Part II

Control Flow

Control Flow

General

- Indentation used to group blocks!
- Code within same block has to be same level
- Needs getting used to, but encourages readable code

if-statement

```
if x < y:
    print("x is smaller")
    print(x)
elif x == y:
    print("both are equal")
else:
    print("y is smaller")
    print(y)
```

Comparison and Equality

- Check for equality

```
"Bilbo" == "Frodo"
"Bilbo" != "Frodo"
```

- Comparison

```
"Bilbo" < "Frodo"
"Bilbo" > "Frodo"
"Bilbo" <= "Frodo"
"Bilbo" >= "Frodo"
```

- `is` and `in`

```
"Bilbo" is "Frodo"                  # object identity
"Bilbo" in ["Frodo", "Sam", "Bilbo"] # membership
```

Objects: Type, Value & Identity

- Each object: possesses identity (id), type (type) and value

```
>>> b = 42   # value: 42
>>> type(b)  # <type 'int'>
>>> id(b)    # 158788940
```

- Comparison of two objects:

```
if a is b:
  print('a␣and␣b␣are␣the␣identical␣object')
if a == b:
  print('a␣and␣b␣have␣the␣same␣value')
if type(a) is type(b):
  print('a␣and␣b␣have␣the␣same␣type')
```

- Combining and grouping (`and`, `or`, `not`)

```
if ("Bilbo" != "Frodo" and not (1 > 2 or 2 > 3)):
  print("Sam␣was␣here")
```

Control Flow - for

Python's for iterates over the items of a sequence type

```
for i in range(5,10):
  print("loop index:", i)
```

```
a = ("a", "tuple", "with", 5, "words")
for i in a:
  print(i)
```

Control Flow - break and while

- `break` breakes out of the enclosing loop (`for` or `while`)
- `continue` continues with the lext iteration of the loop

```
a = ("let", "us", "find", "Bilbo", "again")
for word in a:
  if word == "Bilbo":
    print("found␣Bilbo")
    break
print(word)
```

- `while` loops until condition is `False`

```
a = 2048; exp = 0
while a > 1:
  a /= 2          # a = a / 2
  exp+=1          # exp = exp + 1
print(a, "=", 2, "^", exp)
```

Command-Line Options

Using sys

- `sys.argv` is list of command-line options
- First one (`sys.argv[0]`) is program name

```
import sys
print("Executing: %s" % (sys.argv[0]))
if len(sys.argv) < 2:
    print("Not enough parameters")
    sys.exit(1)
print("Parameters:")
print(", ".join(sys.argv[1:]))
```

- Parse parameters...
- Pro Tip: Use module `argparse`

List comprehension

- Short notation to create (complex) lists

```
even_squares = []
for i in range(10):
    if i%2 == 0:
        even_squares.append(i*i)

even_squares = [i**2 for i in range(10) if i%2==0]
```

Compare: $\{i^2 \mid i \in \{0,\ldots,9\}, i \text{ even}\}$

- Nesting: can contain more than one `for ... in ... [if ...]`

```
[(x,y.upper()) for x in range(4) if x%2 == 0
               for y in ["a", "b", "c"]]
```