ENGR 101
Engineering Design Workshop

Lecture 3: Modules, Loops: while, for
Edgardo Molina

City College of New York
Modules
Modules

• Python Strength: large collection of open source modules

• Modules are collections (packages) of useful (tested) code you can reuse

• Useful modules we’ve seen: random, math, turtle
Modules

- **Python Standard Library** (packages included with most python distributions)
  - [http://docs.python.org/library/index.html](http://docs.python.org/library/index.html)

- **PyPI** (Python Package Index)
  - [http://pypi.python.org/pypi](http://pypi.python.org/pypi)
  - repository of optional modules available (11,000+)
Using Modules

• Math module contains many useful functions and values: math.sin, math.cos, math.pow, math.sqrt, ...

• Using modules:

```python
import math
value = math.sin(math.pi)
```

```python
import random
x = random.randint(1,6)
```
Getting help

• In python interpreter you can get documentation

```python
>>> import math
>>> import random
>>> help(math.sin)
...
>>> help(random.randint)
...
```
Control Structures: Repetition, Iteration
Repetition

* selection statements (if-elif-else) let us make simple decisions

* repeating statements and decisions let us build more complex programs
While the Boolean expression is True, keep looping.
Testing Primeness

number = input("What number should I check for primeness? ")

#initialize
is_number_prime = True

# loop parameters
start = 2; end = int(number); step = 1

# initialize loop
x = start

# test
while x < end:
    if end % x == 0:
        is_number_prime = False
    # increment
    x += step

# print result
if is_number_prime:
    print("The number", x, "is a prime number.")
else:
    print("The number", x, "is not a prime number.")
break statement

while test1:
    Do some stuff
    if test2:
        break
    Do some other stuff
Break takes you here

• break immediately ends loops

• DO NOT overuse; Can be difficult to read/understand logic
Testing Primeness

```python
number = input("What number should I check for primeness? ")

#initialize
is_number_prime = True

# loop parameters
start = 2; end = int(number); step = 1

# initialize loop
x = start

# test
while x < end:
    if end % x == 0:
        is_number_prime = False
        break
    # increment
    x += step

# print result
if is_number_prime:
    print("The number", x, "is a prime number.")
else:
    print("The number", end, "is not a prime number.")
```

Stop checking
range(...)

- Built in function; produces a sequence (list)

  - range(0, 3) → [0, 1, 2]
  - range(3) → [0, 1, 2]
  - range(1, 3) → [1, 2]
The for loop is used for iteration

```python
for anElement in list:
    Do some stuff
    # continue with next element
```
continue statement

while test1:
    Do some stuff
    if test2:
        continue
        skipped if test2 is true
Break takes you here

- break and continue work in both while and for loops
Find all primes

```python
number = int(input("Find all primes up to: "))

for element in range(2, number):
    # initialize
    is_number_prime = True

    # loop parameters
    start = 2; end = element; step = 1

    # initialize loop
    x = start

    # test
    while x < end:
        if end % x == 0:
            is_number_prime = False
            break
        # increment
        x += step

    # print primes only
    if is_number_prime:
        print("The number", x, "is a prime number.")
```