§5.9: Sieve of Eratosthenes (example of arrays)

- *Quiz ch4-5 today*

2 Oct 2006
CMPT14x
Dr. Sean Ho
Trinity Western University

Announcements

- M lab (Edie) and W lab (Jenna): TAs will be out

Review last time (§5.5-5.10, Py ch8)

- Python lists vs. M2/C arrays
- Lists as function parameters
- Multidimensional arrays/lists
- Python-specific list operations
  - Membership (in)
  - Concatenate (+), repeat (*)
  - Delete (del), slice ([s:e])
  - Aliasing vs. copying lists

Quiz ch4-5

- Name two standard container (aggregate) types in Python.
- Name two operations/functions/properties that Python lists have that M2/C arrays do not.
- Write a Python function
  `create_matrix(n_rows, n_cols)`
  that returns a new matrix of the specified size.
  - Contents of the matrix don't matter
  - Docstring required!
  - Partial credit for pseudocode

Quiz ch4-5 answers: #1

- Name two standard container (aggregate) types in Python.
  - **Immutable sequences**
    - **Strings (str):** **"Hello"**
    - **Tuples (tuple):** **(2, 5.0, "hi")**
  - **Mutable sequences**
    - **Lists (list):** **[2, 5.0, "hi"]**
  - **Mappings**
    - **Dictionaries (dict):** **{"apple": 5, "orange": 8}**

Quiz ch4-5 answers: #2

- Name two operations/functions/properties that Python lists have that M2/C arrays do not.
  - Can change length dynamically
  - Items need not all be same type
  - Concatenate (+), repeat (*)
  - List membership test (in)
  - Insert, delete (del)
  - List slice ([:])
  - …

Quiz ch4-5 answers: #3

- Create a matrix:
  - Use range() to create a 1D list, then
  - Use range() to turn each element into a whole row:

```
def create_matrix(n_rows, n_cols):
    """Create a new 2D list of given size.
    pre: n_rows, n_cols are integers > 0.
    """
    matrix = range(n_rows)
    for row in range(n_rows):
        matrix[row] = range(n_cols)
    return matrix
```

Problem statement: list primes

- Problem: list all the prime numbers between 2 and some given big number.
  - You had a homework that was similar: test if a given number is prime, and list its factors
  - How did you solve that?
    - Procedure is_prime() (pseudocode):

```
Iterate for factor in 2 .. sqrt(n):
    If (n % factor == 0), then
            We've found a factor!
```

- But this is wasteful: really only need to test prime numbers for potential factors

Listing all primes

- We could tackle this problem by repeatedly calling is_prime() on every number in turn:

```
for num in range(2, max):
    if is_prime(num) ...
```

- But this could be really slow if max is big

Sieve of Eratosthenes

- The sieve works by a process of elimination: we eliminate all the non-primes by turn:

$\cancel{1}$	$\boxed{2}$	$\boxed{3}$	$\cancel{4}$	$\boxed{5}$	$\cancel{6}$	$\boxed{7}$	$\cancel{8}$	$\cancel{9}$	$\cancel{10}$
$\boxed{11}$	$\cancel{12}$	$\boxed{13}$	$\cancel{14}$	$\cancel{15}$	$\cancel{16}$	$\boxed{17}$	$\cancel{18}$	$\boxed{19}$	$\cancel{20}$
$\cancel{21}$	$\cancel{22}$	$\boxed{23}$	$\cancel{24}$	$\cancel{25}$	$\cancel{26}$	$\cancel{27}$	$\cancel{28}$	$\boxed{29}$	$\cancel{30}$
$\boxed{31}$	$\cancel{32}$	$\cancel{33}$	$\cancel{34}$	$\cancel{35}$	$\cancel{36}$	$\boxed{37}$	$\cancel{38}$	$\cancel{39}$	$\cancel{40}$
$\boxed{41}$	$\cancel{42}$	$\boxed{43}$	$\cancel{44}$	$\cancel{45}$	$\cancel{46}$	$\boxed{47}$	$\cancel{48}$	$\cancel{49}$	$\cancel{50}$
$\cancel{51}$	$\cancel{52}$	$\boxed{53}$	$\cancel{54}$	$\cancel{55}$	$\cancel{56}$	$\cancel{57}$	$\cancel{58}$	$\boxed{59}$	$\cancel{60}$
$\boxed{61}$	$\cancel{62}$	$\cancel{63}$	$\cancel{64}$	$\cancel{65}$	$\cancel{66}$	$\boxed{67}$	$\cancel{68}$	$\cancel{69}$	$\cancel{70}$
$\boxed{71}$	$\cancel{72}$	$\boxed{73}$	$\cancel{74}$	$\cancel{75}$	$\cancel{76}$	$\cancel{77}$	$\cancel{78}$	$\boxed{79}$	$\cancel{80}$
$\cancel{81}$	$\cancel{82}$	$\boxed{83}$	$\cancel{84}$	$\cancel{85}$	$\cancel{86}$	$\cancel{87}$	$\cancel{88}$	$\boxed{89}$	$\cancel{90}$
$\cancel{91}$	$\cancel{92}$	$\cancel{93}$	$\cancel{94}$	$\cancel{95}$	$\cancel{96}$	$\boxed{97}$	$\cancel{98}$	$\cancel{99}$	$\cancel{100}$

Prime sieve: pseudocode

1) Create an array of booleans and set them all to true at first. (true = prime)
2) Set array element 1 to false. Now 2 is prime.
3) Set the values whose index in the array is a multiple of the last prime found to false.
4) The next index where the array holds the value true is the next prime.
5) Repeat steps 3 and 4 until the last prime found is greater than the square root of the largest number in the array.

Prime sieve: Python code

```
"""Find all primes up to a given number, using
   Eratosthenes' prime sieve."""
import math                                    # sqrt
size = input("Find all primes up to: ")

# Initialize: all numbers except 0, 1 are prime
primeFlags = range(size+1)         # so pF[size] exists
for num in range(size+1):
    primeFlags[num] = True

primeFlags[0] = False
primeFlags[1] = False
```

Prime sieve: Python code (p.2)

```
# Computation: eliminate all non-primes
for num in range(2, int(math.sqrt(size))+1):
    if primeFlags[num]:            # got a prime
        # Eliminate its multiples
        for multiple in range(num**2, size+1, num):
            primeFlags[multiple] = False

# Output
print "Your primes, sir/madam:",
for num in range(2, size+1):
    if primeFlags[num]:
        print num,
```

http://twu.seanho.com/python/primesieve.py