Accelerating Information Technology Innovation

http://aiti.mit.edu

Ghana Summer 2012
Lecture 5 – Control Structures, Decisions

Beyond sequential execution

- So far, all our programs have looked like this:

```
<do thing 1>
<do thing 2>
<do thing 3>
…
```

Start with first command. Execute commands in order until there are no more.

But often sequential execution is not enough.

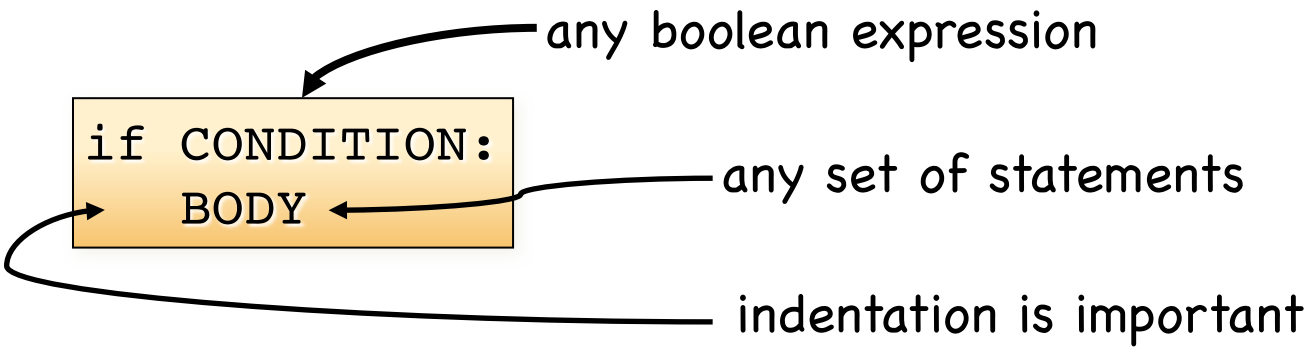
```
if <something>:
  <do thing 1>
else:
  <do thing 2>
```

If something is true, execute the first command. Otherwise, execute the second command.

Control statements

- Conditionals: control which set of statements is executed.
  - if / else
- Iteration: control how many times a set of statements is executed.
  - while loops
  - for loops

The if statement



- If the condition is True, the body gets executed.
- Otherwise, nothing happens.

```
if x < 0:
   print 'x is negative'
```

- NOTE: IDLE editor helps with indentation.

The if/else statement

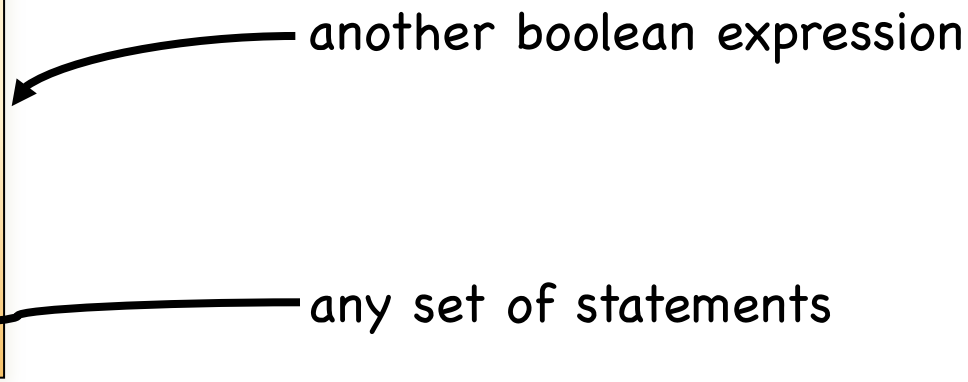
```
if CONDITION:
    BODY1
else:
    BODY2
```

- If the condition is True, body1 gets executed.
- Otherwise, body2 gets executed.

```
if x < 0:
    print 'x is negative'
else:
    print 'x is positive or zero'
```

Chained conditionals

```
if CONDITION1:
    BODY1
elif CONDITION2:
    BODY2
else:
    BODY3
```



- If the condition1 is True, body1 gets executed.
- Otherwise, if condition2 is True, body2 gets executed.
- If neither condition is True, body3 gets executed.

Chained conditionals

- if temp x < 0:
    print "x is negative"
  elif x==0:
    print "x is zero"
  else:
    print "x is positive“

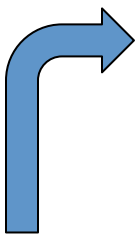
An example

```
a = False
b = True
if a and b:
   print 'I love red.'
elif a or b:
   print 'I love green.'
else:
   print 'I love blue.'
   print 'I also love purple.'
```

What does this output? `I love green.`

An example

```
a = False
b = True
if a and b:
   print 'I love red.'
elif a or b:
   print 'I love green.'
else:
   print 'I love blue.'
print 'I also love purple.'
```



What does this output?

```
I love green.
I also love purple.
```

Nested conditionals

```
if is_adult:
    if is_senior_citizen:
        print 'Admission $2 off.'
    else:
        print 'Full price.'
else:
    print 'Admission $5 off.'
```

outer conditional

inner conditional

- Can get confusing. Indentation helps to keep the code readable and the python interpreter happy!

Another example

```
x = 4
y = -3
if x < 0:
    if y > 0:
        print x
+ y
    else:
        print x
- y
else:
```

What does this output?

-12

Common if errors

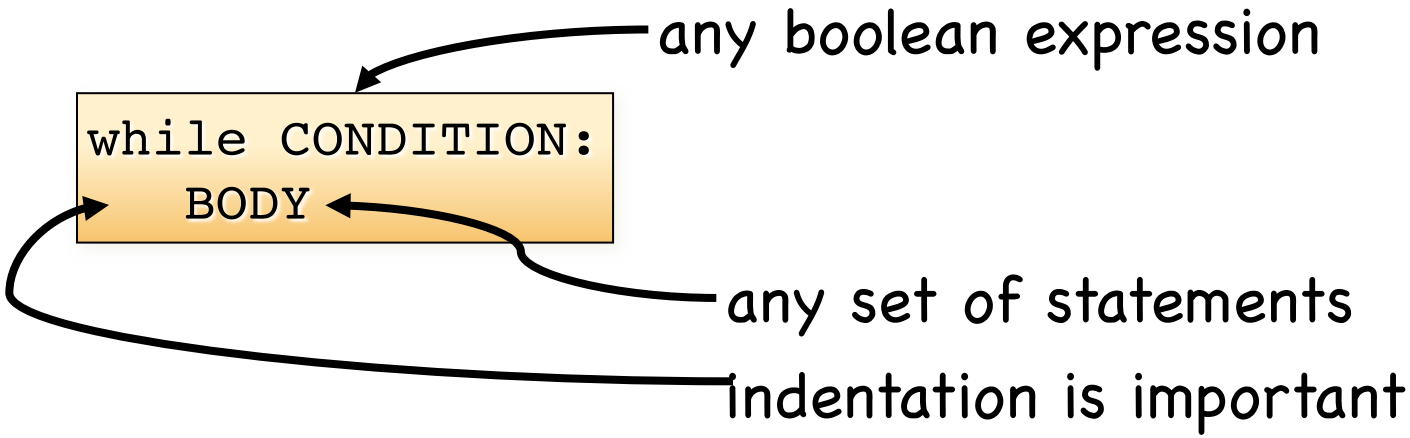
- Syntax errors
  - Mixing up = and == in the condition

```
b = False
if b = False
   print b
  print 'inside if maybe'
```

```
SyntaxError: invalid syntax
```

```
IndentationError: unindent does not match
any outer indentation level
```

The while loop



- As long as the condition is true, the body gets executed repeatedly.
- The first time the condition is false, execution ends.

The while loop

```
i = 0
while i < 3:
    print i
    i = i + 1
```

- What does this output?

```
0
1
2
```

Side note: if the condition is false the first time it is tested, the body is never executed

The break statement

- Immediately exits the innermost loop.

```
i = 0
while True:
    i+=1
    line = raw_input('>>> ')
    if line == 'done':
        break
    print i
print 'Done!'
```

(An if statement is not a loop!)

What' will happen with this code?

```
i = 0
while i < 3:
    print i
```

- It will loop forever (aka Infinite loop)! How do we fix it?

```
i = 0
while i < 3:
    print i
    i = i + 1
```

The infinite loop

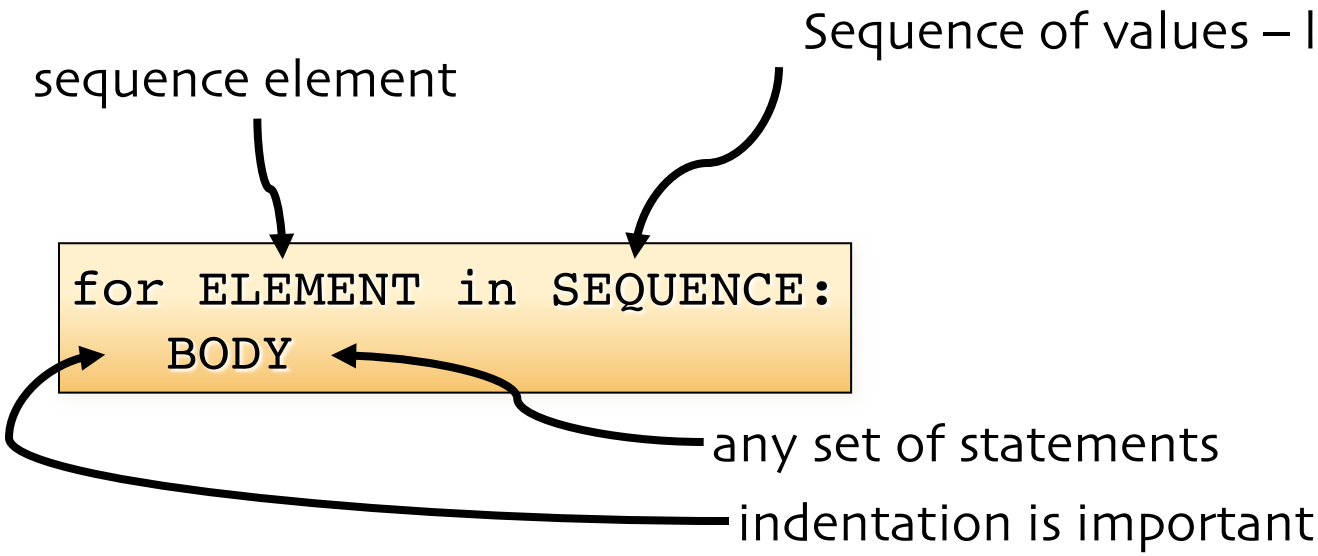
```
i = 4
while i > 0:
    print i
    i = i + 1
```

This code also loops forever!

Why? And how do you fix this?

```
i = 4
while i > 0:
    print i
    i = i - 1
```

The for loop

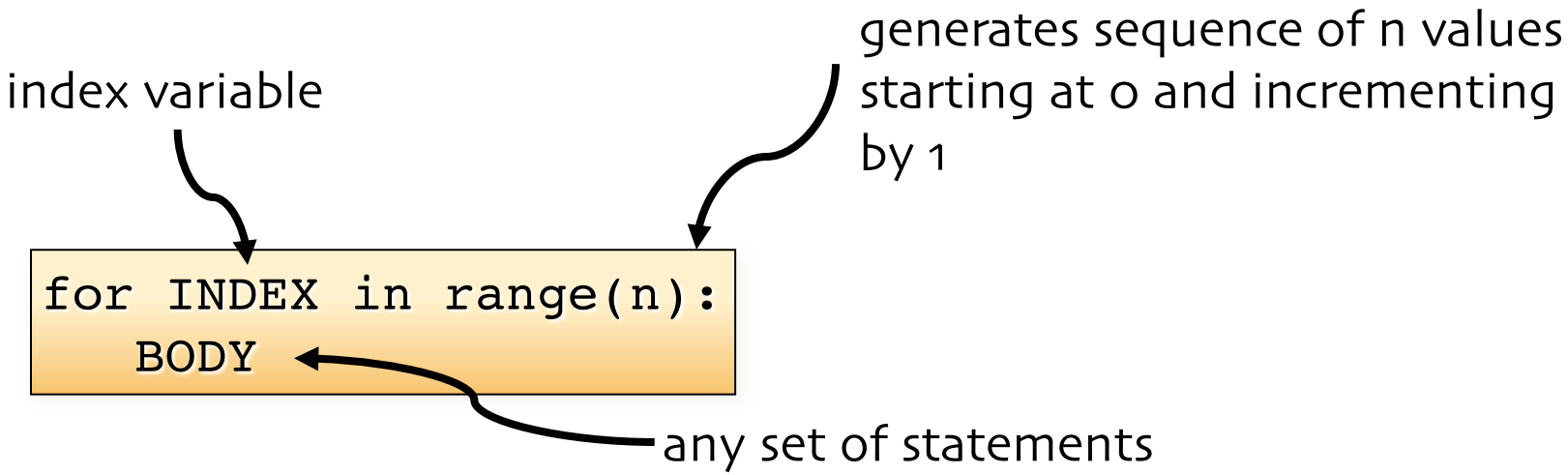


- Example:

```
for i in [0,1,2,3]:
    print i
```

```
0
1
2
3
```

Using range

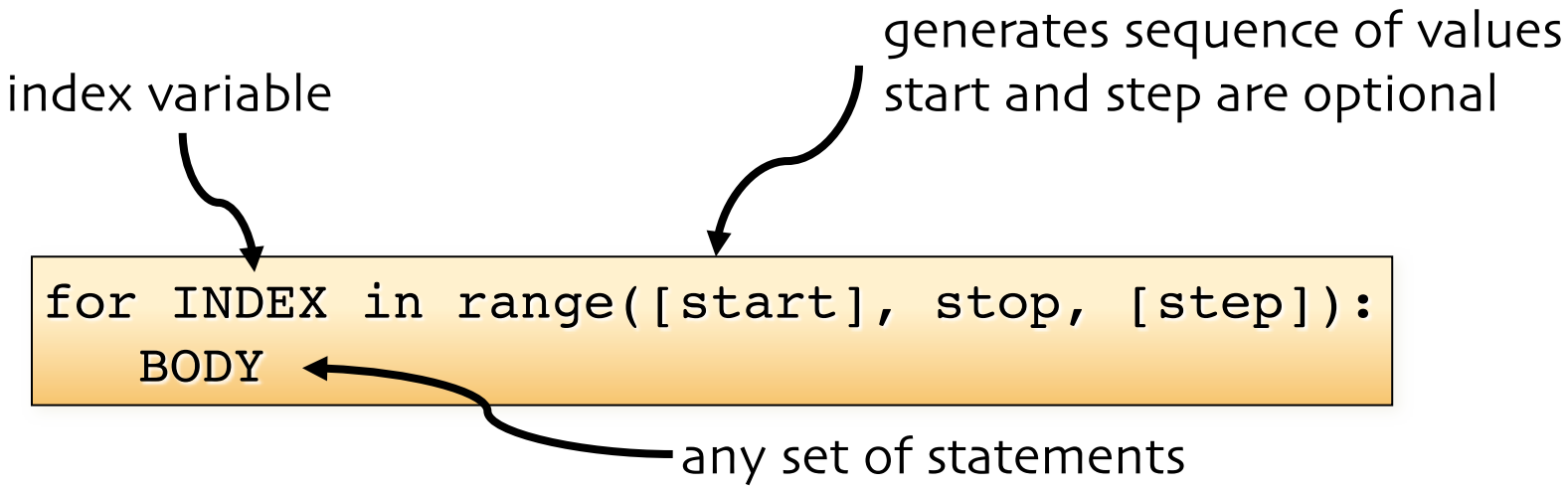


- What does this output?

```
for i in range(4):
    sq = i * i
    print i, sq
```

```
0   0
1   1
2   4
3   9
```

Using range



- What does this output?

```
for i in range(1, 7, 2):
    print i
```

1
3
5

For loop and strings

- Iterating through the characters of a string

```
str1 = 'stressed'
for c in str1:
    print c,
```

```
s t r e s s e d
```

For loop and strings

- What is the output?

```
str1 = 'stressed'
res = ''
for c in str1:
    res = c + res
print res
```

desserts

Iteration #	c	res
0	s	s
1	t	ts
2	r	rts
3	e	erts
4	s	serts
5	s	sserts
6	e	esserts
7	d	desserts

Combining for and if

```
for i in range(6):
    if i % 2 == 0:
        print i, 'is even.'
    else:
        print i, 'is odd.'
```

- What does this output?

```
0 is even.
1 is odd.
2 is even.
3 is odd.
4 is even.
5 is odd.
```

Nested for loops

must use new index variable for inner loop

```
for i in range(1,6):
    for j in range(1, 6):
        prod = i * j
        # use comma to print all on one line
        print prod,
    print
```

- What does this output?

```
1   2   3   4   5
2   4   6   8   10
3   6   9   12  15
4   8   12   16   20
5   10   15   20   25
```

33 24

For vs While

- For loop is primarily used
  - for iterating over a sequence of values
  - when we know the number of iterations in advance
- While loop is primarily used
  - when we don't know the number of iterations in advance (they could be controlled by user input)

Questions?