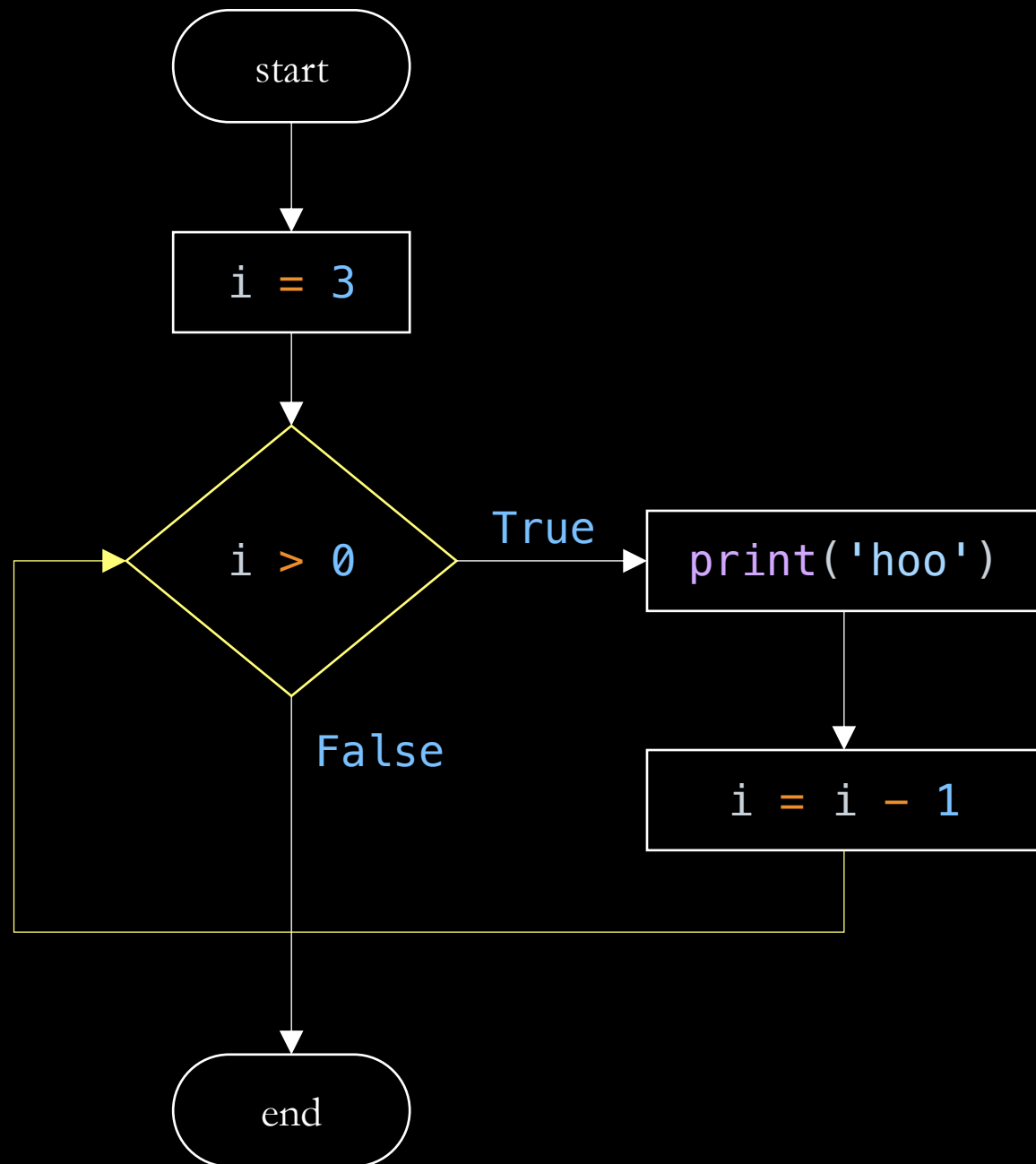
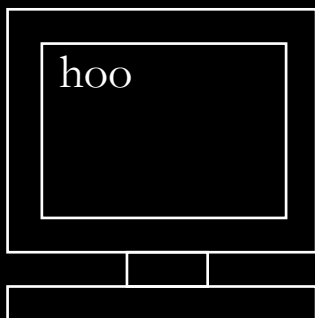
Variabler	Objekter
i	3
	2



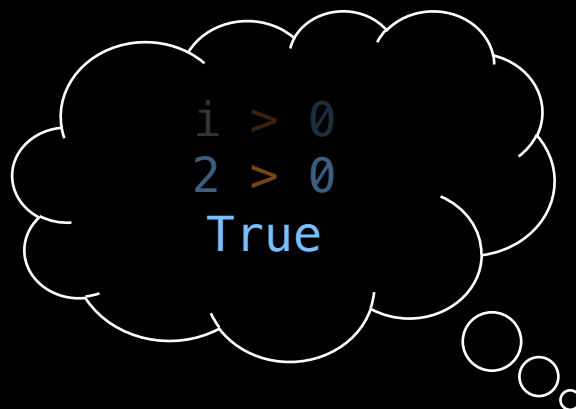
→ `i = 3`
`while i > 0:`
 `print('hoo')`
 `i = i - 1`

Variabler
i

Objekter
3
2

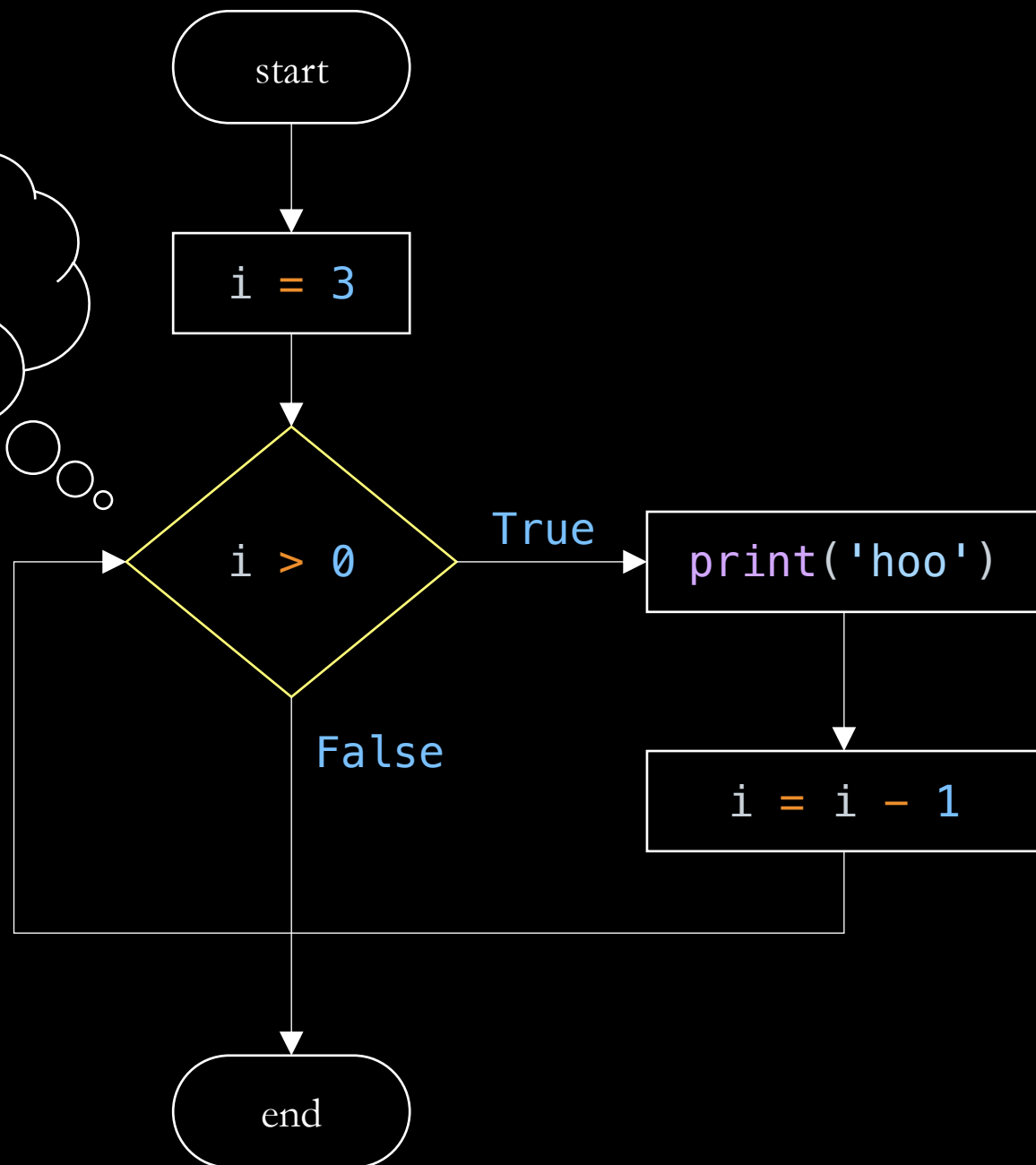
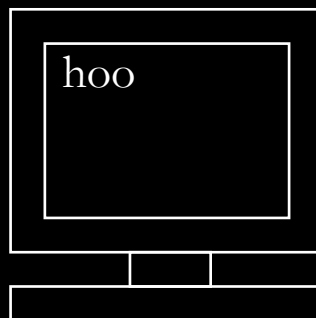


→ `i = 3`
`while i > 0:`
 `print('hoo')`
 `i = i - 1`

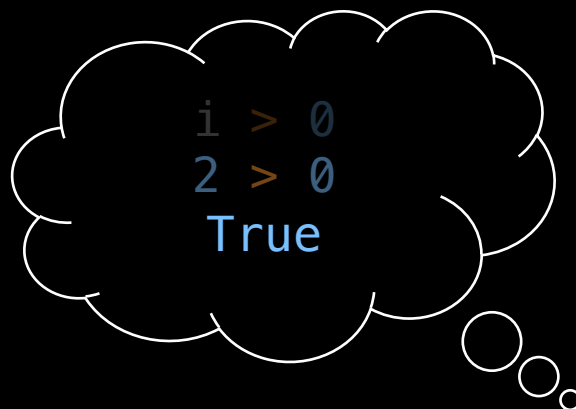


Variabler
i

Objekter
3
2

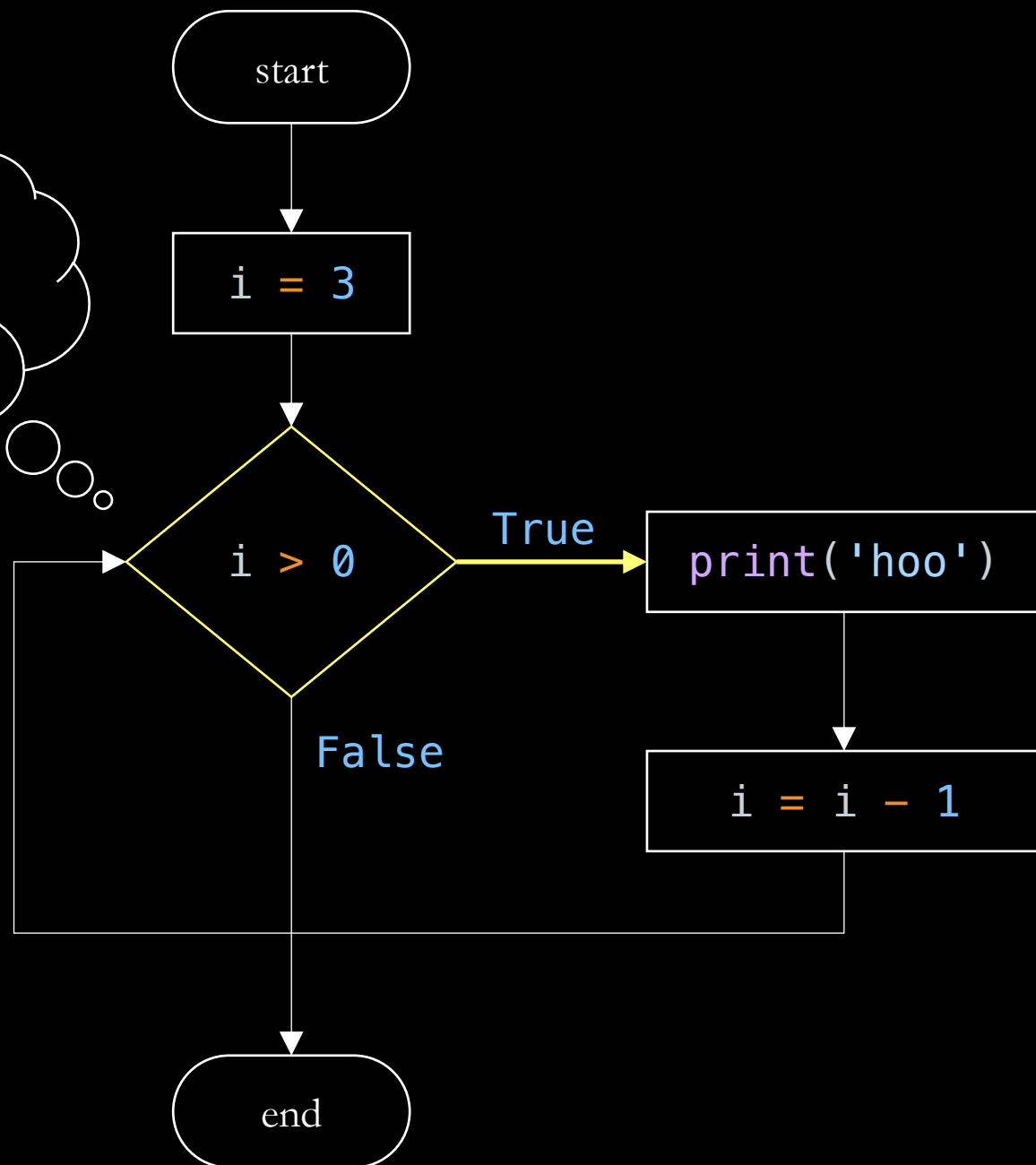
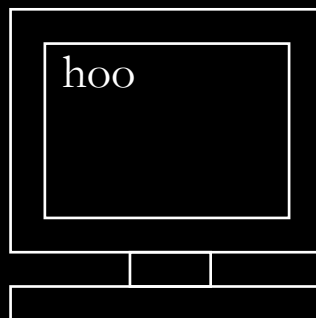


→ `i = 3`
`while i > 0:`
 `print('hoo')`
 `i = i - 1`



Variabler
i

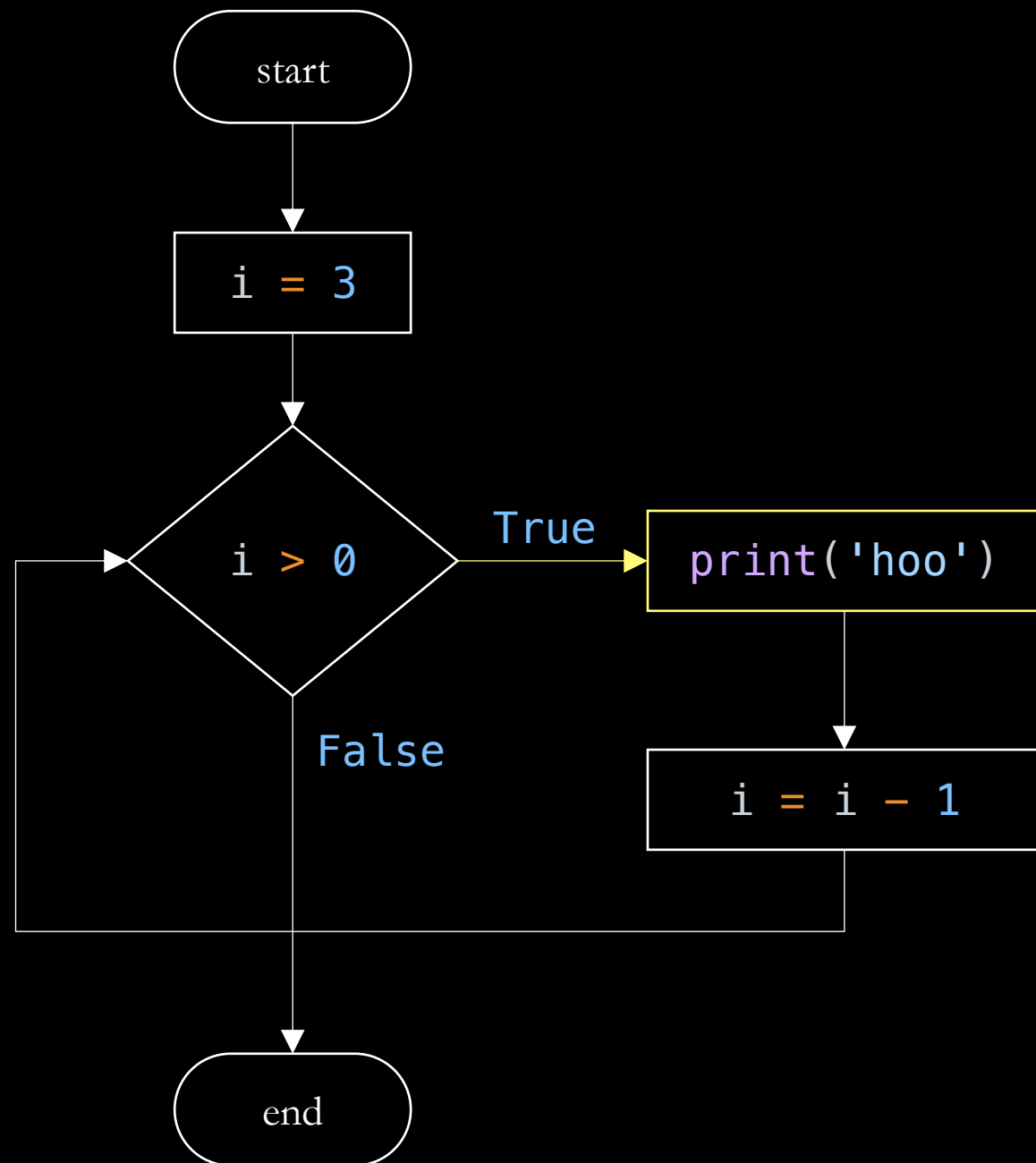
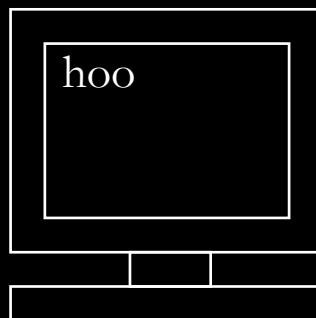
Objekter
3
2



→
`i = 3`
`while i > 0:`
 `print('hoo')`
 `i = i - 1`

Variabler
i

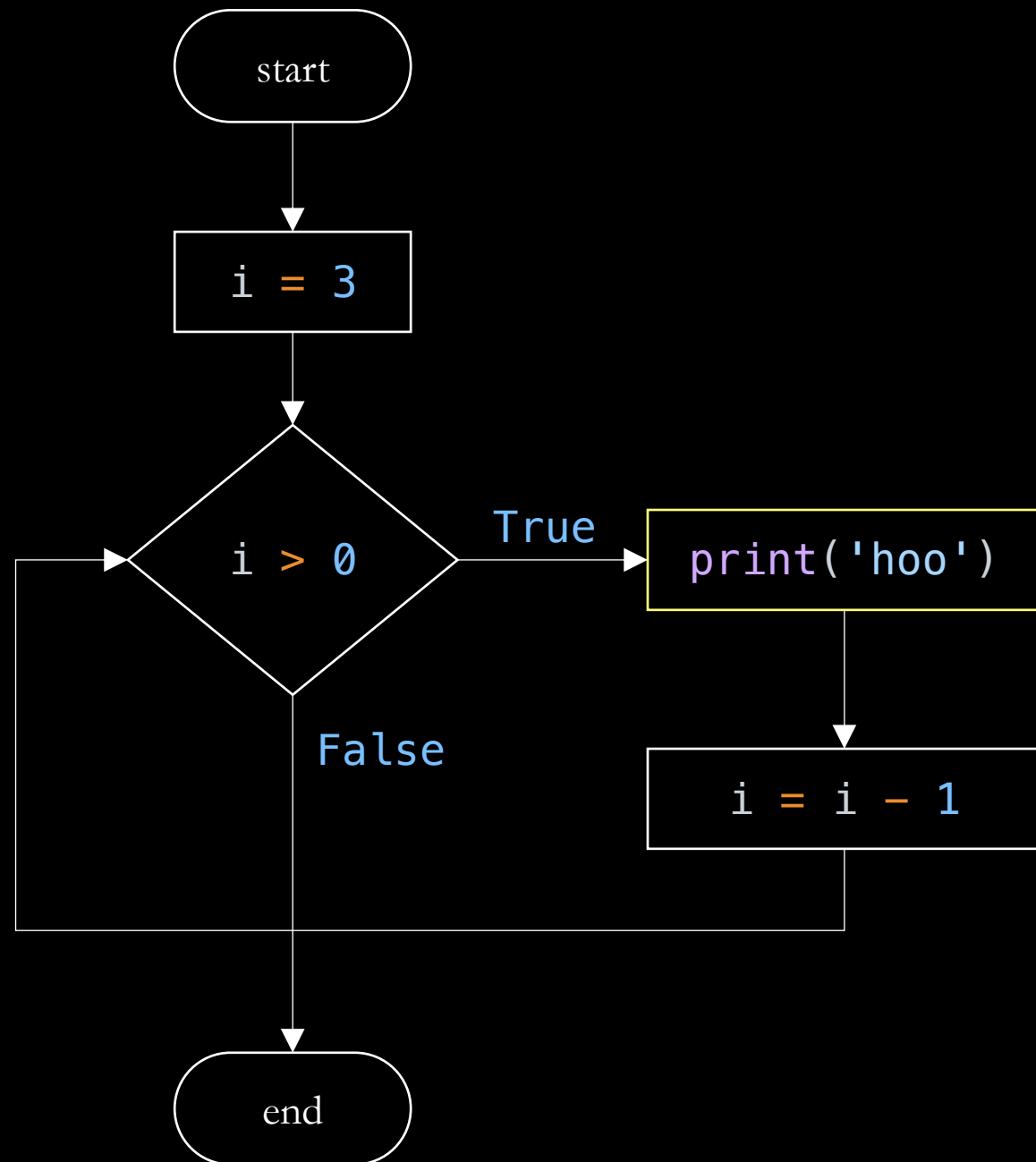
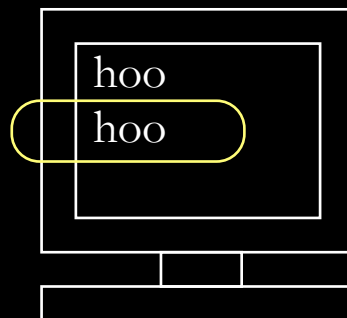
Objekter
3
2



→
`i = 3`
`while i > 0:`
 `print('hoo')`
 `i = i - 1`

Variabler
i

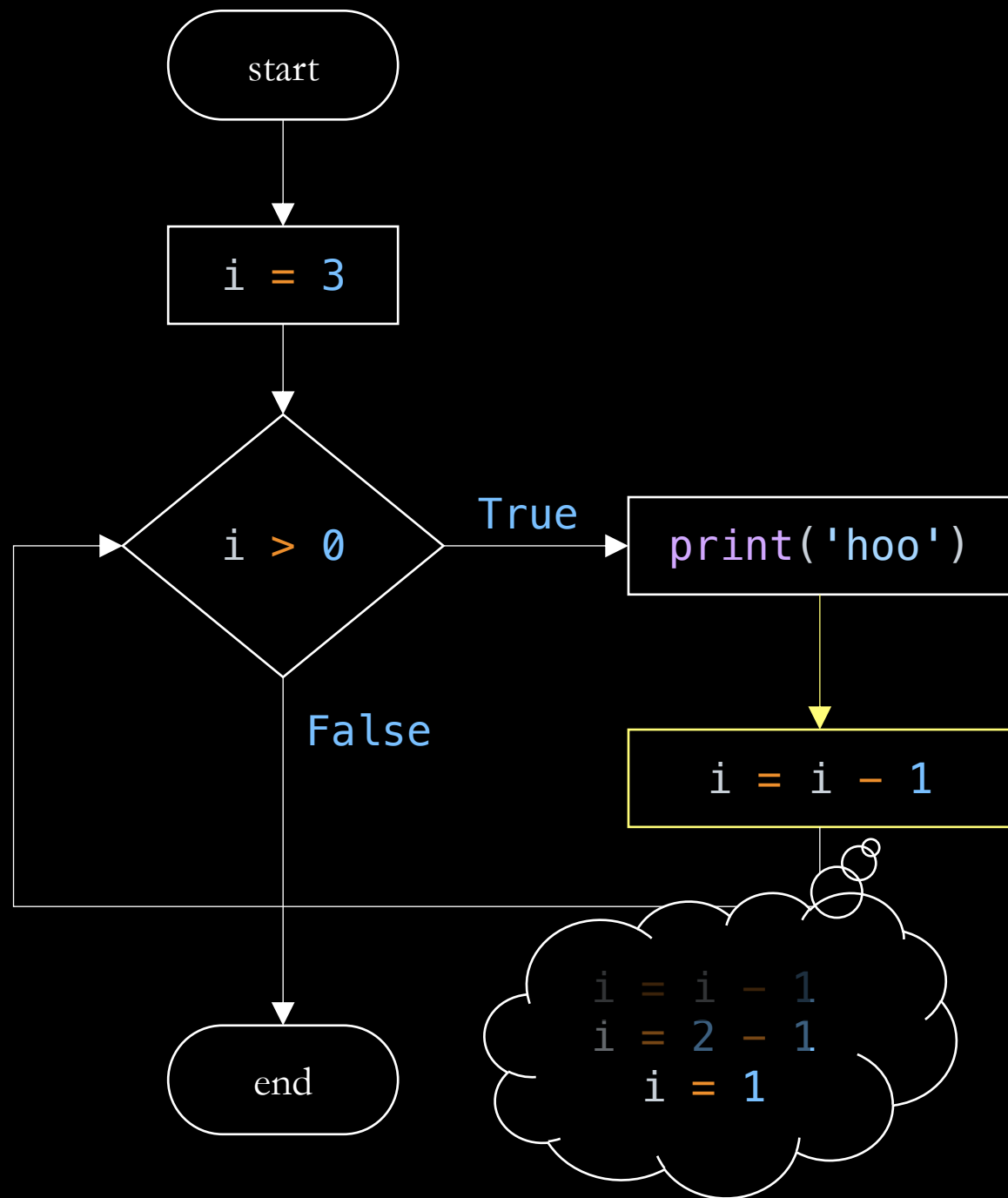
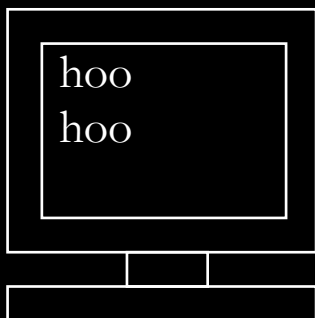
Objekter
3
2



→
`i = 3`
`while i > 0:`
 `print('hoo')`
 `i = i - 1`

Variabler
i

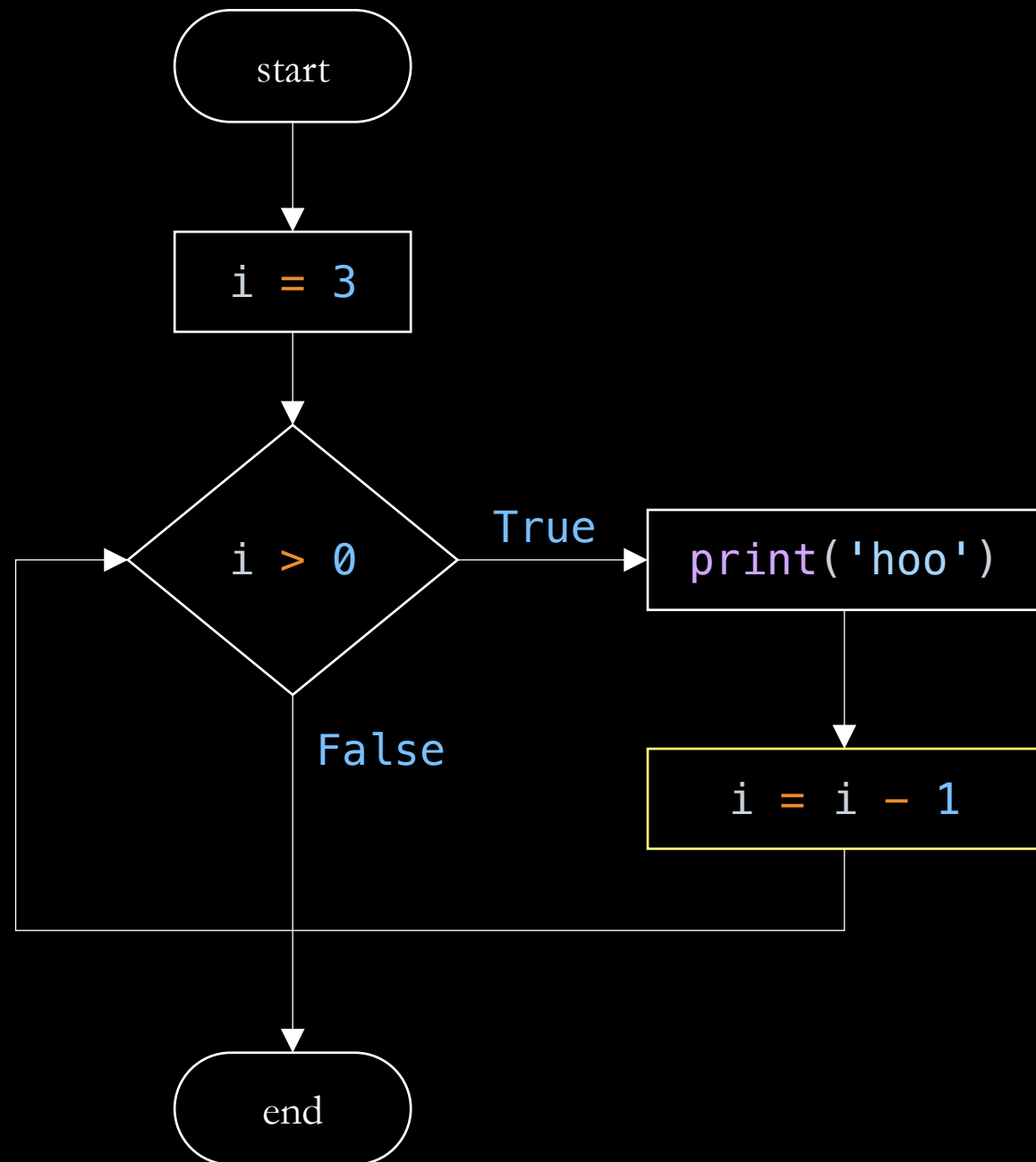
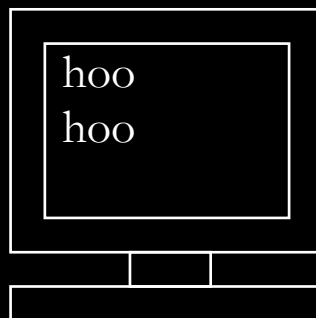
Objekter
3
2



→

```
i = 3
while i > 0:
    print('hoo')
    i = i - 1
```

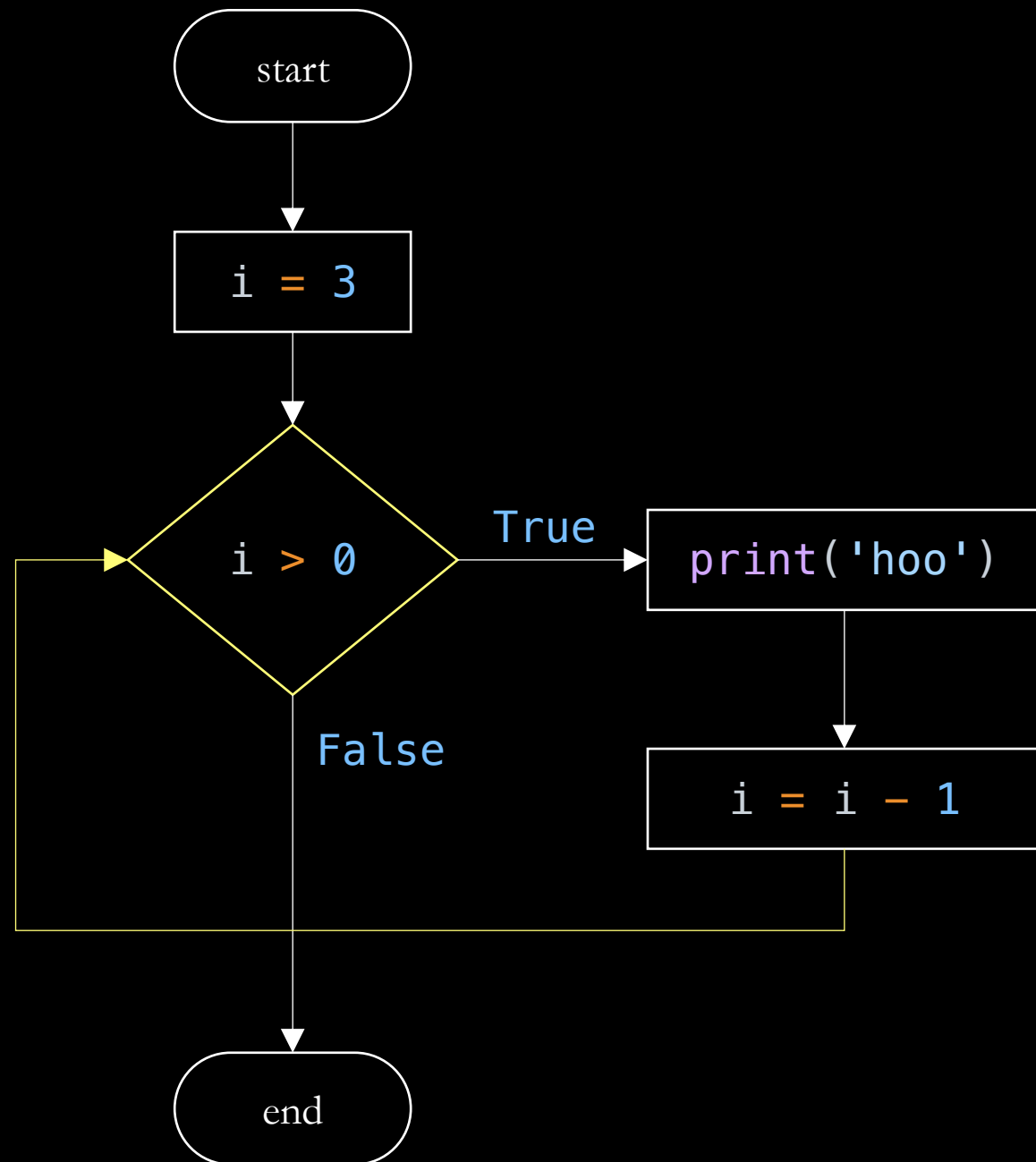
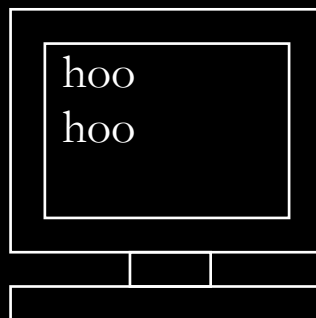
Variabler	Objekter
i	3
	2
	1



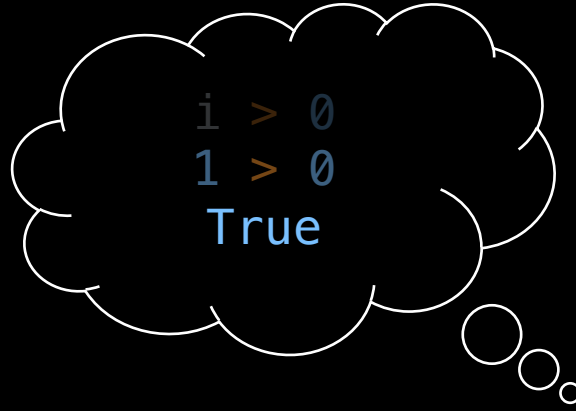
→ `i = 3`
`while i > 0:`
 `print('hoo')`
 `i = i - 1`

Variabler
i

Objekter
3
2
1

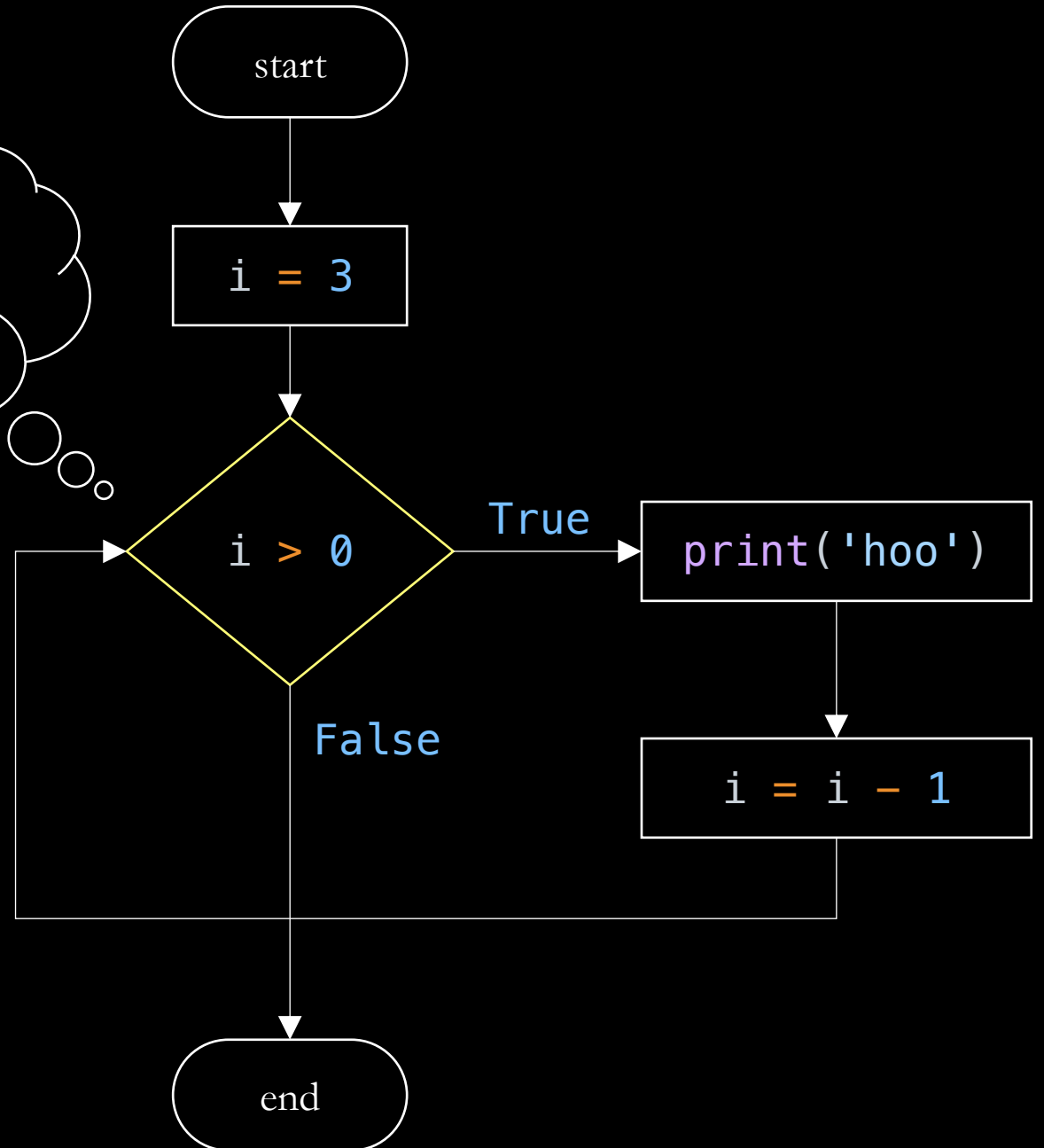
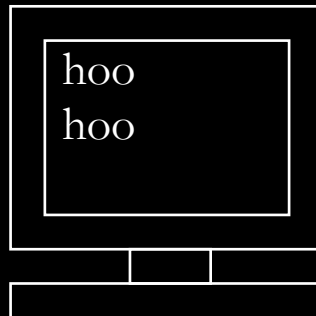


→ `i = 3`
`while i > 0:`
 `print('hoo')`
 `i = i - 1`

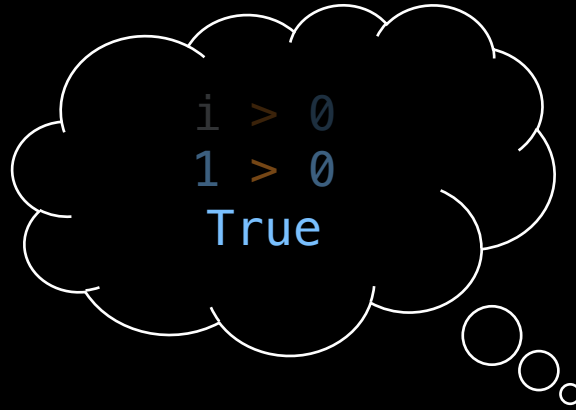


Variabler
i

Objekter
3
2
1

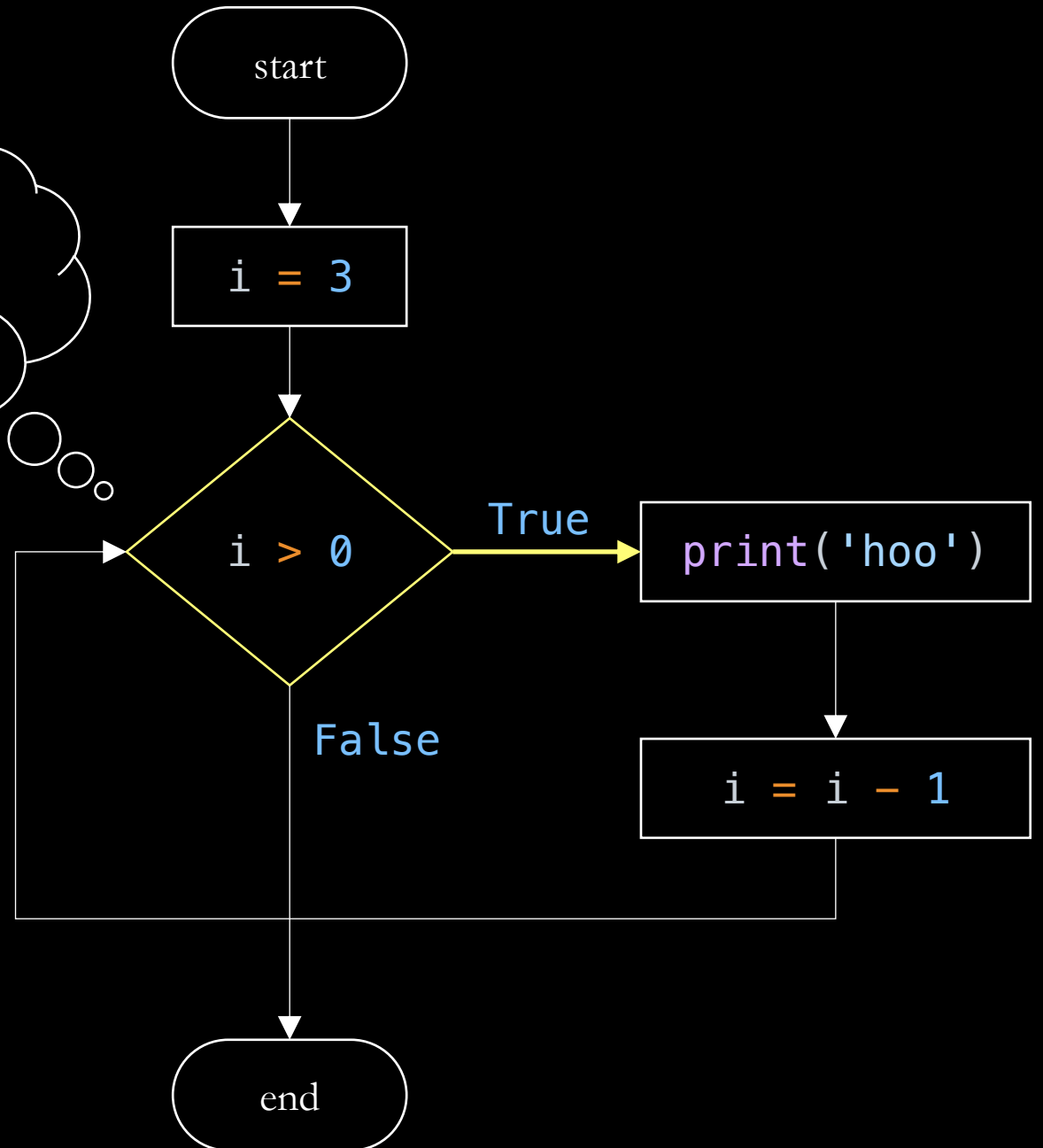
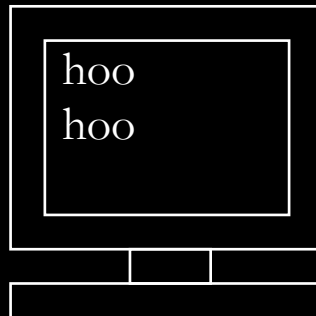


→ `i = 3`
`while i > 0:`
 `print('hoo')`
 `i = i - 1`



Variabler
i

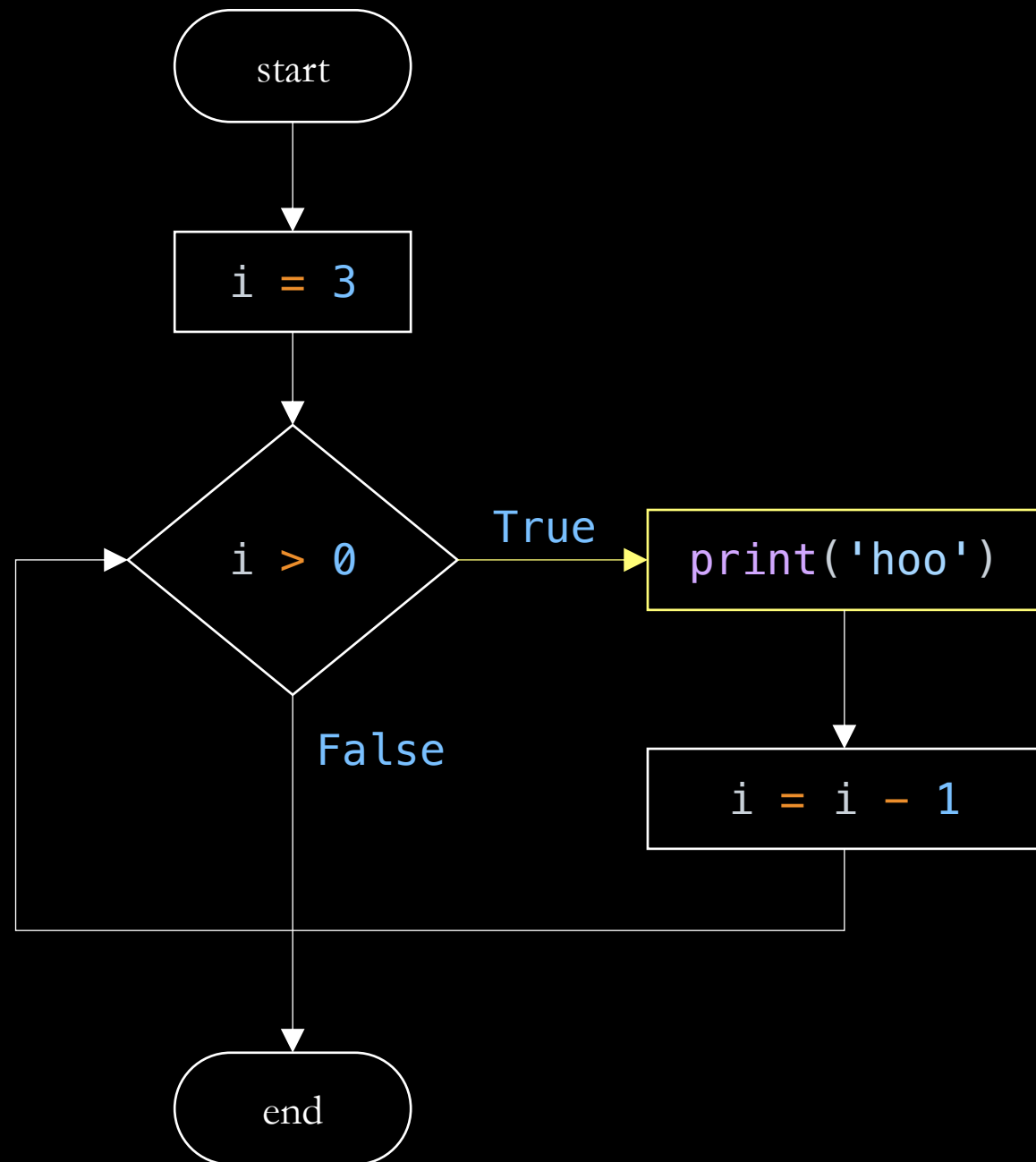
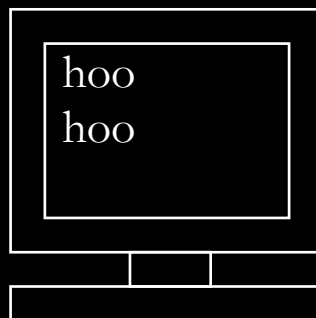
Objekter
3
2
1



→
`i = 3`
`while i > 0:`
 `print('hoo')`
 `i = i - 1`

Variabler
i

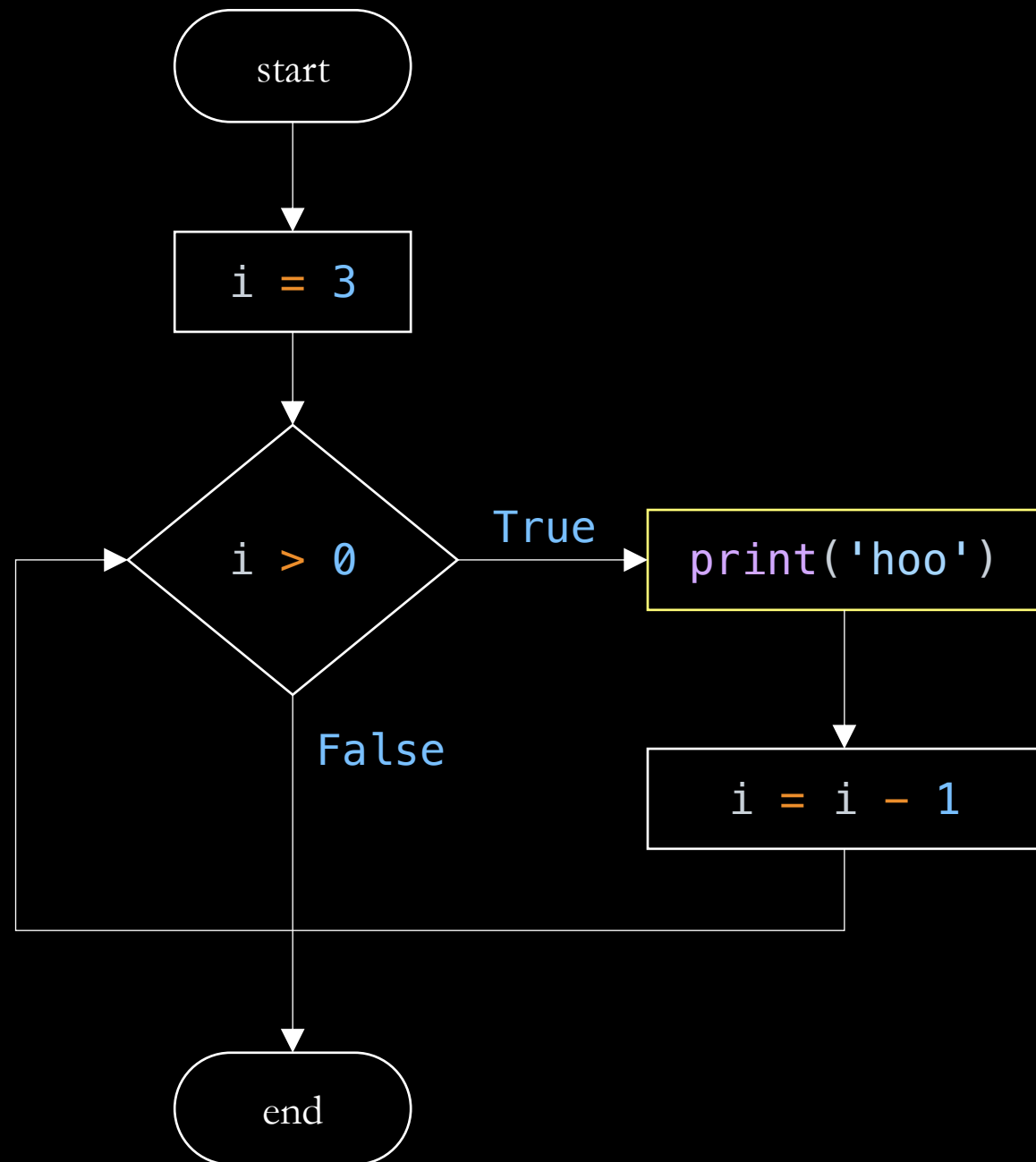
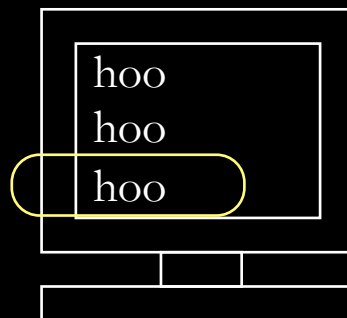
Objekter
3
2
1



→
`i = 3`
`while i > 0:`
 `print('hoo')`
 `i = i - 1`

Variabler
i

Objekter
3
2
1

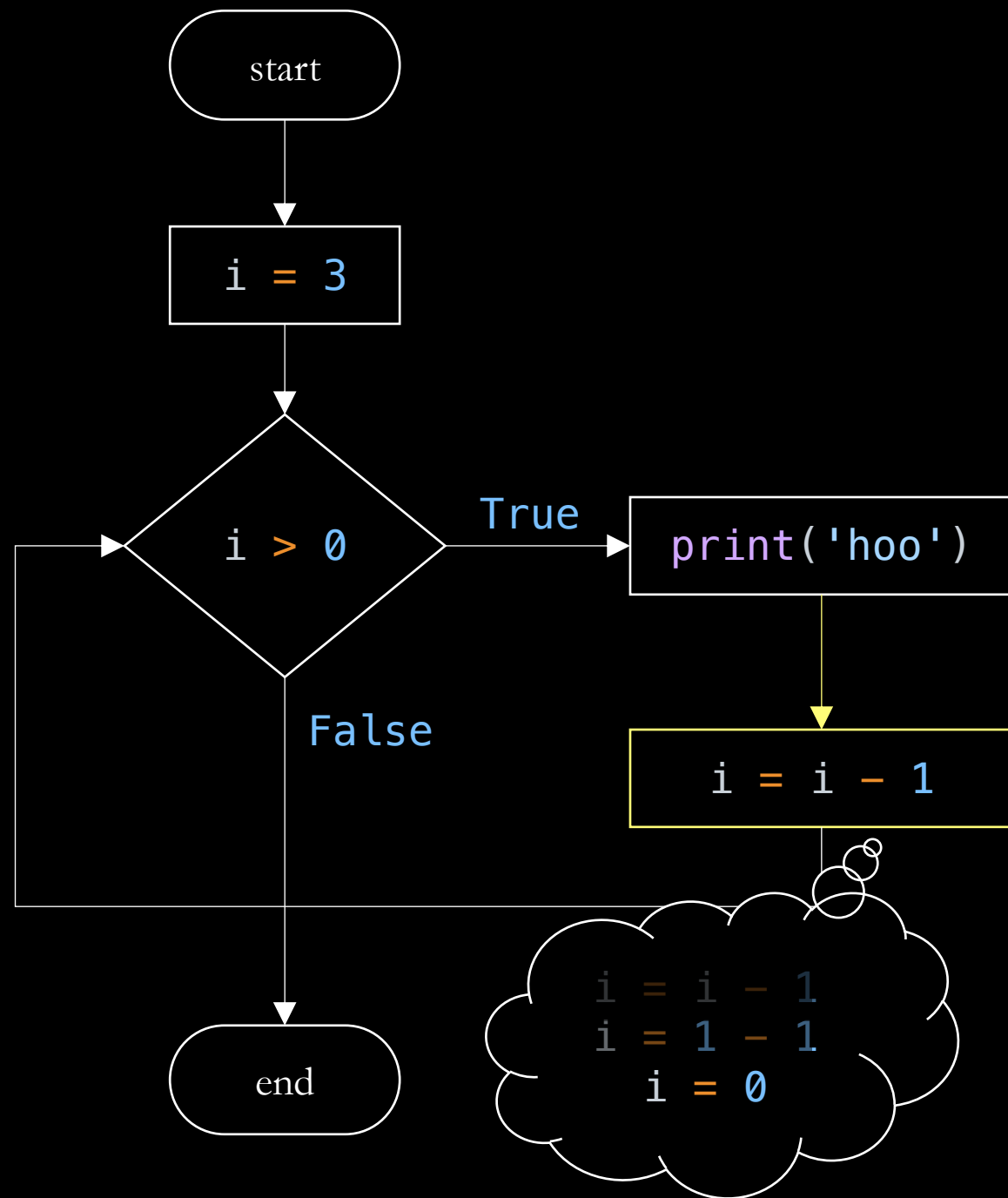
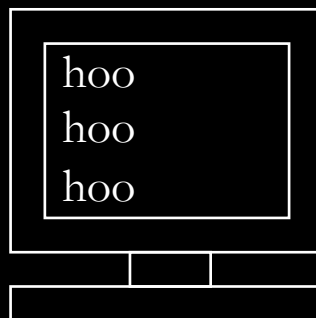


→

```
i = 3
while i > 0:
    print('hoo')
    i = i - 1
```

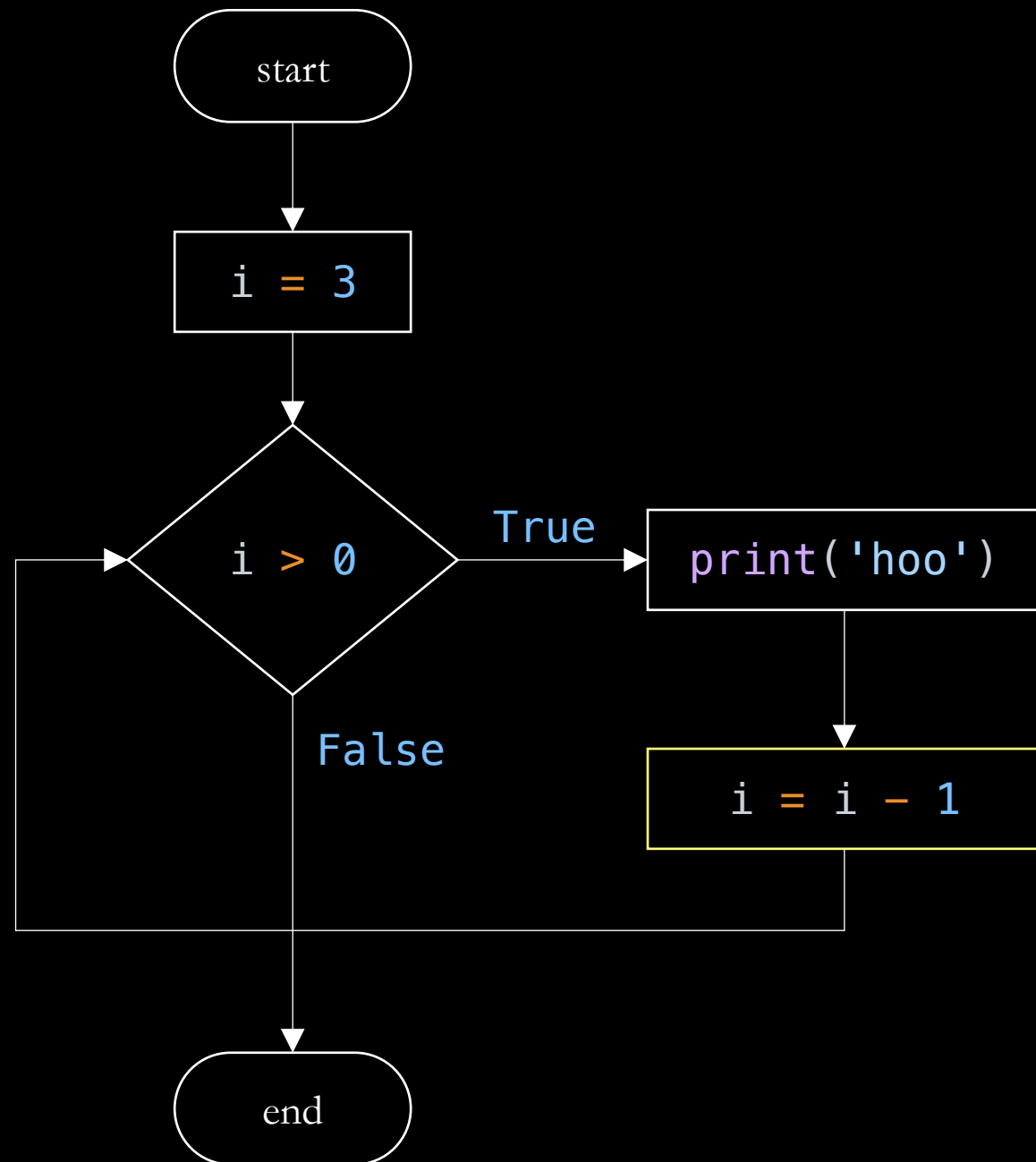
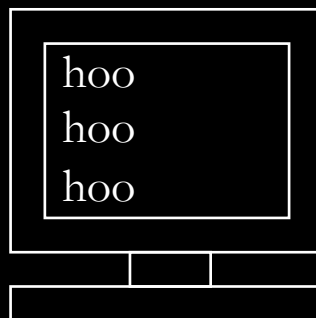
Variabler
i

Objekter
3
2
1



→
`i = 3`
`while i > 0:`
 `print('hoo')`
 `i = i - 1`

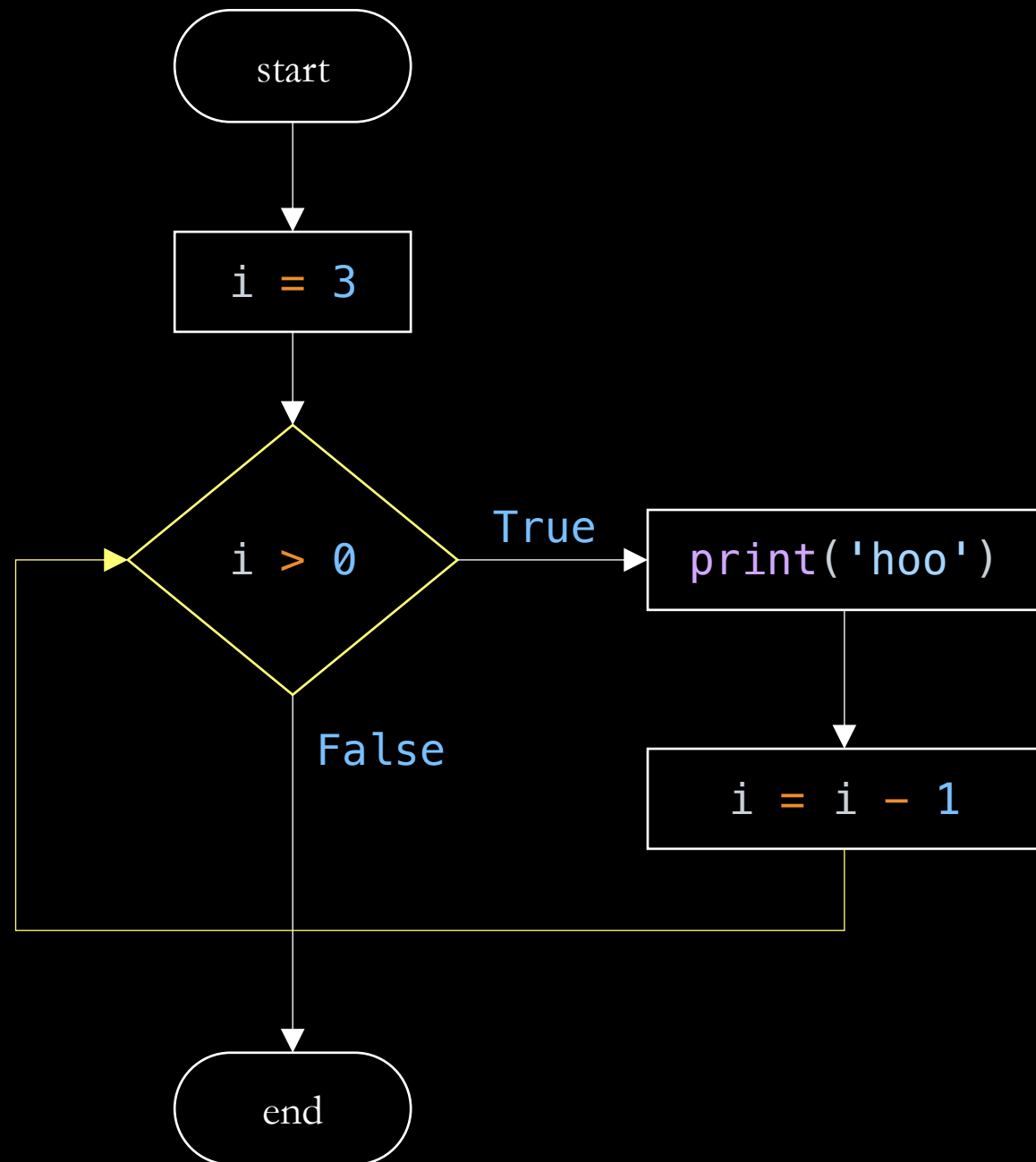
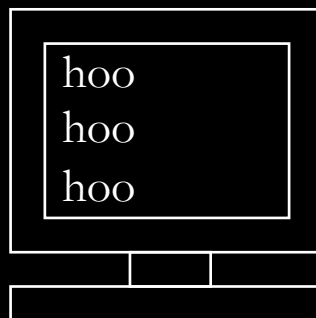
Variabler		Objekter
i	→	0
		2
		1



→ `i = 3`
`while i > 0:`
 `print('hoo')`
 `i = i - 1`

Variabler
i

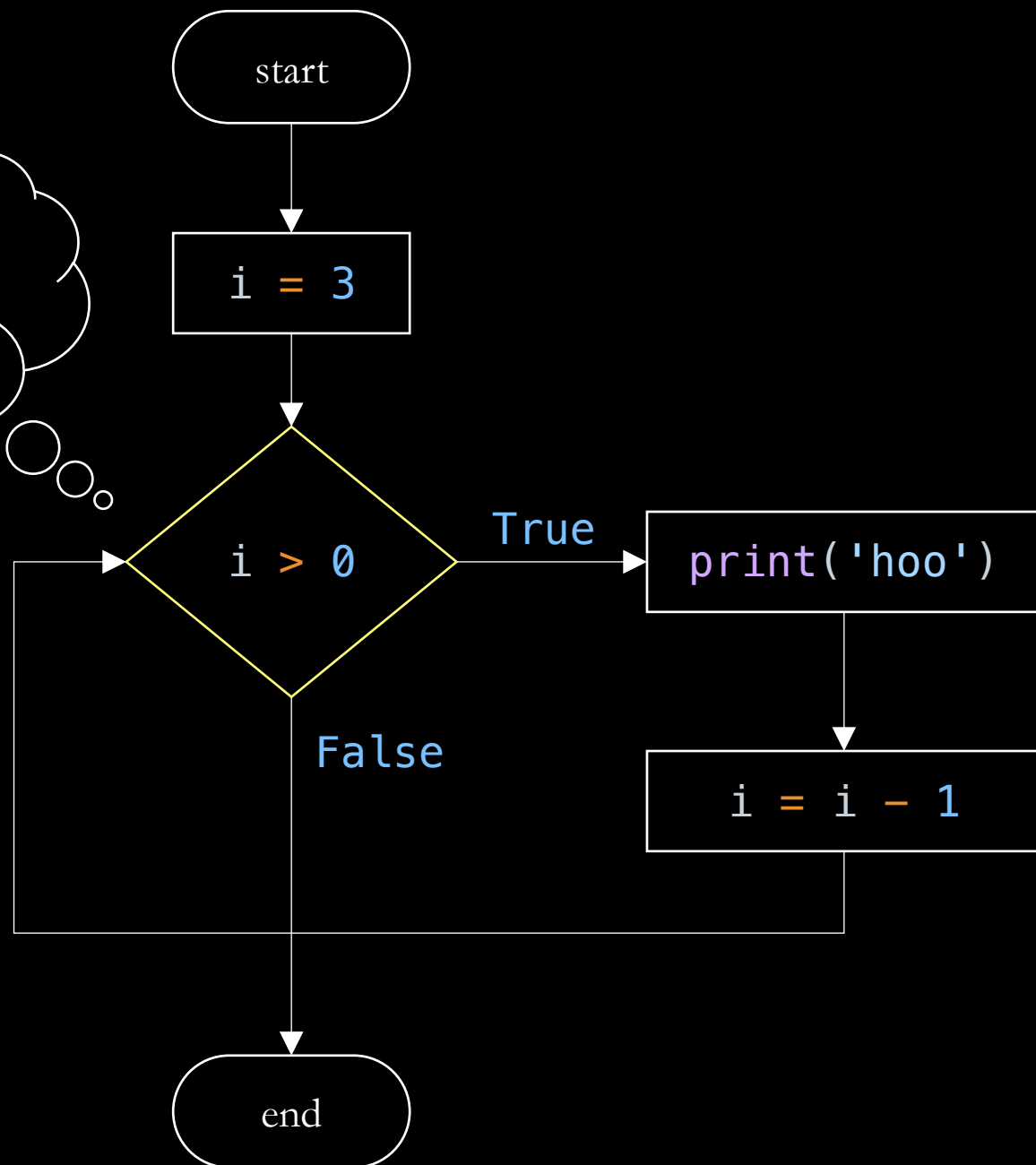
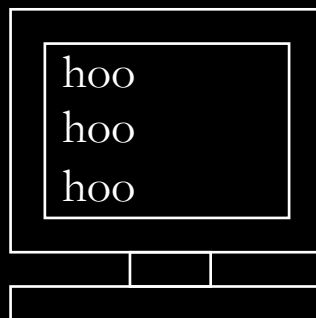
Objekter
0
2
1



→ `i = 3`
`while i > 0:`
 `print('hoo')`
 `i = i - 1`

`i > 0`
`0 > 0`
`False`

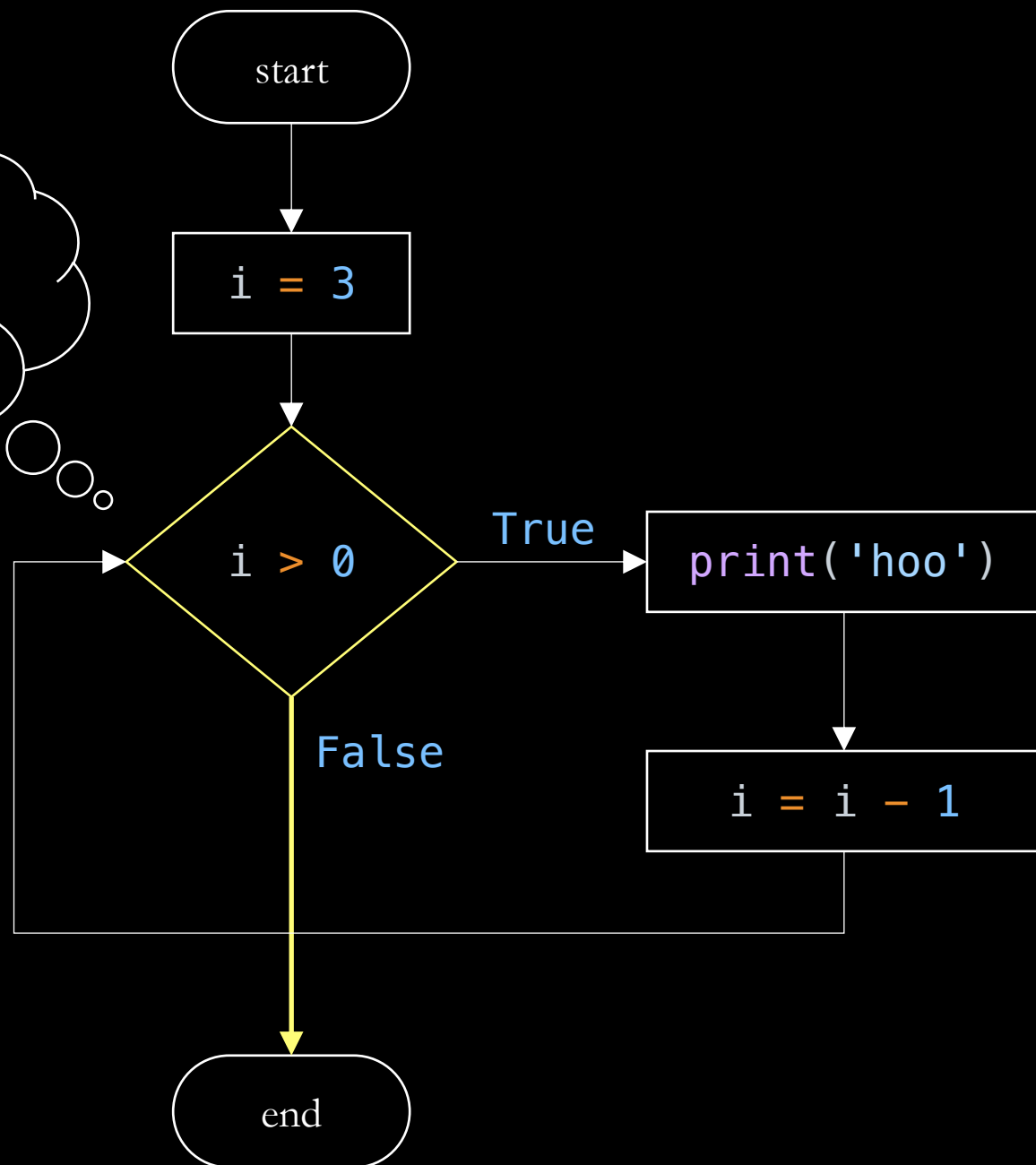
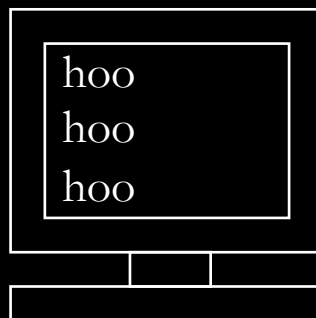
Variabler	Objekter
i	0
	2
	1



→ `i = 3`
`while i > 0:`
 `print('hoo')`
 `i = i - 1`

`i > 0`
`0 > 0`
`False`

Variabler	Objekter
i	0
	2
	1

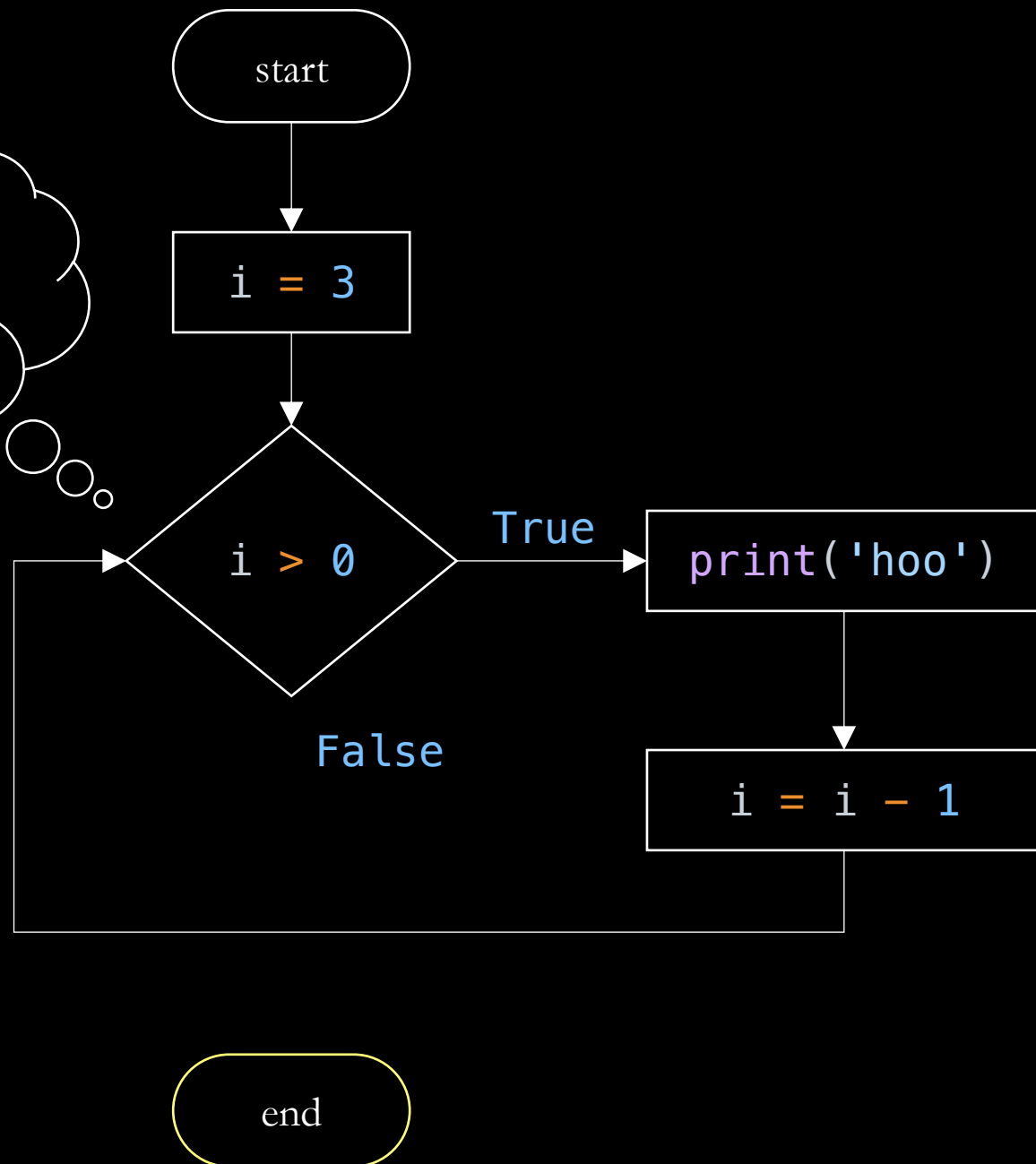
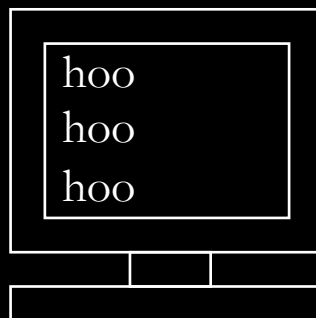


```
i = 3
while i > 0:
    print('hoo')
    i = i - 1
```

`i > 0`
`0 > 0`
False

Variabler
i

Objekter
0
2
1



```
i = 3
while i > 0:
    print('hoo')
    i = i - 1
```



```
for _ in range(3):
    print('hoo')
```

```
range(5)
```

```
0, 1, 2, 3, 4
```

```
range(3, 7)
```

```
3, 4, 5, 6
```

```
range(10, 20, 2)
```

```
10, 12, 14, 16, 18
```

```
range(20, 10, -2)
```

20, 18, 16, 14, 12

for

while

kan vare evig



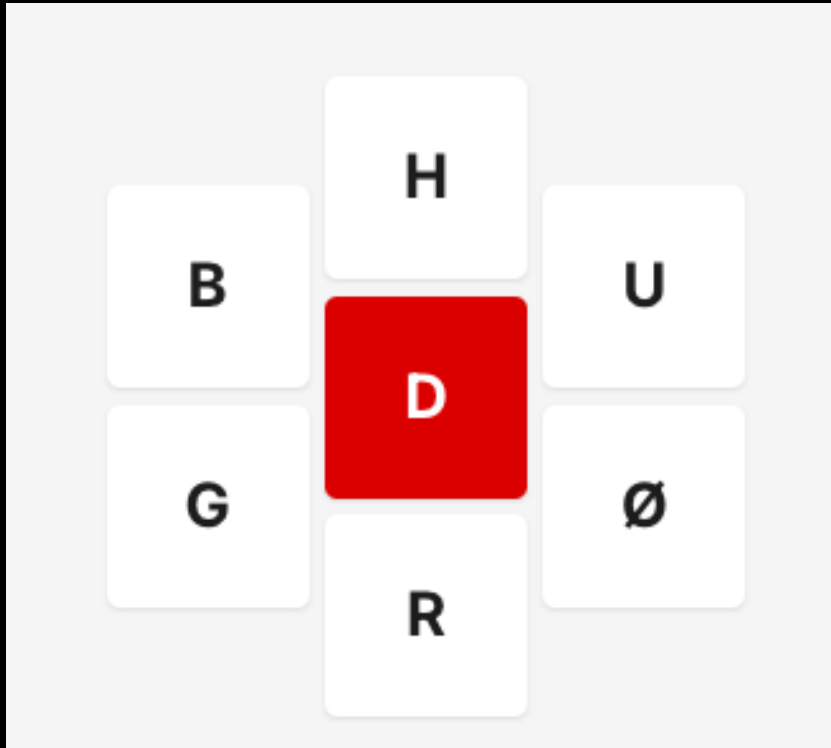
kortere å skrive



mindre bugs

SPILLE ORDSTJERNE

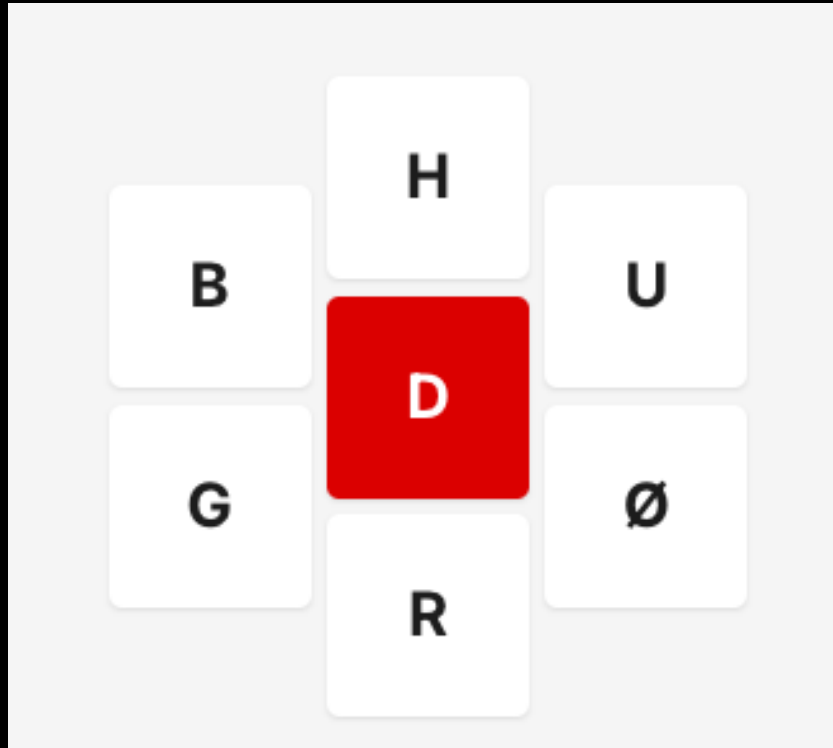
ORDSTJERNE



vg.no/spill

- Finn så mange ord du kan
- Regler:
 - Alle ord må inneholde bokstaven i midten
 - Du kan bruke samme bokstav flere ganger
 - Ordene må ha minst fire bokstaver

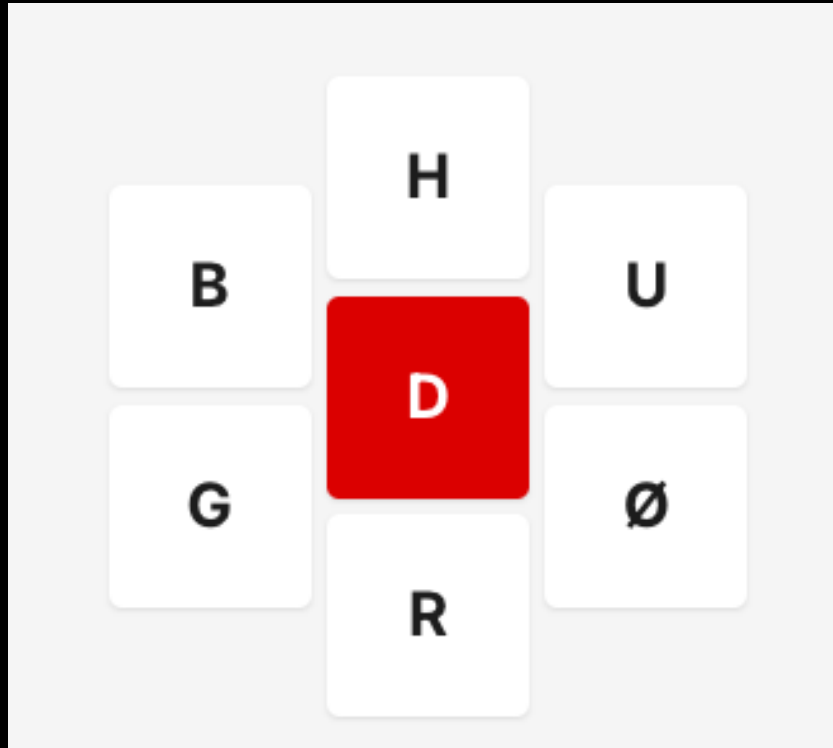
ORDSTJERNE



- Finn så mange ord du kan
- Regler:
 - Alle ord må inneholde bokstaven i midten
 - Du kan bruke samme bokstav flere ganger
 - Ordene må ha minst fire bokstaver
- Steg 1: en funksjon som avgjør om ett ord er godkjent eller ikke

```
def is_legal_word(word, req_char, avail_chars):  
    ...
```

ORDSTJERNE



- Finn så mange ord du kan
- Regler:
 - Alle ord må inneholde bokstaven i midten
 - Du kan bruke samme bokstav flere ganger
 - Ordene må ha minst fire bokstaver
- Steg 1: en funksjon som avgjør om ett ord er godkjent eller ikke
- Steg 2: prøv alle mulige ord

ALLE MULIGE ORD



NORSK SCRABBLEFORBUND

Hjem Bli medlem Om oss og Scrabble **Ordliste** Regler Turneringer Engelsk Kontakt oss End

Ordliste og søkeside

19. august 2023 publiserte vår dyktige språkkomite ny utgave av ordlista. Det ble lagt til 22 431 ord, og fjernet 560.

[Klikk her for å laste ned en zippet \(pakket\) utgave av hele lista.](#)

nsf2022.txt X

05_lister > nsf2022.txt

1	a
2	ab
3	abaca
4	abacaen
5	abacaene
6	abacaer
7	abaki
8	abakien
9	abakiene
10	abakier
11	abakus
12	abakusen
13	abakusene
14	abakuser
15	abalienasjon
16	abalienasjonen
17	abalienasjonene

ALLE MULIGE ORD

inf100.ii.uib.no/notat/strenger/#split-og-join

Split og join

Split kan brukes for å klippe opp en streng i biter, mens join kan brukes for å lime sammen biter til én streng.

```
# .split() deler opp en streng i biter, og legger bitene i en liste
names = 'Marshall,Rubble,Chase,Rocky,Zuma,Sky'
names_list = names.split(',')
print(names)
print(names_list) # ['Marshall', 'Rubble', 'Chase', 'Rocky', 'Zuma', 'Sky']
```

```
def get_all_possible_words():
    content = Path('nsf2025.txt').read_text(encoding='utf-8')
    return content.split('\n')
```

```
# Lese fra fil
name_of_file = 'foo.txt'
content = Path(name_of_file).read_text(encoding='utf-8')
```

Kopier

Ordliste

19. august 2025
431 ord, og

[Klikk her for](#)

ier

us

usen

usene

user

15 abalienasjon

16 abalienasjonen

17 abalienasjonene

PRØV ALLE MULIGE ORD

opprettet tom liste



```
legal_words = []  
for candidate_word in list_of_words:  
    if word_is_legal(candidate_word, required, available):  
        legal_words.append(candidate_word)
```

løkke over alle mulige ord



legger til candidate_word på slutten av legal_words



PRØV ALLE MULIGHETER

opprettet tom liste



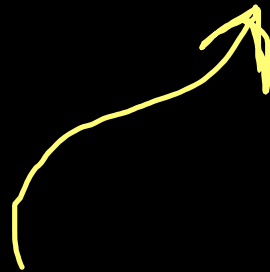
løkke over alle potensielle muligheter



```
filtered_things = []  
for candidate_thing in collection_of_everything:  
    if matches_criteria(candidate_thing):  
        filtered_things.append(candidate_thing)
```



*sjekk om tingen er
en sann ting vi vil ha*

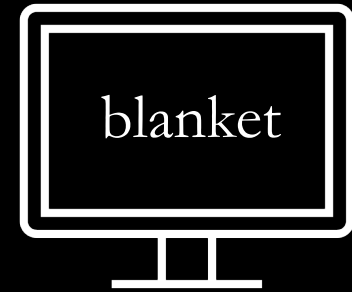


legg til candidate_thing på slutten av filtered_things

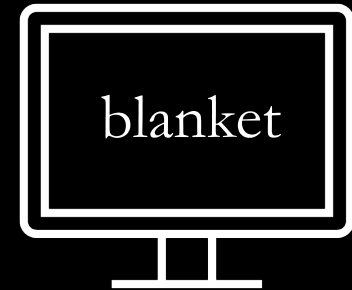
RECAP:

INDEKSERING OPPSLAG

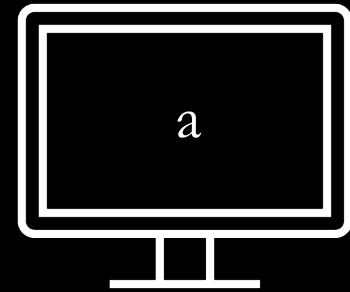
```
my_things = ['teddy', 'blanket', 'fish']  
x = my_things[1]  
print(x)
```



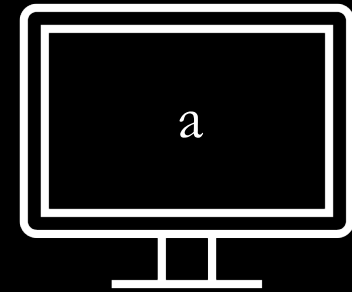
```
my_things = ['teddy', 'blanket', 'fish']  
x = my_things[-2]  
print(x)
```



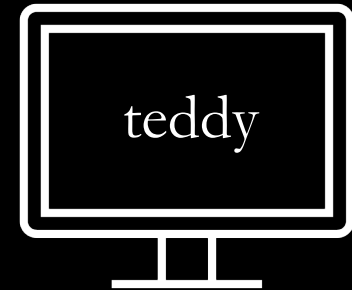
```
my_things = ['teddy', 'blanket', 'fish']  
x = my_things[1]  
y = x[2]  
print(y)
```



```
my_things = ['teddy', 'blanket', 'fish']  
y = my_things[1][2]  
  
print(y)
```

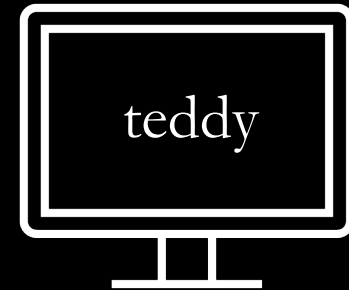


```
everyones_things = [  
    ['Torstein', 'teddy', 'blanket', 'fish'],  
    ['Linn Astrid', 'cat', 'oven', 'vegetables'],  
    ['Gutama', 'duck', 'jacket', 'gummy bears'],  
    ['Amy', 'kitten', 'lighter', 'gummy bears'],  
    ['Sasha', 'panda', 'wool', 'fish'],  
    ['Hilde', 'teddy', 'jacket', 'fish'],  
]  
my_things = everyones_things[0]  
my_thing = my_things[1]  
print(my_thing)
```



```
everyones_things = [  
    ['Torstein', 'teddy', 'blanket', 'fish'],  
    ['Linn Astrid', 'cat', 'oven', 'vegetables'],  
    ['Gutama', 'duck', 'jacket', 'gummy bears'],  
    ['Amy', 'kitten', 'lighter', 'gummy bears'],  
    ['Sasha', 'panda', 'wool', 'fish'],  
    ['Hilde', 'teddy', 'jacket', 'fish'],  
]  
my_thing = everyones_things[0][1]  
print(my_thing)
```

↑ ↑
«rad»
«kolonne»



BESKJÆRING

- Som å indeksere; man får ikke (nødvendigvis) ett element, men et utklipp

'abcdef'[1]



'b'

Indeksering

fra og med opp til
'abcdef'[1:4]



'bcd'

Beskjæring

BESKJÆRING

- Som å indeksere; vi får ikke (nødvendigvis) ett element, men et utklipp

`'abcdef'[1]`



`'b'`

Indeksering

fra og med *opp til*

`'abcdef'[1:4]`



`'bcd'`

Beskjæring

hvis start mangler, anta 0

`[42, 95, 77, 22][0:2]`

`[42, 95]`

`[42, 95, 77, 22][:2]`

`[42, 95]`

hvis slutt mangler, anta listens lengde

`[42, 95, 77, 22][1:]`

`[95, 77, 22]`

`[42, 95, 77, 22][1:2]`

`[95]`

hvis kun ett element, så er resultatet likevel en liste

JOBBE I FLERE FILER

foo.py

```
from bar import add

num1 = int(input('num1 = '))
num2 = int(input('num2 = '))

ans = add(num1, num2)
print(f'Added nums: {ans}')
```

```
$ python foo.py
Testing add... OK
Testing multiply... OK
num1 = 2
num2 = 2
Added nums: 4
```

bar.py

```
def add(a, b):
    return a + b

print('Testing add... ', end='')
assert 4 == add(2, 2)
print('OK')

def multiply(a, b):
    return a * b

print('Testing multiply... ', end='')
assert 6 == multiply(2, 3)
print('OK')
```

```
IF __NAME__ == "__MAIN__":
```

foo.py

```
from bar import add

num1 = int(input('num1 = '))
num2 = int(input('num2 = '))

ans = add(num1, num2)
print(f'Added nums: {ans}')
```

```
$ python foo.py
num1 = 2
num2 = 2
Added nums: 4
```

bar.py

```
def add(a, b):
    return a + b

def test_add():
    print('Testing add... ', end='')
    assert 4 == add(2, 2)
    print('OK')

def multiply(a, b):
    return a * b

def test_multiply():
    print('Testing multiply... ', end='')
    assert 6 == multiply(2, 3)
    print('OK')

if __name__ == '__main__':
    test_add()
    test_multiply()
```