Outline

- Follow Berkeley's CS61a course: Structure and Interpretation of Computer Programs
- The successor of the famous book and course from MIT – the first course on programming for any MIT student
- Originally taught in Scheme
- Lecture 1: Functions
- Lecture 2: Values
- Lecture 3: Objects
- Lecture 4: TBD (maybe scientific programming: matplotlib, numpy, scipy)
- The CS61a course has videos and many exercises
- Automated testing
- We will do a part in class – I'll will assign some homeworks
- We will check next time in class

Introductory explanations

Interpreter. - Denoted by the prompt `>>>` - The interpreter reads and executes what we type - To exit: `exit()` or `CTRL-D`

Expressions. - Primitive expressions: integers, float, booleans (other data types – to be covered later) - Numbers can be combined to form compound expressions using arithmetic operations (`+`, `-`, `*`, `/`, `//`)

```
-1 - -1
1 / 2 + 1 / 4 + 1 / 8 + 1 / 16
```

- Booleans can be combined using boolean operators: `and`, `or`, `not`
- Combining integers to obtain booleans

Call expressions. - Apply a function to some arguments

```
max(7.5, 9.5)
pow(2, 100)
max(1, -2, 3, -4)
max(min(1, -2), min(pow(3, 5), -4))
```

Importing library functions.

```
import math
math.sqrt(100)
```

```
from math import sqrt
sqrt(100)
from math import sqrt as s
s(100)
```

Variables. - A name bineds to a value

```
radius = 10
2 * radius
from math import pi
pi * 71 / 223
```

- Names can also be bound to functions

  ```
  f = max
  f
  f(2, 3, 4)
  ```

- Multiple assignments and swapping (all expression on the left are evaluated before any names to the left are bound to those values):

  ```
  x, y = 3, 4
  y, x = x, y
  x
  y
  ```

Defining functions. - Statement to define a function:

```
def <name>(<parameters>):
    return <expression>
```

- Note the indentation and the keywords: `def`, `return`

Control. - If statement

```
if <expr>:
    <statement>
    <statement>
    ...
elif <expr>:
    <statement>
    <statement>
    ...
else:
    <statement>
    <statement>
    ...
```

- While loop

  ```
  while <expr>:
  <statement>
  ```

```
<statement>
<statement>
...
```

Special topics. - Divison - Functions: indentation, return, print - Control: Short-circuiting - Error messages

- ? Special mention: += operator (L1-Q2)
- ? Doc strings (L1-Q3)