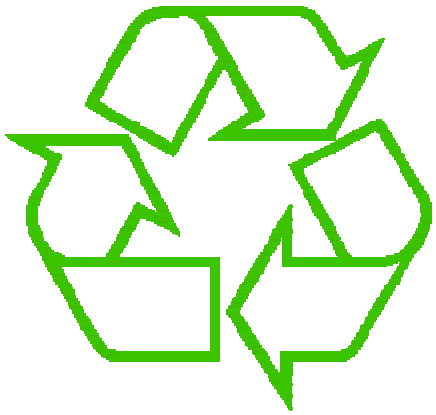
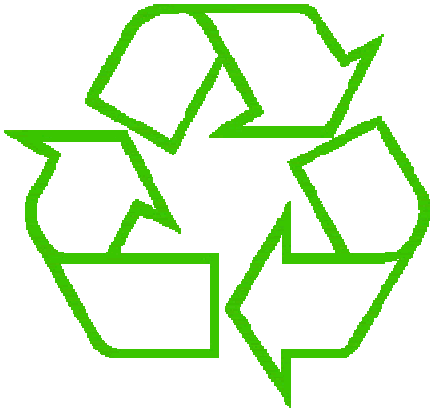


Control flow

Michael Ernst

UW CSE 190p

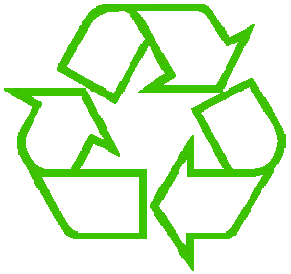
Summer 2012



Repeating yourself

Making decisions

Temperature conversion chart



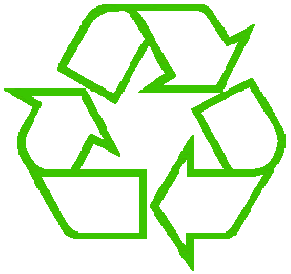
Recall exercise from previous lecture

```
fahr = 30
cent = (f-32)/9.0*5
print fahr, cent
fahr = 40
cent = (f-32)/9.0*5
print fahr, cent
fahr = 50
cent = (f-32)/9.0*5
print fahr, cent
fahr = 60
cent = (f-32)/9.0*5
print fahr, cent
fahr = 70
cent = (f-32)/9.0*5
print fahr, cent
print "All done"
```

Output:

30 -1.11
40 4.44
50 10.0
60 15.56
70 21.11
All done

Temperature conversion chart



Revisit exercise from previous lecture

```
for f in [30,40,50,60,70]:
    print f, (f-32)/9.0*5
print "All done"
```

`for` loop

loop variable or iteration variable

A list

Loop *body* is indented

Execute the body 5 times:

- once with f = 30
- once with f = 40
- …

Indentation is significant

Output:

30 -1.11
40 4.44
50 10.0
60 15.56
70 21.11
All done

The body can be multiple statements

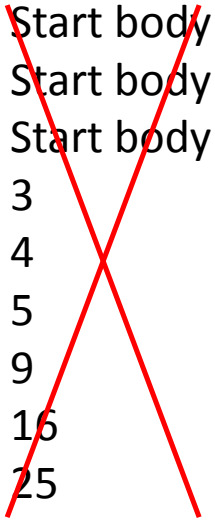
```
for i in [3,4,5]:
  print "Start body"
  print i
  print i*i
```

loop body: 3 statements

Output:

```
Start body
3
9
Start body
4
16
Start body
5
25
```

NOT:



Execute whole body, then execute whole body again, etc.

```
for i in [0,1]:
  print "Outer", i
  for j in [2,3]:
    print "  Inner", j
    print "    Sum", i+j
  print "Outer", i
```

"nested" loop body: 2 statements

loop body: 3 statements

Output:

```
Outer 0
  Inner 2
    Sum 2
  Inner 3
    Sum 3
Outer 0
Outer 1
  Inner 2
    Sum 3
  Inner 3
    Sum 4
Outer 1
```

Convention: often use i or j as loop variable

This is an exception to the rule that variable names should be descriptive

Indentation is significant

- Every statement in the body must have exactly the same indentation

```
for i in [3,4,5]:
    print "Start body"
Error!  print i
    print i*I
```

- Compare the results of these loops:

```
for f in [30,40,50,60,70]:
    print f, (f-32)/9.0*5
print "All done"

for f in [30,40,50,60,70]:
    print f, (f-32)/9.0*5
    "All done"
```

Fix this loop

```
# Goal:  print 1, 2, 3, …, 48, 49, 50
for tens_digit in [0, 1, 2, 3, 4]:
  for ones_digit in [1, 2, 3, 4, 5, 6, 7, 8, 9]:
    print tens_digit * 10 + ones_digit
```

What does it actually print?

How can we change it to correct its output?

Moral: Watch out for *edge conditions* (beginning or end of loop)

How a loop is executed (2 versions)

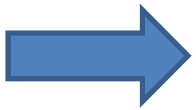
Transformation approach:

1. Evaluate sequence expression
2. Write an assignment to the loop variable for each sequence element
3. Write a copy of the loop after each assignment
4. Execute the resulting statements

Direct approach:

1. Evaluate sequence expression
2. While there are sequence elements left:
   1. Assign the loop variable to the first remaining sequence element
   2. Execute the loop body

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



```
i = 1
print i
i = 2
print i
i = 3
print i
```

Another example of the transformation approach

Key idea:

1. Assign each sequence element to the loop variable
2. Duplicate the body

```
for i in [0,1]:
  print "Outer", i
  for j in [2,3]:
    print " Inner", j
```

```
i = 0
print "Outer", i
for j in [2,3]:
  print " Inner", j
i = 1
print "Outer", i
for j in [2,3]:
  print " Inner", j
```

```
i = 0
print "Outer", i
j = 2
print " Inner", j
j = 3
print " Inner", j
i = 1
print "Outer", i
for j in [2,3]:
  print " Inner", j
```

Test your understanding of loops

Puzzle 1:

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

Output:

0
1
1

Puzzle 2:

```
i = 5
for i in []:
  print i
```

(no output)

Puzzle 3:

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

Reusing loop variable (don't do this)

inner loop body

outer loop body

Outer 0
 Inner 2
 Inner 3
Outer 3
Outer 1
 Inner 2
 Inner 3
Outer 3

The range function

A typical for loop does not use an explicit list:

```
for i in range(5):
    … body …
```

The list [0,1,2,3,4]

Upper limit (*exclusive*)

`range(5)` = [0,1,2,3,4]

Lower limit (*inclusive*)

`range(1,5)` = [1,2,3,4]

step (distance between elements)

`range(1,10,2)` = [1,3,5,7,9]

Making decisions

- How do we compute absolute value?

  abs(5) = 5

  abs(0) = 0

  abs(-22) = 22

Absolute value solution

If the value is negative, negate it.

Otherwise, use the original value.

```
val = -10
if val < 0:
    result = - val
print result
```

```
val = -10
if val < 0:
    result = - val
else:
    result = val
print result
```

```
val = -10
if val < 0:
    print - val
else:
    print val
```

The if body can be any statements

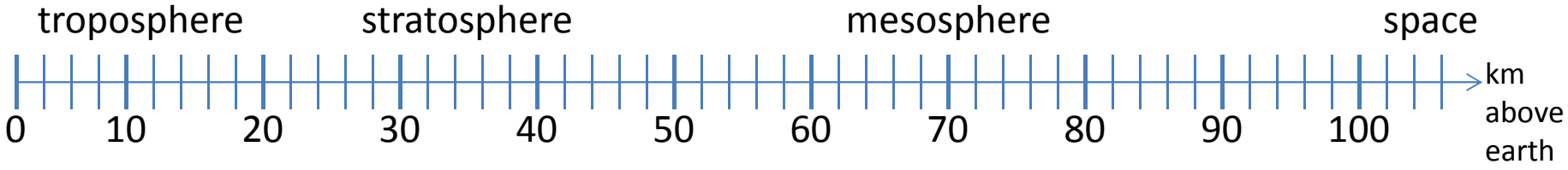
```
# height is in km
if height > 100:
    print "space"
else:
    if height > 50:
        print "mesosphere"
    else:
        if height > 20:
            print "stratosphere"
        else:
            print "troposphere"
```

then clause

else clause

Execution gets here only if "height > 100" is false

Execution gets here only if "height > 100" is false AND "height > 50" is true



The then clause *or* the else clause is executed

```
if is_prime(x):
    y = x / 0
else
    y = x*x
```