CS 61A
Discussion 1

Control and Higher Order Functions

Raymond Chan
Discussion 121
UC Berkeley

Announcements

- Lab 01 due Friday
- Project 1 Hog is released. Due Thu 2/4
- Office Hours B6 Evans
  - Mon-Thur 11-6
  - Fri 11-4

If Statements

- Execute different code based on different conditions
- Conditional expressions require expressions that return True/False
    - Boolean operators, comparing numbers (3 == 5)

```
if <conditional>:
    <suite of statements>
elif <conditional>:
    <suite of statements>
else:
    <suite of statements>
<rest of code>
```

If Statements

```
if <conditional>:
    <suite of statements>
elif <conditional>:
    <suite of statements>
else:
    <suite of statements>
<rest of code>
```

```
if <conditional>:
    <suite of statements>
if <conditional>:
    <suite of statements>
if <conditional>:
    <suite of statements>
<rest of code>
```

Boolean Operators

- not: returns the opposite
    - always returns True or False
- and: returns the first False value (short-circuiting)
    - if all True -> evaluates the last value
- or: returns the first True value
    - if all False -> evaluates the last value

Boolean Operators

- and/or uses the bool(x) function to determine True/False values
- do not have to return True/False
  - False values: False, 0, None, “”, []…
  - True values: True, non-zero integers, almost everything else

While loops

- As long as the conditional returns True, the body is executed
- Watch out for infinite loops!!!
- Within the body, the conditional needs to change after each iteration

```
while <conditional>:
        <body>
```

```
i = 0
while i < n:
        <body>
        ...
        i = i + 1
```

Higher Order Functions

- Function arguments can be other functions
- Pass in the name of the function
- Don’t make a function call (Demo)

Higher Order Functions

- Functions can also return other functions
  - Usually returns the name
  - But can be a function call (in the next couple of weeks)