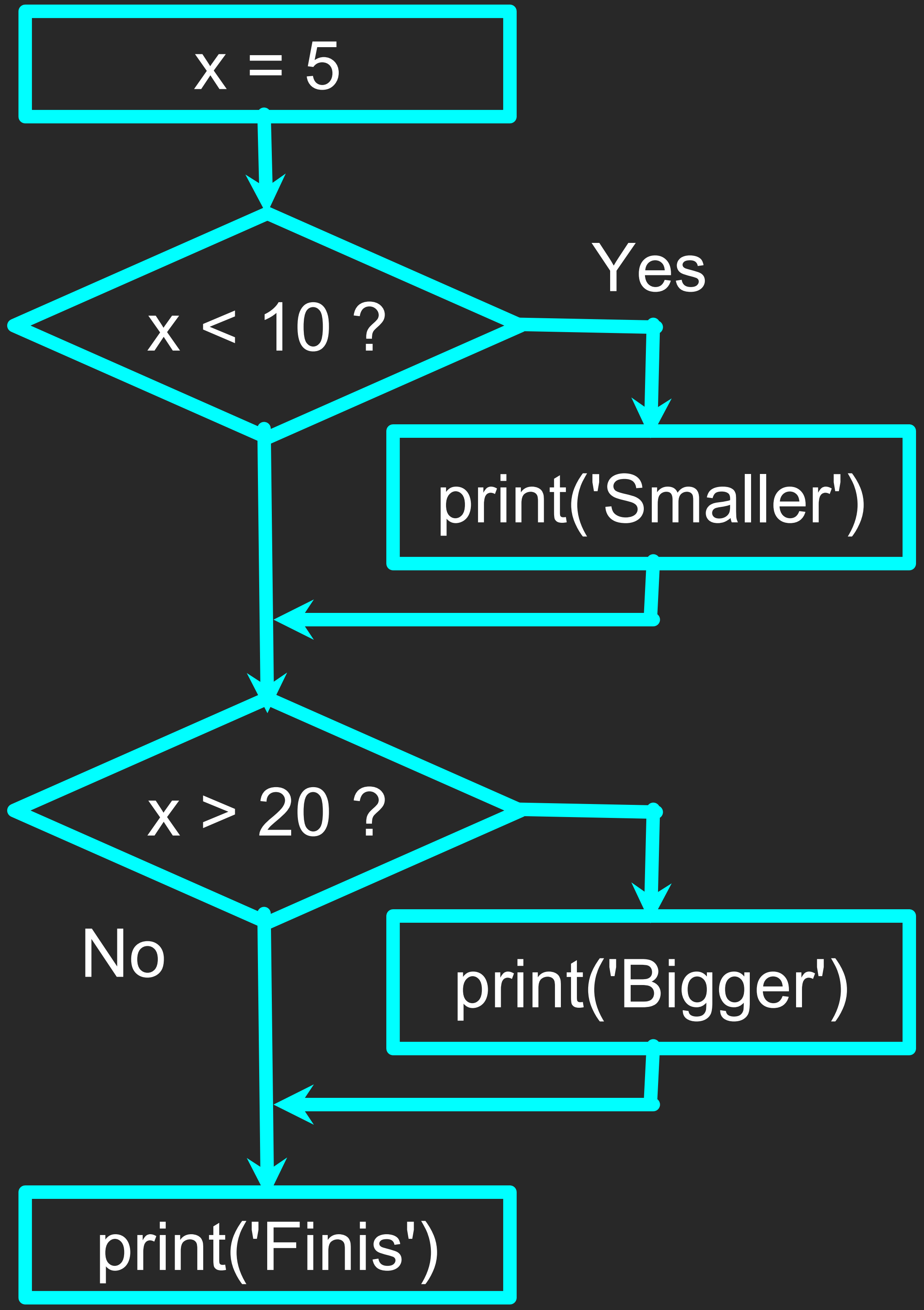
# Conditional Execution

## Chapter 3

Python for Everybody

www.py4e.com

# Conditional Steps



x = 5

x < 10 ?

Yes

print('Smaller')

x > 20 ?

No

print('Bigger')

print('Finis')

| Program | Output |
|---|---|
| `x = 5` | |
| `if x < 10:`<br>`    print('Smaller')` | Smaller |
| `if x > 20:`<br>`    print('Bigger')` | |
| `print('Finis')` | Finis |

# Comparison Operators

- Boolean expressions ask a question and produce a Yes or No result which we use to control program flow
- Boolean expressions using comparison operators evaluate to True / False or Yes / No
- Comparison operators look at variables but do not change the variables

| Python | Meaning |
|---|---|
| < | Less than |
| <= | Less than or Equal to |
| == | Equal to |
| >= | Greater than or Equal to |
| > | Greater than |
| != | Not equal |

Remember: “=” is used for assignment.

http://en.wikipedia.org/wiki/George_Boole

# Comparison Operators

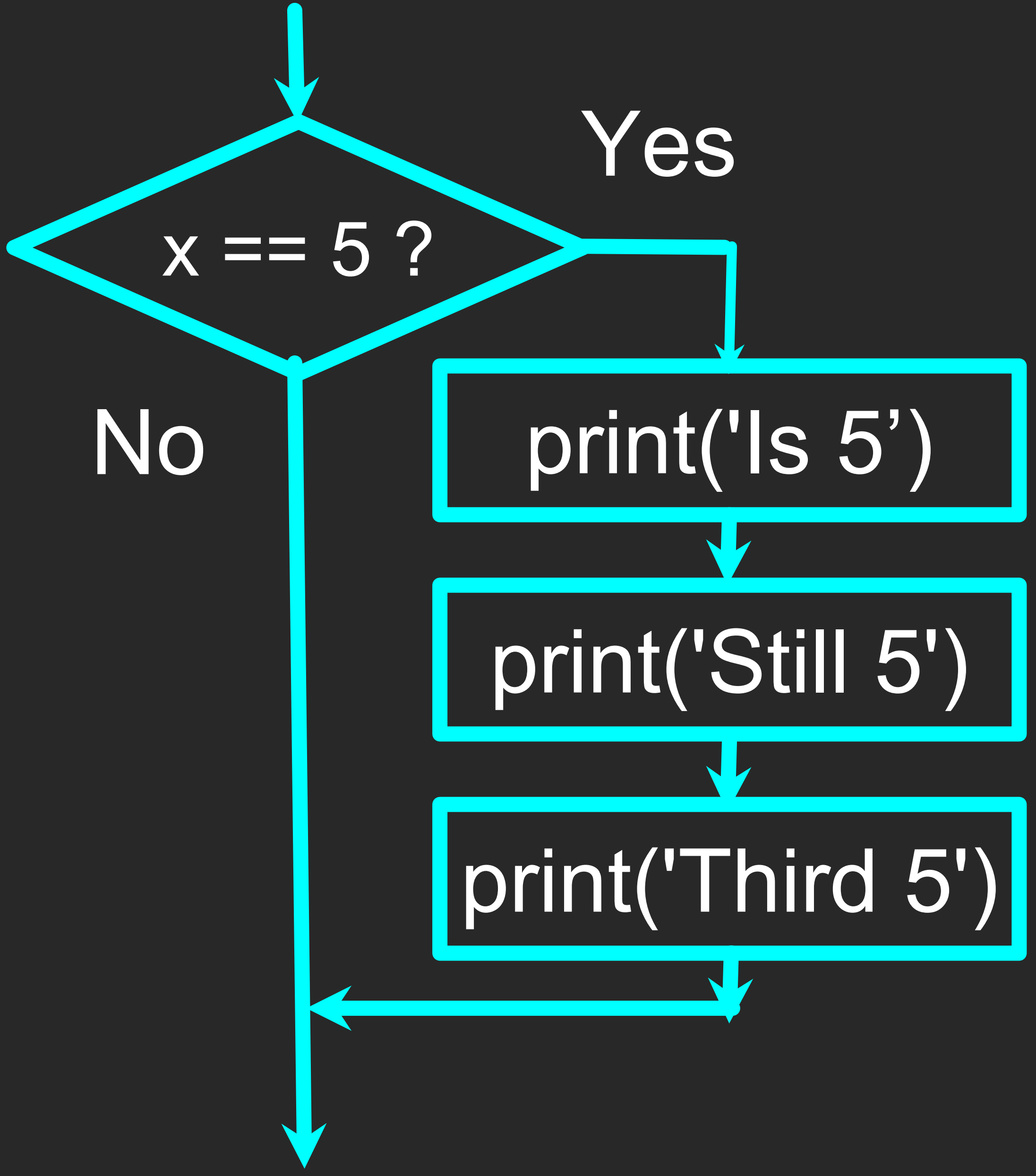
```
x = 5
if x == 5 :
    print('Equals 5')
if x > 4 :
   print('Greater than 4')
if  x >= 5 :
    print('Greater than or Equals 5')
if x < 6 : print('Less than 6')
if x <= 5 :
    print('Less than or Equals 5')
if x != 6 :
    print('Not equal 6')
```

```
Equals 5
Greater than 4
Greater than or Equals 5
Less than 6
Less than or Equals 5
Not equal 6
```

# One-Way Decisions

```
x = 5
print('Before 5')
if  x == 5 :
    print('Is 5')
    print('Is Still 5')
    print('Third 5')
print('Afterwards 5')
print('Before 6')
if x == 6 :
    print('Is 6')
    print('Is Still 6')
    print('Third 6')
print('Afterwards 6')
```

```
Before 5
Is 5
Is Still 5
Third 5
Afterwards 5
Before 6
Afterwards 6
```



# Indentation

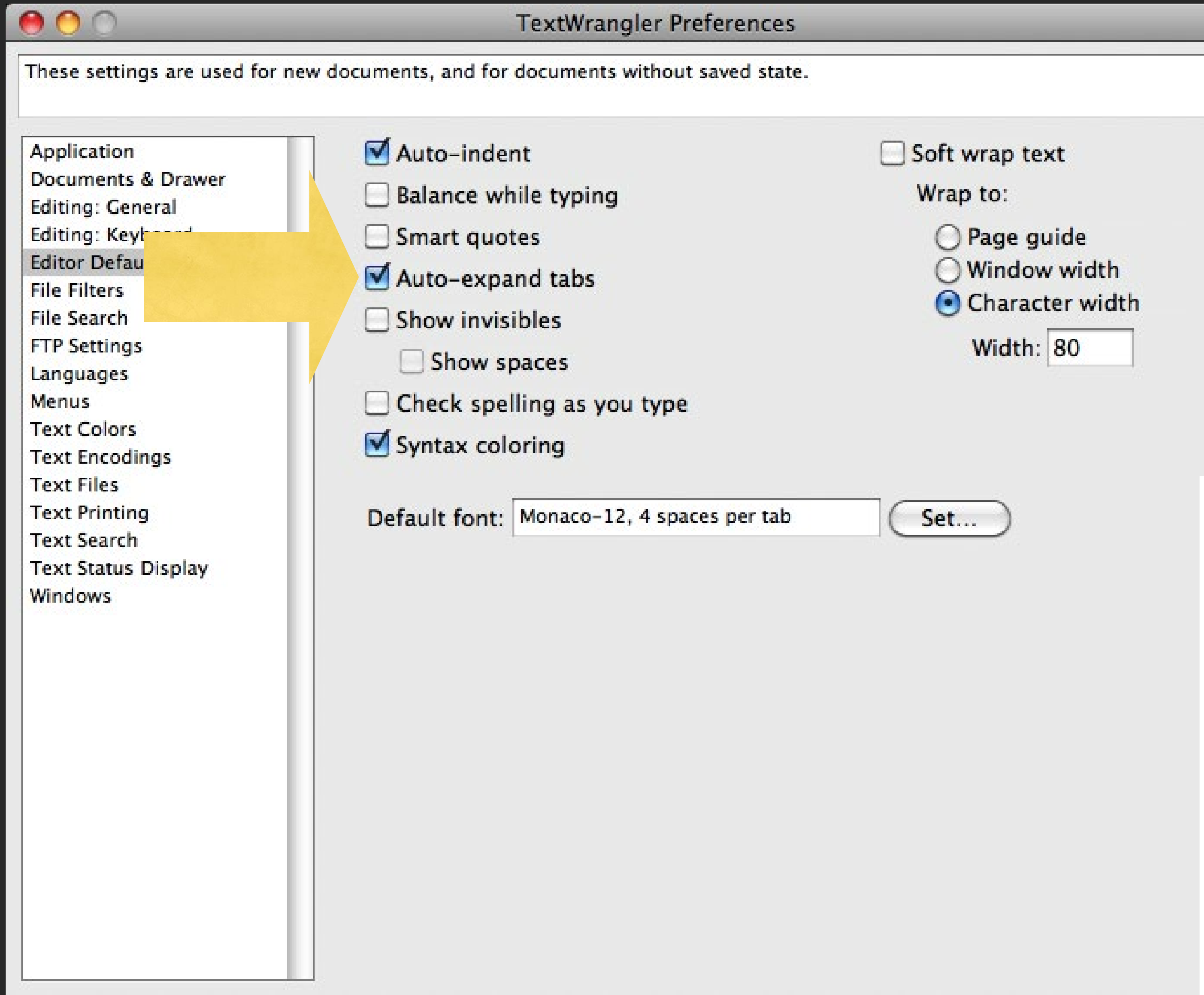
- Increase indent indent after an if statement or for statement (after : )
- Maintain indent to indicate the scope of the block (which lines are affected by the if/for)
- Reduce indent back to the level of the if statement or for statement to indicate the end of the block
- Blank lines are ignored - they do not affect indentation
- Comments on a line by themselves are ignored with regard to indentation

# Warning: Turn Off Tabs!!

Atom automatically uses spaces for files with ".py" extension (nice!)

- Most text editors can turn tabs into spaces - make sure to enable this feature
- Python cares a *lot* about how far a line is indented. If you mix tabs and spaces, you may get “indentation errors” even if everything looks fine

# Conditional – Part 1



TextWrangler Preferences

These settings are used for new documents, and for documents without saved state.

- ☑ Auto-indent
- □ Balance while typing
- □ Smart quotes
- ☑ Auto-expand tabs
- □ Show invisibles
  - □ Show spaces
- □ Check spelling as you type
- ☑ Syntax coloring

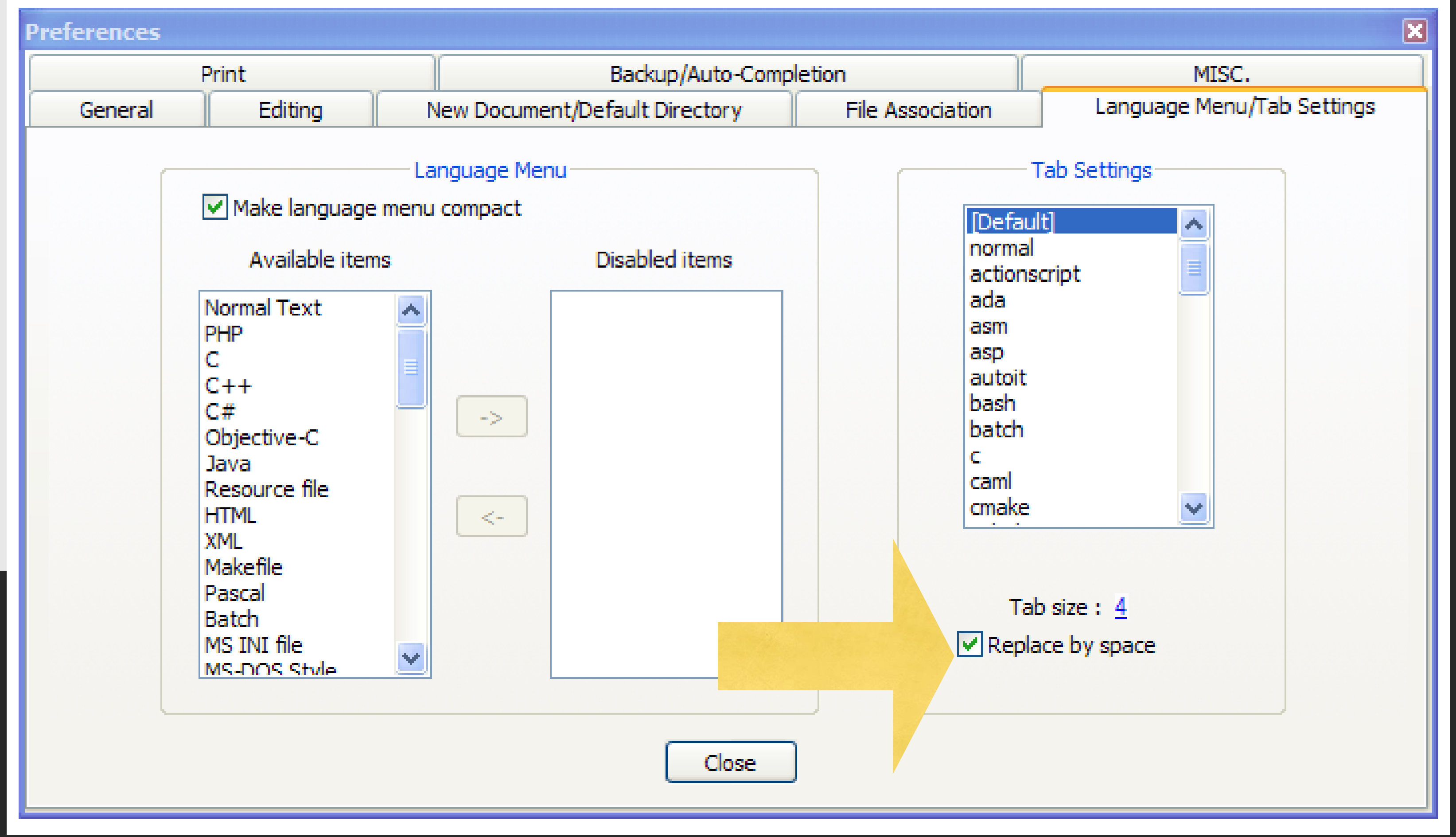
□ Soft wrap text

Wrap to:

- ○ Page guide
- ○ Window width
- ● Character width

Width: 80

Default font: Monaco-12, 4 spaces per tab



Preferences

☑ Make language menu compact

Tab size : 4

☑ Replace by space

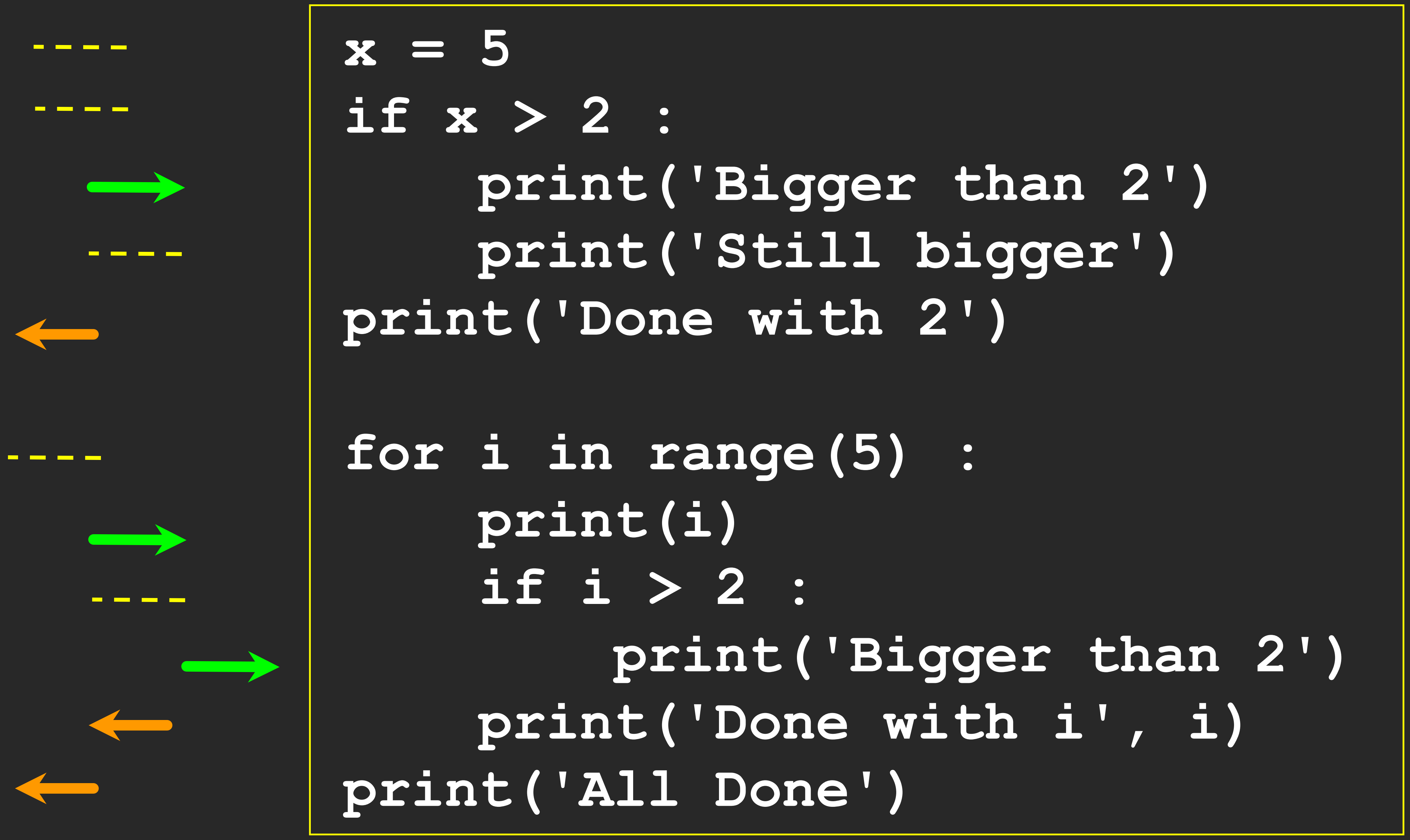
This will save you much unnecessary pain.

→ ----

increase / maintain after if or for

←

decrease to indicate end of block



```
x = 5
if x > 2 :
    print('Bigger than 2')
    print('Still bigger')
print('Done with 2')

for i in range(5) :
    print(i)
    if i > 2 :
        print('Bigger than 2')
    print('Done with i', i)
print('All Done')
```

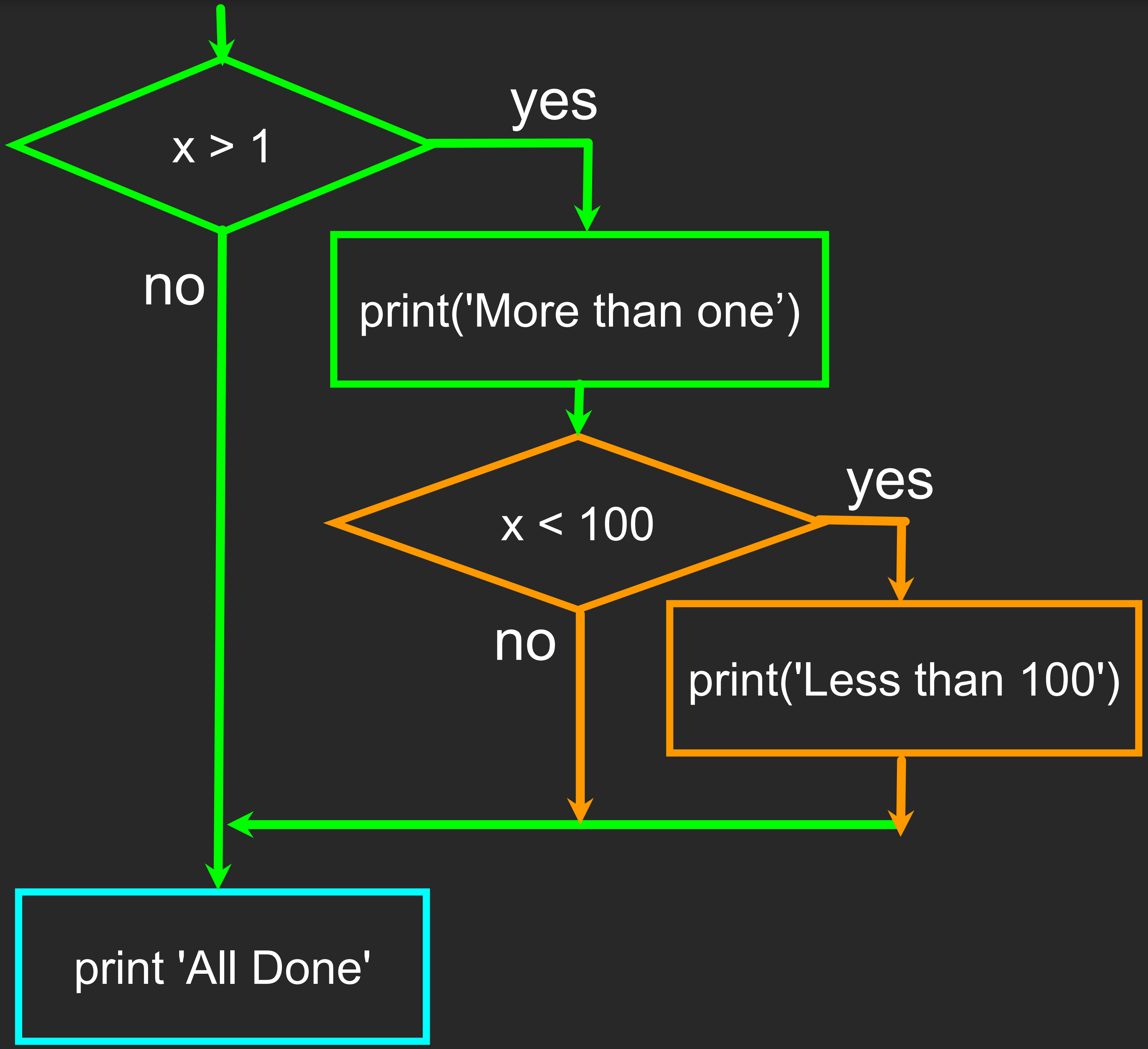
# Think about begin/end blocks

```
x = 5
if x > 2 :
    print('Bigger than 2')
    print('Still bigger')
print('Done with 2’)

for i in range(5) :
    print(i)
    if i > 2 :
        print('Bigger than 2')
    print('Done with i', i)
print('All Done')
```

# Nested Decisions

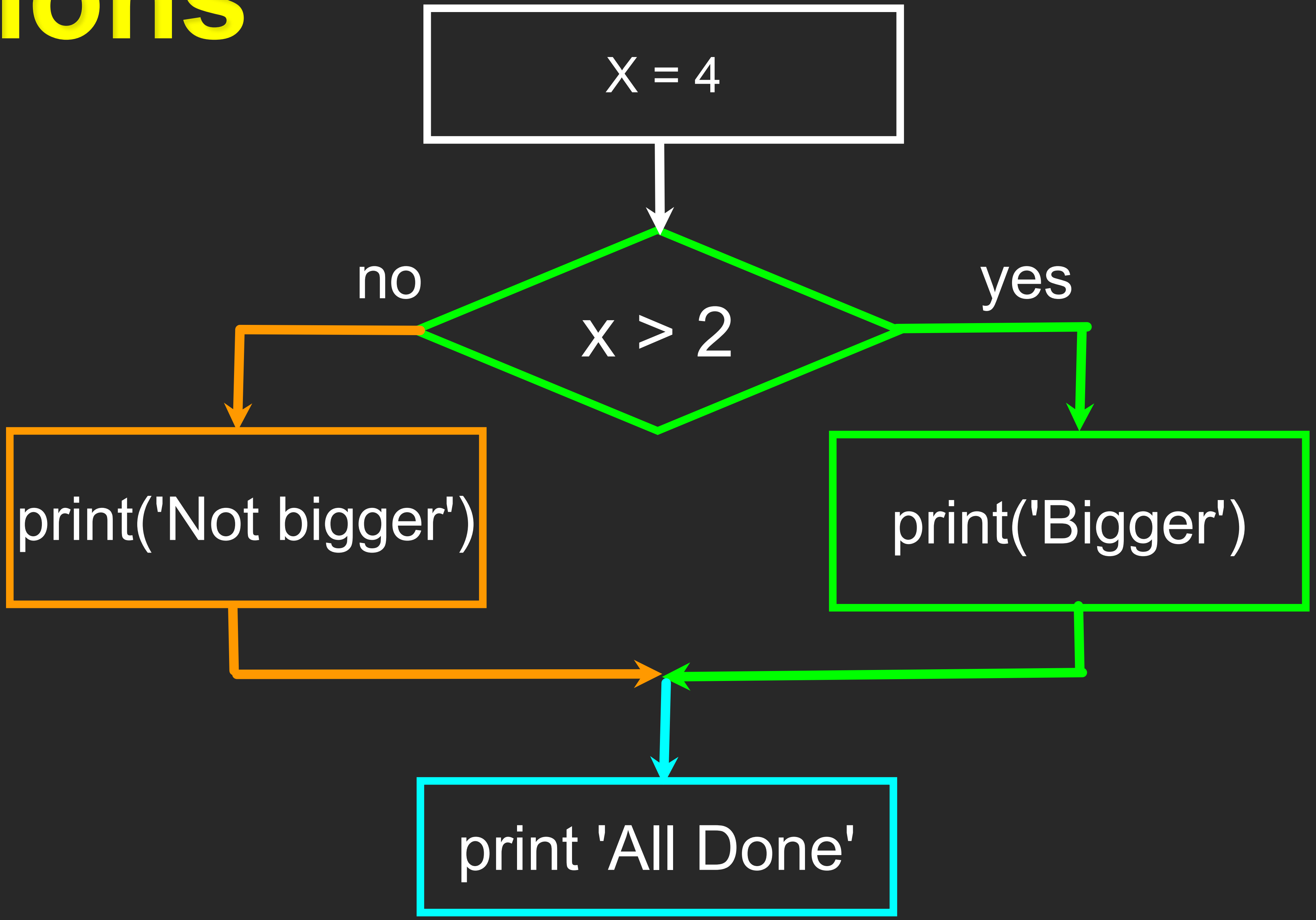
```
x = 42
if x > 1 :
    print('More than one')
    if x < 100 :
        print('Less than 100')
print('All done')
```



# Two-way Decisions

- Sometimes we want to do one thing if a logical expression is true and something else if the expression is false

- It is like a fork in the road - we must choose one or the other path but not both

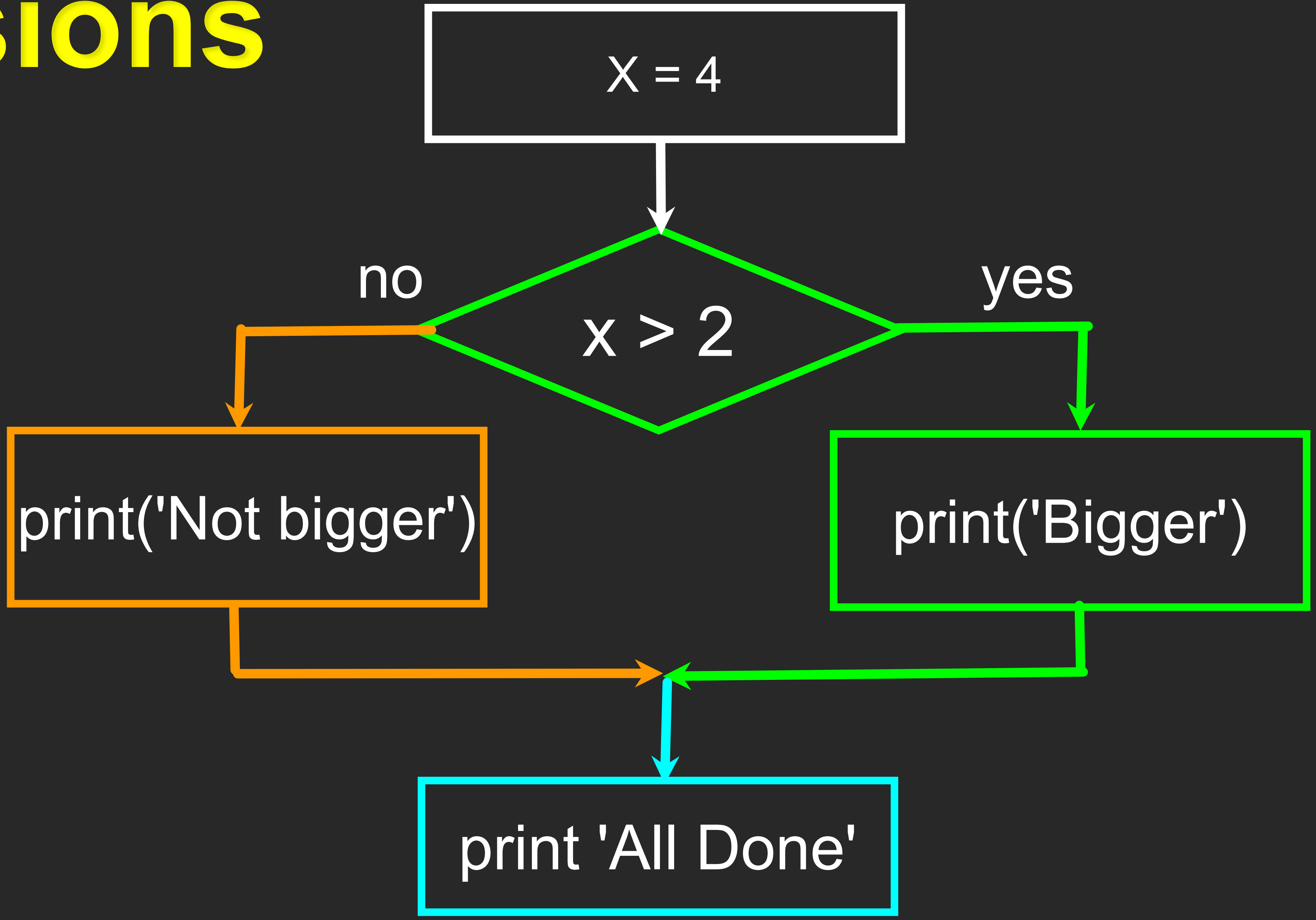


# Two-way Decisions with else:

```
x = 4

if x > 2 :
    print('Bigger')
else :
    print('Smaller')

print 'All done'
```



# More Conditional Execution Patterns