Contents

#**-- mode: org --**

1 Measuring Execution Time

- It's difficult to measure correctly.
- Your computer is doing too much.
- I/O problems.
- May be too fast, hard to measure correctly.
- Run out of memory.

1.1 How to Time Correctly

- TIME MULTIPLE RUNS!
- Statistical significance: 29 or more.
- Use high resolution timers:
  - Use the smallest measure of time possible.
- Only time the important stuff.

1.2 C++11 Example

The following is c++11 only, it uses a high resolution timer rather than tick from C/C++.

```
double time_function()
{
  clock_t start, end, total;
  start = clock();
  for (int i=0; i<ITERS; i++)
    function_to_time();
  end = clock();
  total = end - start;;
  return (total / (float)CLOCKS_PER_SEC) / ITERS;
}
```

1.3 Java Example

```
public static float time_method()
{
  long start, end, total;
  start = System.nanoTime();
  // ITERS defined elsewhere
  for (int i=0; i<ITERS; i++)
    method_to_time()
  end = System.nanoTime();
  total = end = start;
  return total / (float) ITERS;
}
```

1.4 Python Example

Refer to python's Timeit Module (3.7).

2 Math Review

2.1 FLOOR: Largest int that is <= x.

- floor(3.2) => 3
- floor(-6.7) => -7

2.2 CEILING: Smallest int that is >= x.

- ceil(3.2) => 4
- ceil(-6.7) => -6

2.3 LOG

- Let b > 1, x > 0
- Logb(x) = L iff b**L = x
- Log10(1000) = 3
- Log2(8) = 3
- lg = log2
- lg(16) => 4
- floor(lg(10)) => 3

2.4 Log identities

- logb(1) => 0
- logb(b) => 1
- logb(x*y) => logb(x)+logb(y)
- logb(x/y) => logb(x)-logb(y)
- logb(x**g) => glogb(x)
- loga(x) => logb(x)/logb(a) [!]

2.5 ADDITION

- 1 + 2 + 3 + ... + n => sum(1, n, i)
- sum(1, n, 1) => n
- sum(1, n, i) => n(n+1)/2
- sum(1, n, i*i) => [n(n+1)(2n+1)]/6
- sum(1, n, c**i) => (c**(i+1)-1)/(c-1) while c is not 1