

# Program basics

- Variables, expressions and statements
- Conditionals
- Iteration
- Interacting with users
- Randomness

# Variables, expressions and statements

- expressions:
  - ~ nouns
  - of various types
- Statements
  - ~ sentences
- assignment
  - a kind of statement causing a variable to be bound to a value
  - multiple assignment!

```
def intDivBoth(x,y):  
    return x / y, x % y
```

```
dvsor, rem = intDivBoth(17,5)
```

# Conditionals

- if COND1:
  - code run when COND1 == True
- elif COND2:
  - code run when COND2 == True (but COND1 != True)
- else:
  - code run when neither COND1 nor COND2 == True
- Nested conditions
- True == 1; False==0
- Boolean expressions
  - and(CONJUNCTION1,CONJUNCTION2, ...)
  - or()
  - not()
    - not(7) == False
    - not(0) == True

# Iteration

- while COND
  - code run as long as COND == True
- for e in seq:
  - run code len(seq) times, with e bound to each element of s in order
- Premature exit
  - break
  - continue

# Interacting with users

- Asking with clear prompts
- Checking their input
  - Types
  - Ranges
- Confirming what you heard

# Randomness

- Pseudo-random number generation
- Controlling the randomness
- Getting random numbers
- Getting other forms of randomness

# Pseudo-random number generation

- Deterministic algorithm for generating a long sequence of numbers before repeating
- Beginning from an initial “seed” (starting point)
  - Used to reproduce stochastic experiments exactly!
- Python (2.3) uses Mersenne Twister
  - Period =  $2^{19937}-1$
  - Implemented in C

# Controlling the randomness

- `random.seed(i)`
  - Integer used to initialize the generator
- `random.seed()`
  - Current system time used
- `random.getstate()`
  - returns current state of generator
- `random.setstate(x)`

# Getting random numbers

- `random.randint( from, to)`
  - Random integer within range (inclusive)
- `random.random( )`
  - Random float within range [0.0, 1.0)
  - ie, NOT 1.0!

# Getting other forms of randomness

- `choice(seq)`
  - some element of `seq`
- `shuffle(seq)`
  - permutation of `seq`
- `sample(pop, k)`
  - `k` elements of `pop` drawn without replacement
- `sample(xrange(10000000), 60)`

# Functions

- Local variables
  - Defined only within “scope” of function
- Recursion
  - self-referential definitions
  - Note connection to mathematical **INDUCTIVE** proofs!
    - Base case
    - “leap of faith” : Assume true for  $k < N$
- Methods
  - Functions associated with objects

# Modules

- Module basics
- Referencing modules' names
- Module search path
- Other module details

# Module basics

- a file containing Python
- function definitions
- initializing statements
- Accessed by other modules via `import`

# Referencing modules' names

- Names within other modules must be qualified
- by explicit prefix of module name

```
import random
i = random.randint()
```
- by direct reference

```
from random import randint
i = randint()
```

# Module search path

- Where to search for imported module files
- Current directory
- Environment variable PYTHONPATH
- maintained in variable `sys.path`

## Other module details

- `dir(module)` lists its names
- Python's "built in" names in `__builtin__` module
- Packages
  - Hierarchic naming corresponding to subdirectories