香港浸會大學
HONG KONG BAPTIST UNIVERSITY

Conditional Statements

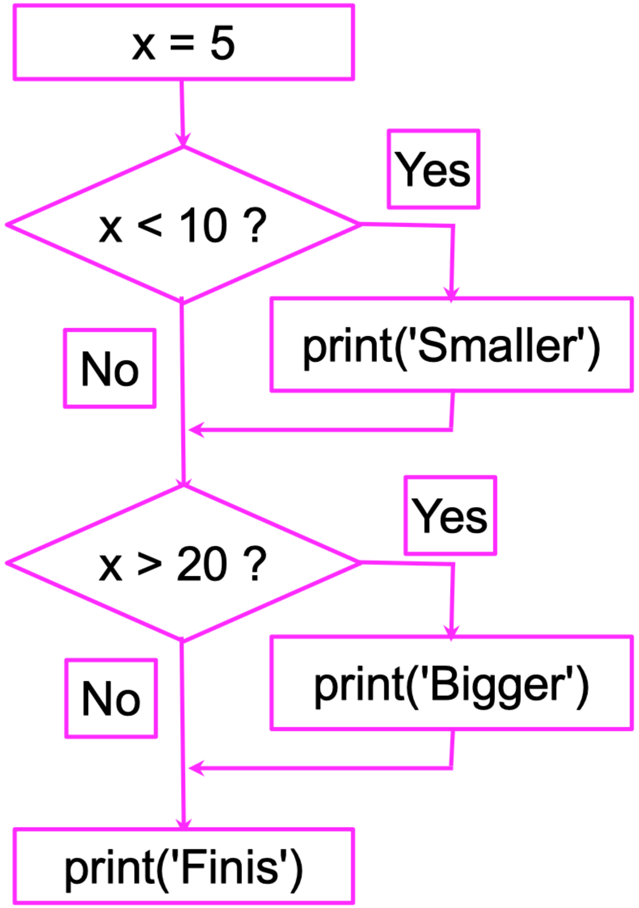
JOUR7280
Big Data Analytics for Media and Communication

Instructor: Dr. Xiaoyi Fu
xiaoyifu@hkbu.edu.hk

Outline

- One-way decisions
- Two-way decisions
- Multi-way decisions
- try / except

Conditional steps



Conditional statement

Program

```
x = 5
if x < 10:
    print('Smaller')
if x > 20:
    print('Bigger')
print('Finish')
```

Output

```
Smaller
Finish
```

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 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

Comparison Operators

```python
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')
```

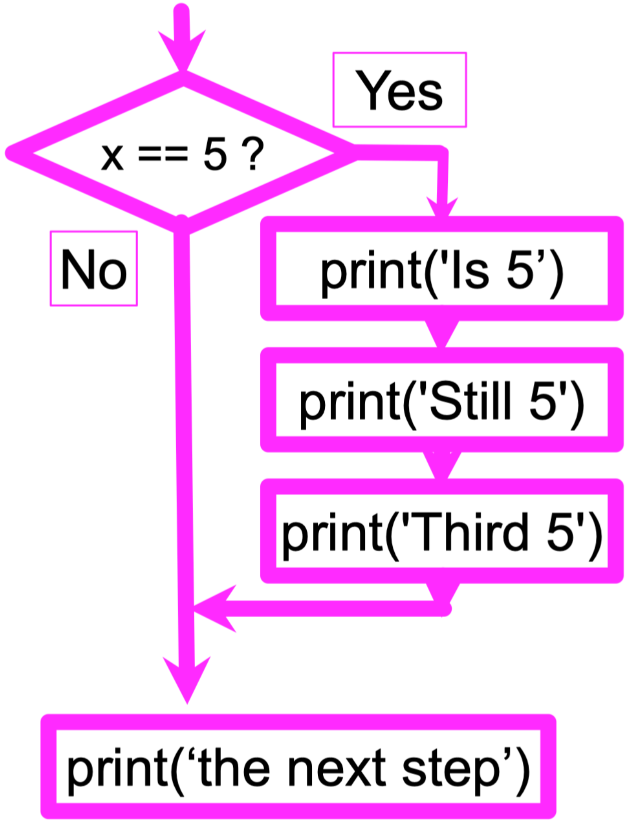
```
Equals 5
Greater than 4
Greater than or Equals 5
Less than 6
Less than or Equals 5
Not equal 6
```

3 conditional-statements.ipynb

One-way Decisions

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

```
Before 5
Is 5
Is Still 5
Third 5
Afterwords 5
Before 6
Afterwords 5
```



x == 5 ?

Yes

No

print('Is 5’)

print('Still 5')

print('Third 5')

print(‘the next step’)

3 conditional-statements.ipynb

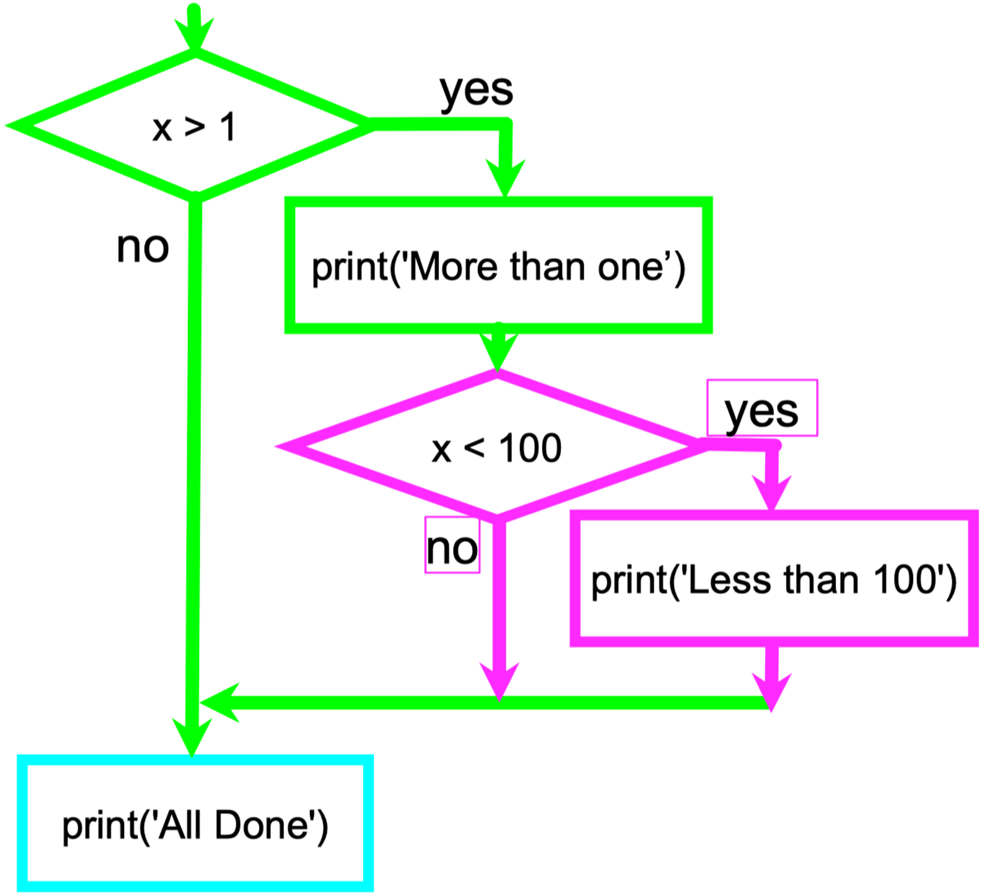
One-way Decisions

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

Increase / maintain after if
Decrease to indicate end of block

Nested Decisions

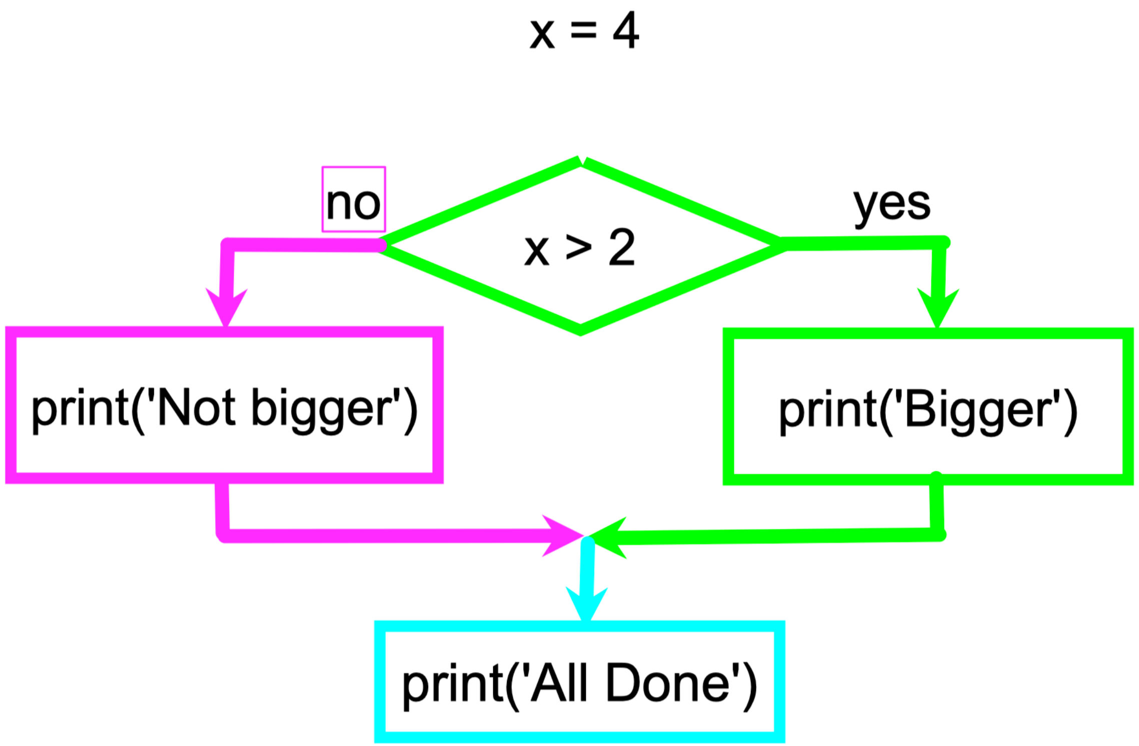
```
x = 42
if x > 1 :
    print('More than one‘)
    if x < 100 :
        print('Less than 100')
print('All done')
```



3 conditional-statements.ipynb

Two-way Decisions

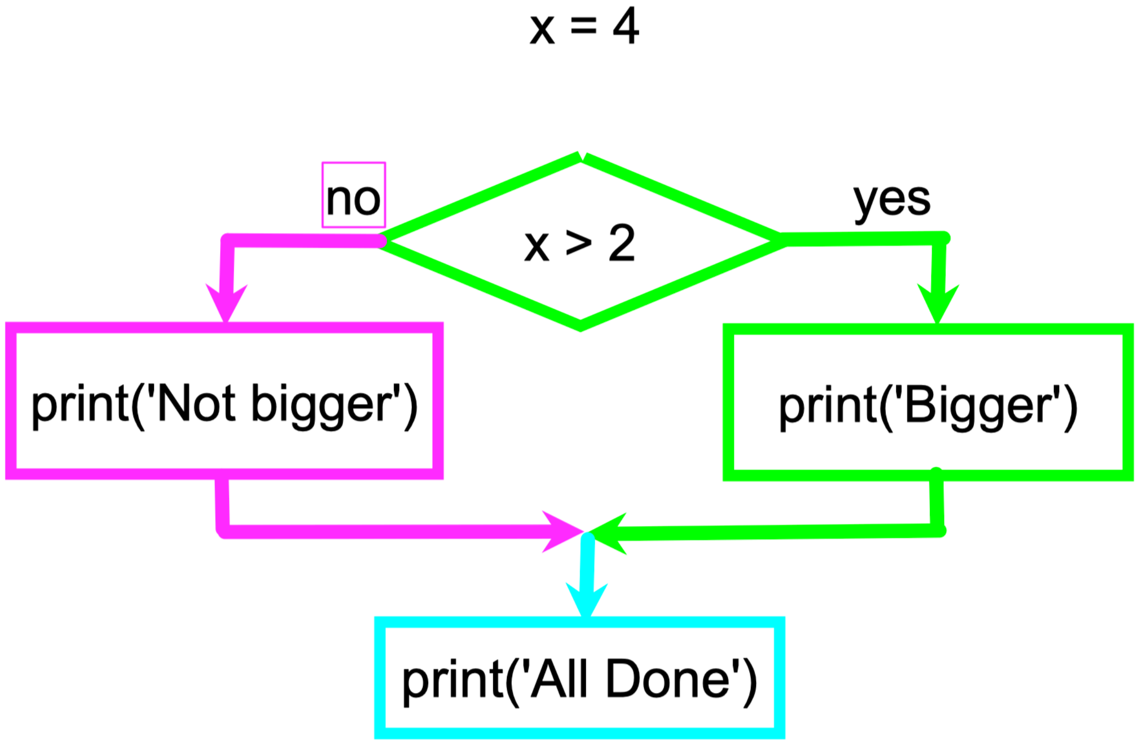
- Sometimes we want to do one thing if a logical expression is true and something else if the expression is false
- It is like a fork in the road - we must choose one or the other path but not both.



3 conditional-statements.ipynb

Two-way decisions with *else*

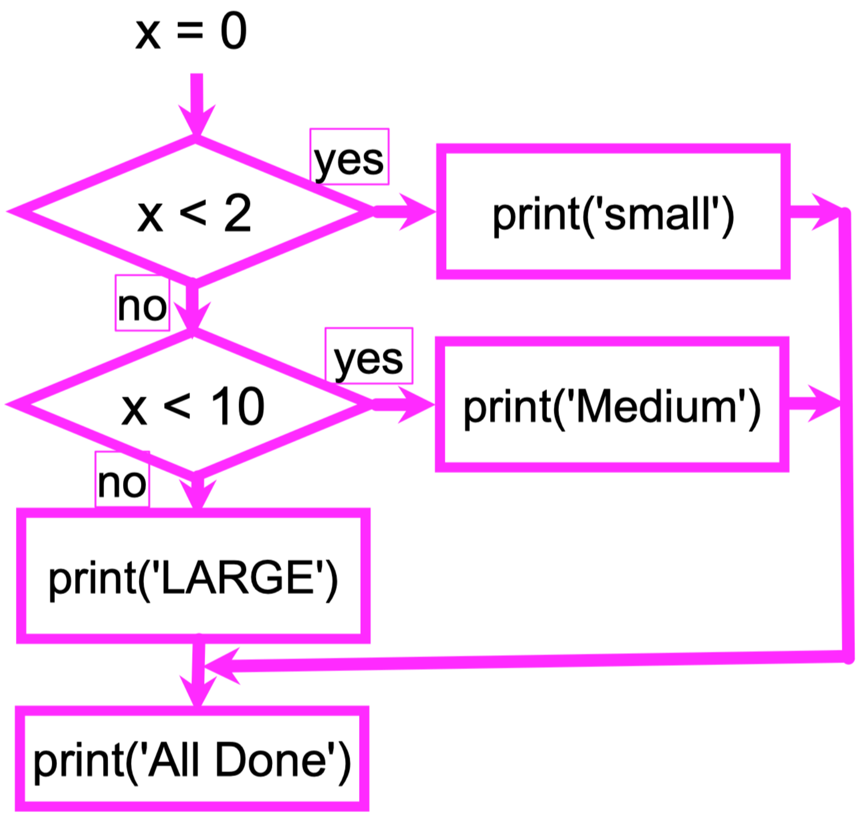
```
x = 4
if x > 2:
   print('Bigger')
else:
   print('Smaller')

print('All done')
```



3 conditional-statements.ipynb

Multi-way

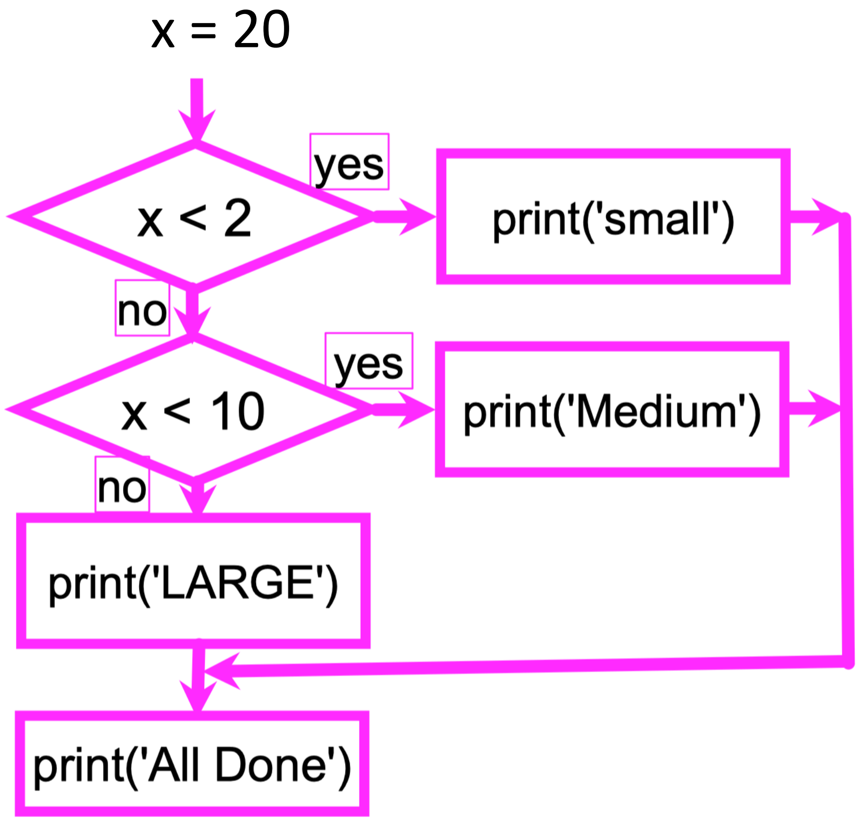
```
x = 0
if x < 2 :
   print('small')
elif x < 10 :
   print('Medium')
else :
   print('LARGE')
print('All done')
```



3 conditional-statements.ipynb

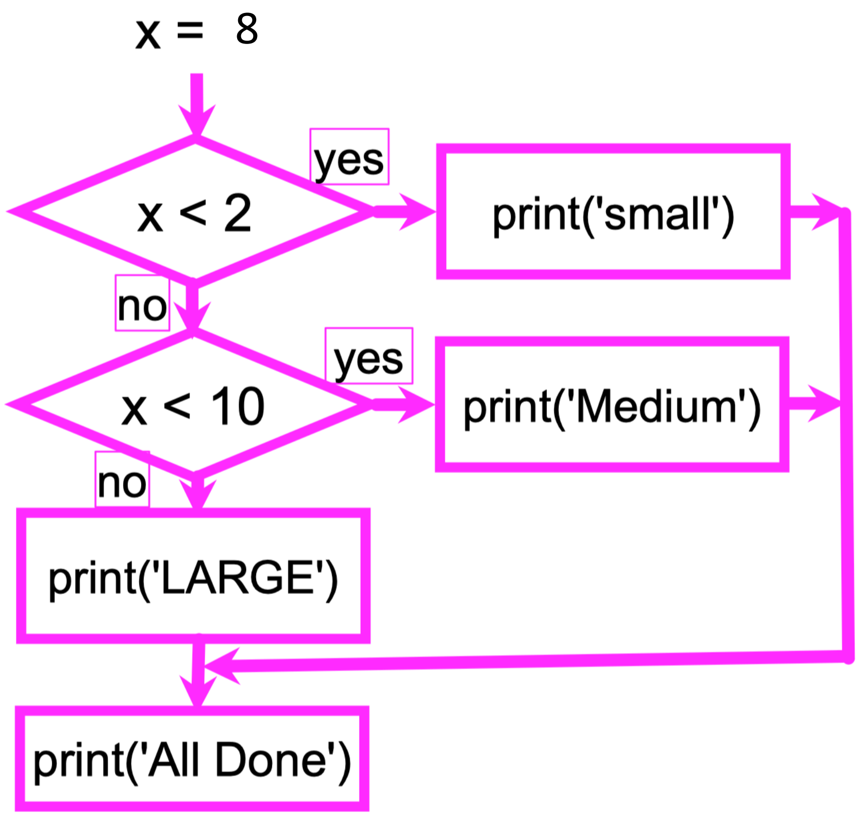
Multi-way

```
x = 20
if x < 2 :
   print('small')
elif x < 10 :
   print('Medium')
else :
   print('LARGE')
print('All done')
```



Multi-way

```
x = 8
if x < 2 :
   print('small')
elif x < 10 :
   print('Medium')
else :
   print('LARGE')
print('All done')
```



Multi-way

```
# No else
x = 5
if x < 2 :
   print('small')
elif x < 10 :
   print('Medium')

print('All done')
```

```
if x < 2 :
   print('small')
elif x < 10 :
   print('Medium')
elif x < 20 :
   print('Big')
elif x < 40 :
   print('Large')
elif x < 100:
   print('Huge')
else :
   print('Ginormous')
```

- There is no limit on the number of elif statements. If there is an else clause, it has to be at the end, but there doesn’t have to be one.

Multi-way Puzzles

- Which will never print regardless of the value for x?

```
if x < 2 :
   print('Below 2')
elif x > 2 :
   print(‘Above 2')
else :
   print('Something else')
```

```
if x < 2 :
   print('Below 2')
elif x < 20 :
   print('Below 20')
elif x < 10 :
   print('Below 10')
else :
   print('Something else')
```

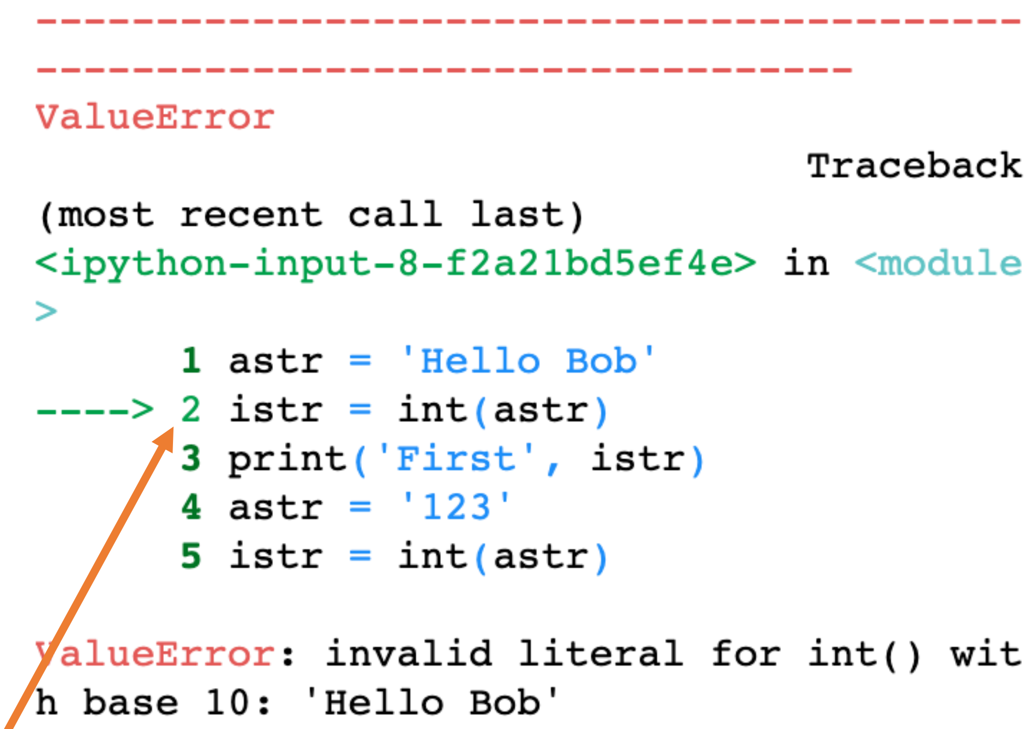
3 conditional-statements.ipynb

The try/except structure

- A way to eliminate/catch “traceback”
- You surround a dangerous section of code with try and except
- If the code in the try works
  - The except is skipped
- If the code in the try fails
  - It jumps to the except section

The try/except structure

```
astr = 'Hello Bob'
istr = int(astr)
print('First', istr)
astr = '123'
istr = int(astr)
print(Second', istr)
```



```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-8-f2a21bd5ef4e> in <module>
      1 astr = 'Hello Bob'
----> 2 istr = int(astr)
      3 print('First', istr)
      4 astr = '123'
      5 istr = int(astr)

ValueError: invalid literal for int() with base 10: 'Hello Bob'
```

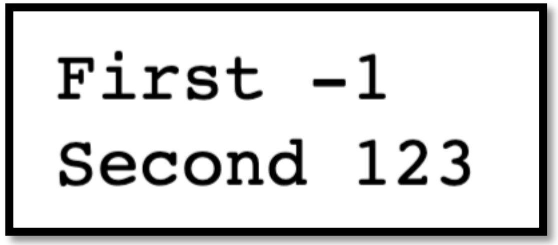
Last line it executed, won't continue
Quit at line 2

3 conditional-statements.ipynb

The try/except structure

```
astr = 'Hello Bob'
try:
    istr = int(astr)
except:
    istr = -1
print('First', istr)

astr = '123'
try:
    istr = int(astr)
except:
    istr = -1
print('Second', istr)
```

When the first conversation fails – it just drops into the except: clause and the program continues



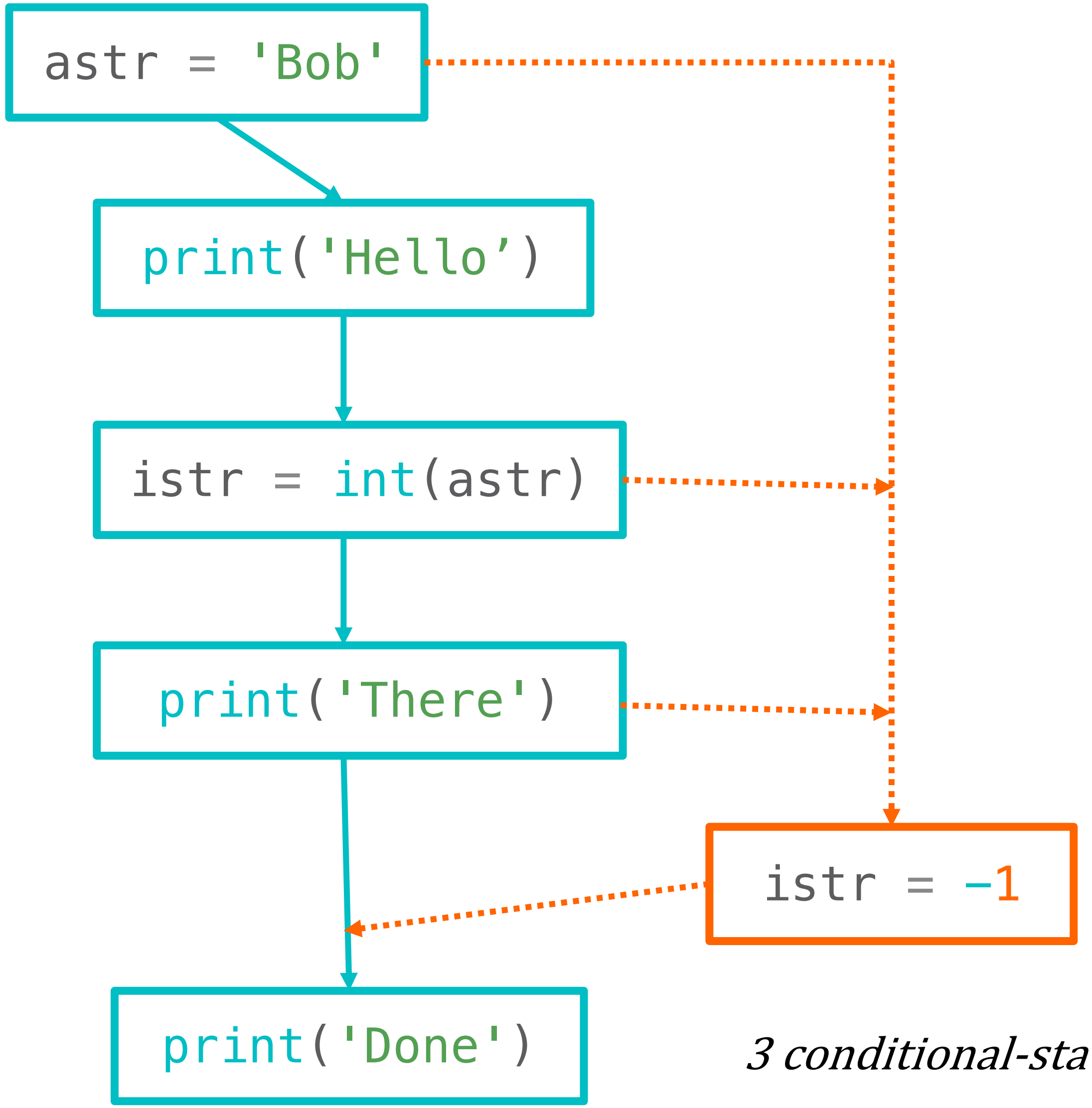
```
First -1
Second 123
```

When the second conversation succeeds – it just skips the except: clause and the program continues

3 conditional-statements.ipynb

try/except

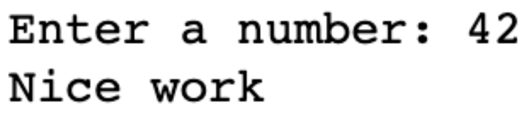
```
astr = 'Bob'
try:
    print('Hello’)
    istr = int(astr)
    print('There')
except:
istr = -1

print('Done', istr)
```



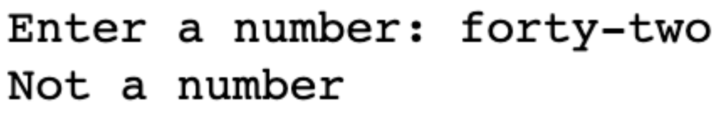
3 conditional-statements.ipynb

try/except

```python
rawstr = input('Enter a number: ')
try:
   ival = int(rawstr)
except:
   ival = -1

if ival > 0:
   print('Nice work')
else:
   print('Not a number')
```



```
Enter a number: 42
Nice work
```



```
Enter a number: forty-two
Not a number
```

3 conditional-statements.ipynb

Thank You