

Loops and Iteration

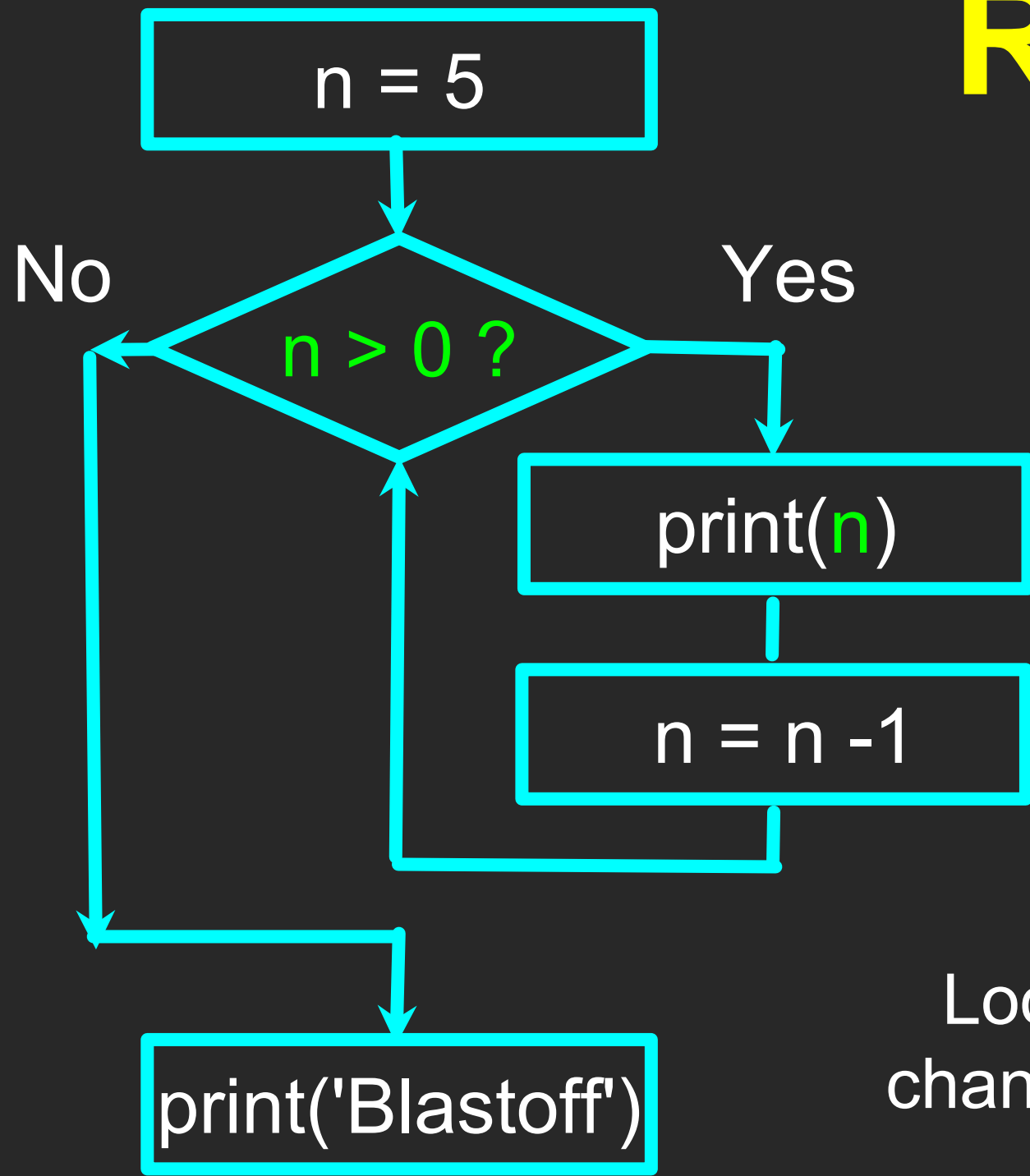
Chapter 5



Python for Everybody
www.py4e.com



Repeated Steps



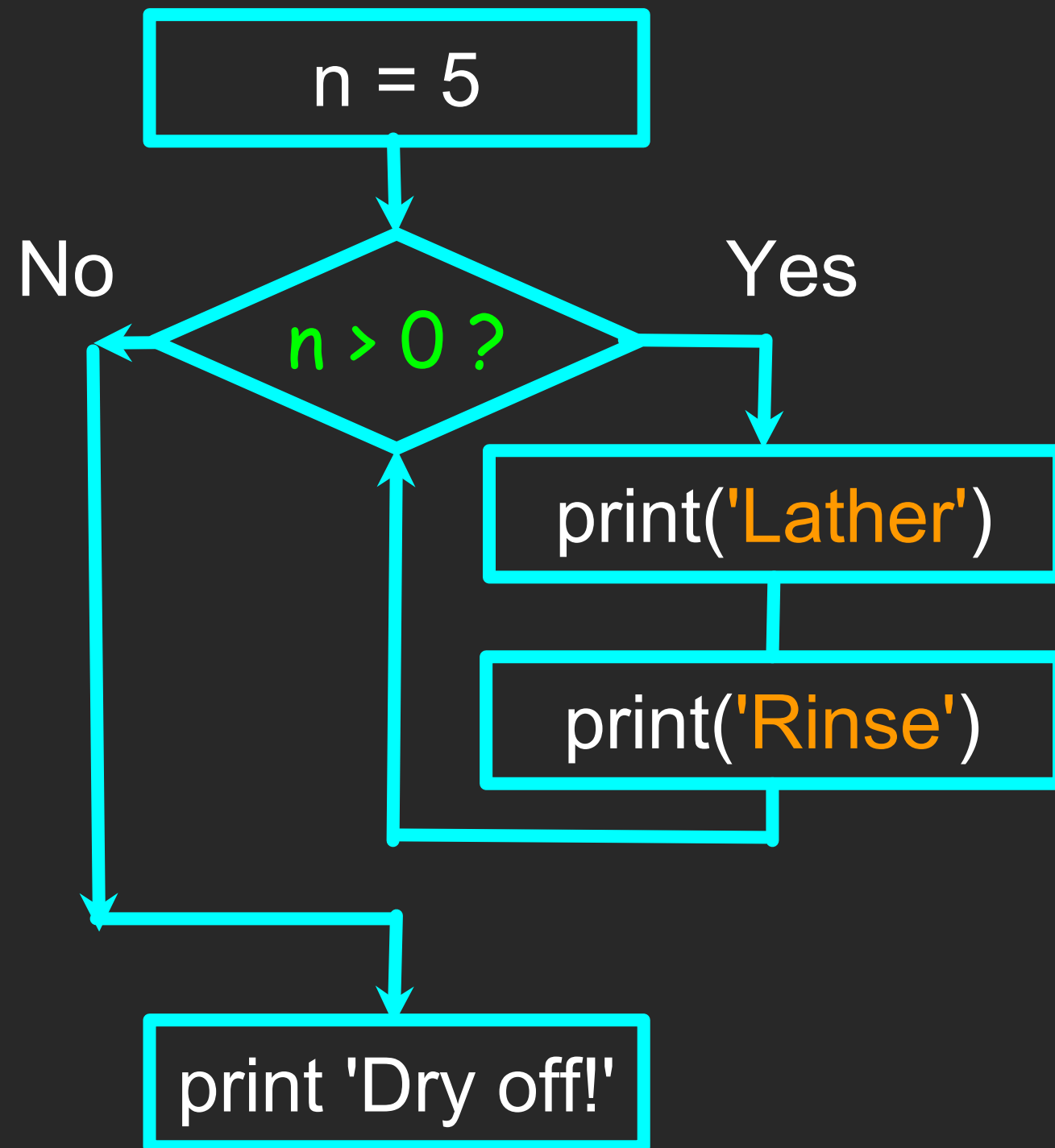
Program:

```
n = 5
while n > 0 :
    print(n)
    n = n - 1
print('Blastoff!')
print(n)
```

Output:

5
4
3
2
1
Blastoff!
0

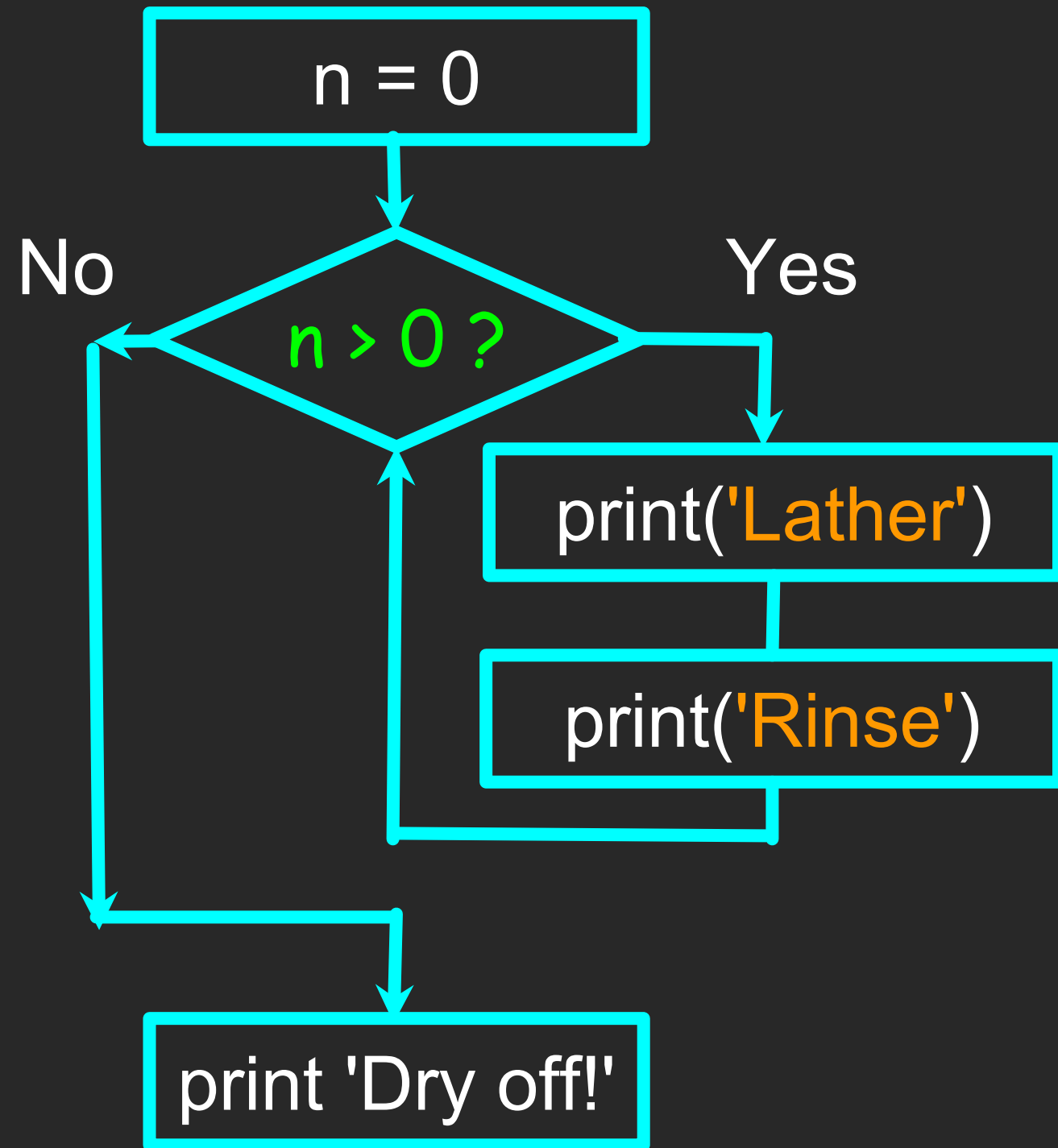
Loops (repeated steps) have **iteration variables** that change each time through a loop. Often these **iteration variables** go through a sequence of numbers.



An Infinite Loop

```
n = 5
while n > 0 :
    print('Lather')
    print('Rinse')
    print('Dry off!')
```

What is wrong with this loop?



Another Loop

```
n = 0
while n > 0 :
    print('Lather')
    print('Rinse')
    print('Dry off!')
```

What is this loop doing?

Breaking Out of a Loop

- The **break** statement ends the current loop and jumps to the statement immediately following the loop
- It is like a loop test that can happen anywhere in the body of the loop

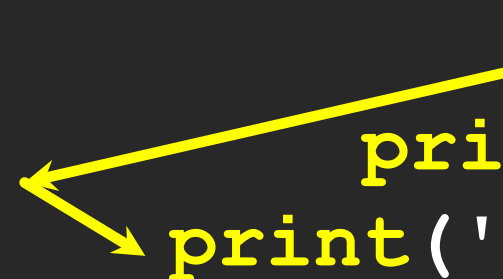
```
while True:
    line = input('> ')
    if line == 'done' :
        break
    print(line)
print('Done!')
```

```
> hello there
hello there
> finished
finished
> done
Done!
```

Breaking Out of a Loop

- The **break** statement ends the current loop and jumps to the statement immediately following the loop
- It is like a loop test that can happen anywhere in the body of the loop

```
while True:
    line = input('> ')
    if line == 'done' :
        break
    print(line)
print('Done!')
```

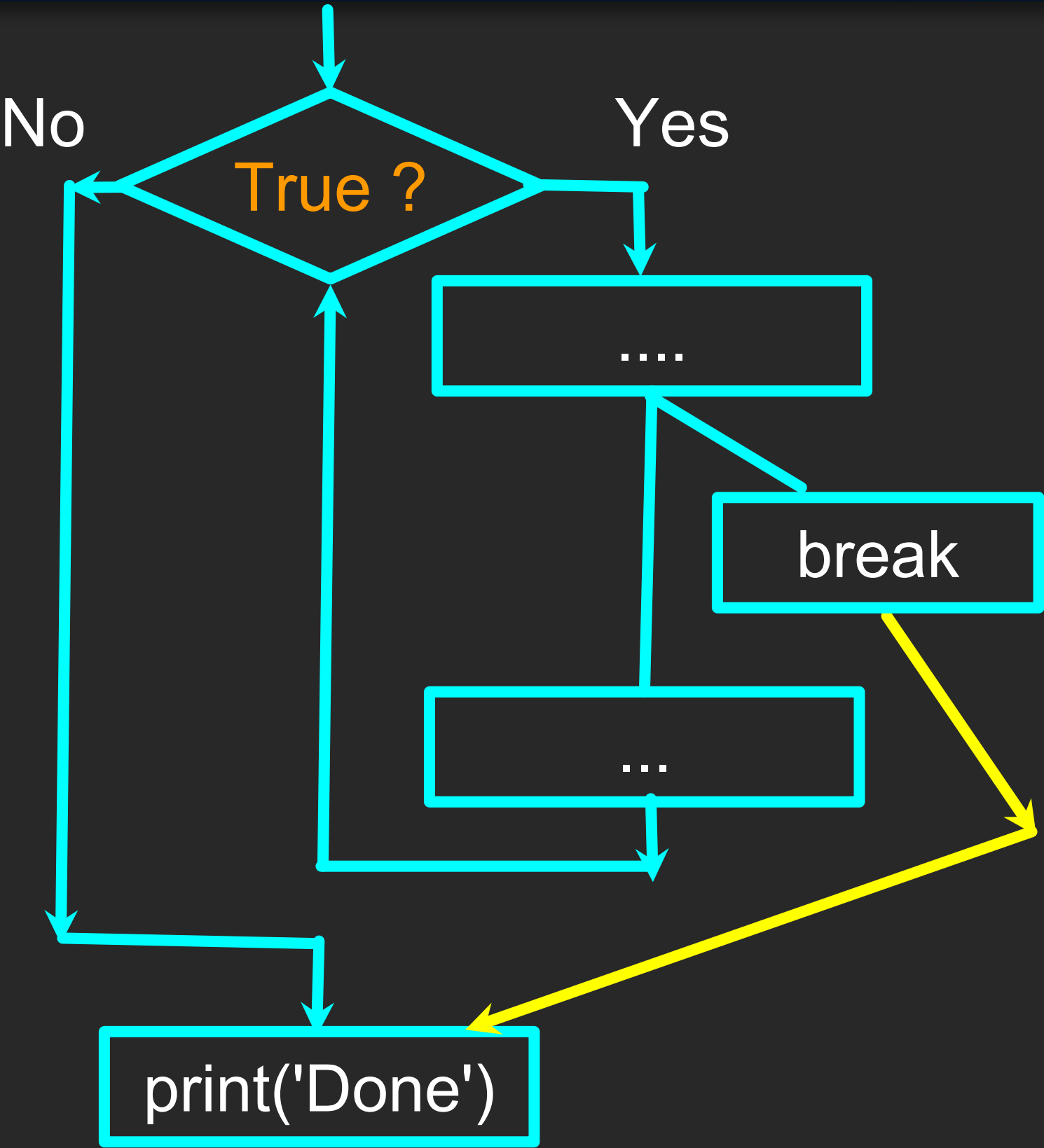


```
> hello there
hello there
> finished
finished
> done
Done!
```

```
while True:
    line = input('> ')
    if line == 'done' :
        break
    print(line)
print('Done!')
```



[http://en.wikipedia.org/wiki/Transporter_\(Star_Trek\)](http://en.wikipedia.org/wiki/Transporter_(Star_Trek))



Finishing an Iteration with Continue

The **continue** statement ends the current iteration and jumps to the top of the loop and starts the next iteration

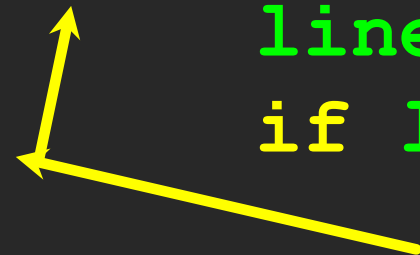
```
while True:
    line = input('> ')
    if line[0] == '#' :
        continue
    if line == 'done' :
        break
    print(line)
print('Done!')
```

```
> hello there
hello there
> # don't print this
> print this!
print this!
> done
Done!
```


Finishing an Iteration with Continue

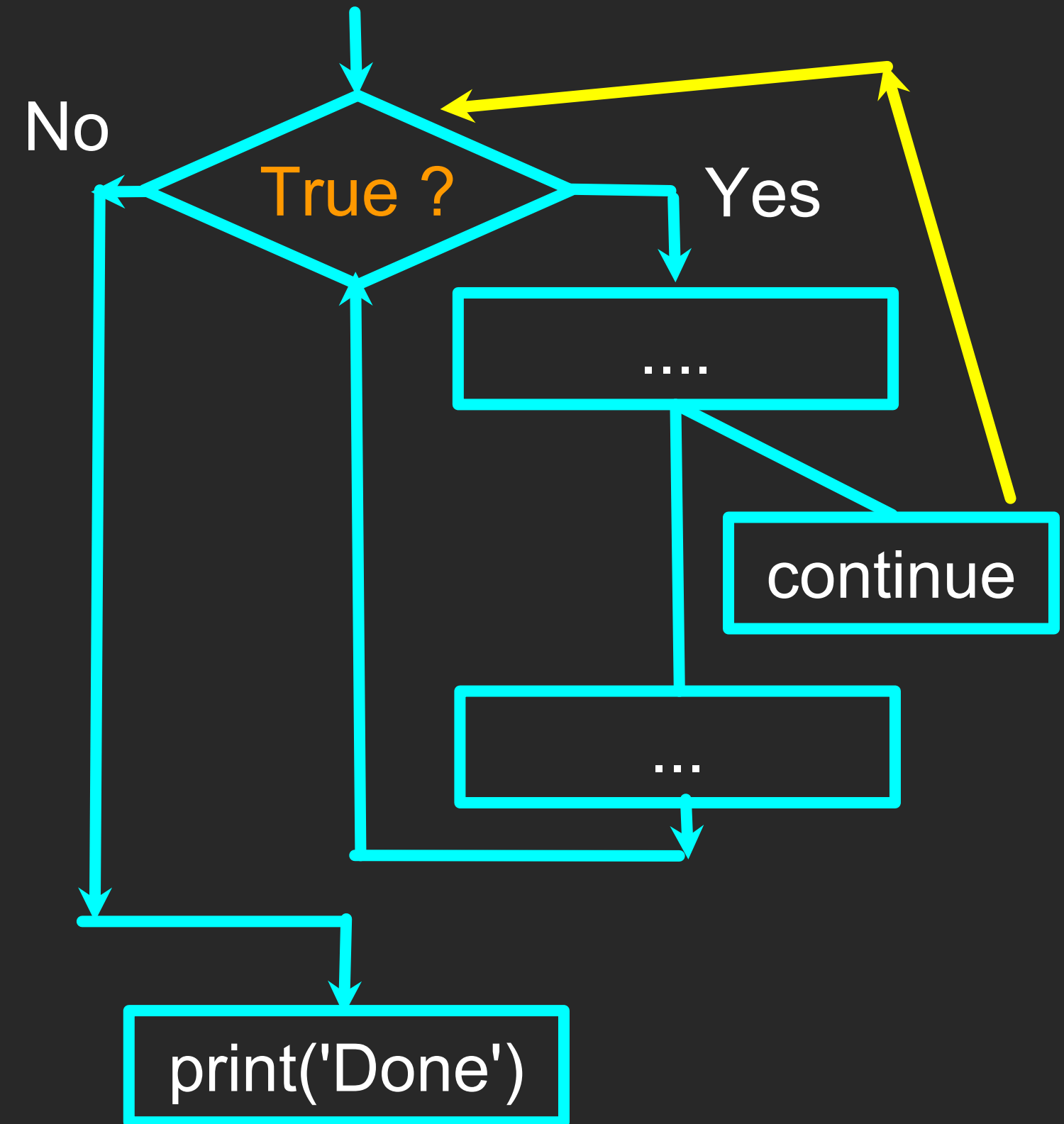
The **continue** statement ends the **current iteration** and jumps to the **top of the loop** and starts the next iteration

```
while True:
    line = input('> ')
    if line[0] == '#' :
        continue
    if line == 'done' :
        break
    print(line)
print('Done!')
```



```
> hello there
hello there
> # don't print this
> print this!
print this!
> done
Done!
```

```
while True:
    line = raw_input('> ')
    if line[0] == '#':
        continue
    if line == 'done':
        break
    print(line)
print('Done!')
```



Indefinite Loops

- While loops are called “indefinite loops” because they keep going until a logical condition becomes **False**
- The loops we have seen so far are pretty easy to examine to see if they will terminate or if they will be “infinite loops”
- Sometimes it is a little harder to be sure if a loop will terminate

Definite Loops



Acknowledgements / Contributions



These slides are Copyright 2010- Charles R. Severance
(www.dr-chuck.com) of the University of Michigan School of
Information and open.umich.edu and made available under a
Creative Commons Attribution 4.0 License. Please maintain this
last slide in all copies of the document to comply with the
attribution requirements of the license. If you make a change,
feel free to add your name and organization to the list of
contributors on this page as you republish the materials.

Initial Development: Charles Severance, University of Michigan
School of Information

... Insert new Contributors and Translators here

...