CLOUD COMPUTING CONCEPTS

with Indranil Gupta (Indy)

AN ORIENTATION TO CLOUD COMPUTING

What this Lecture is about

- Covers basic concepts in Computer Science that will be assumed in the Cloud Computing Concepts (C3) course
- For those of you already familiar, it’s a refresher
- Use this as reference if you don’t understand (during the course) how the basics are being used

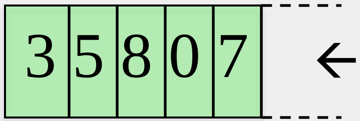
What's in this Lecture

I. Basic datastructures
II. Processes
III. Computer architecture
IV. O() notation
V. Basic probability
VI. Miscellaneous

I. Basic Datastructures: Queue

- Queue: First-in First-out datastructure

Remove from head ←



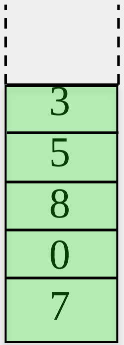
← **Insert** at tail

- Next item dequeued (removed) is 3.
  - Then 5
  - Then 8
  - And so on

Basic Datastructures: Stack

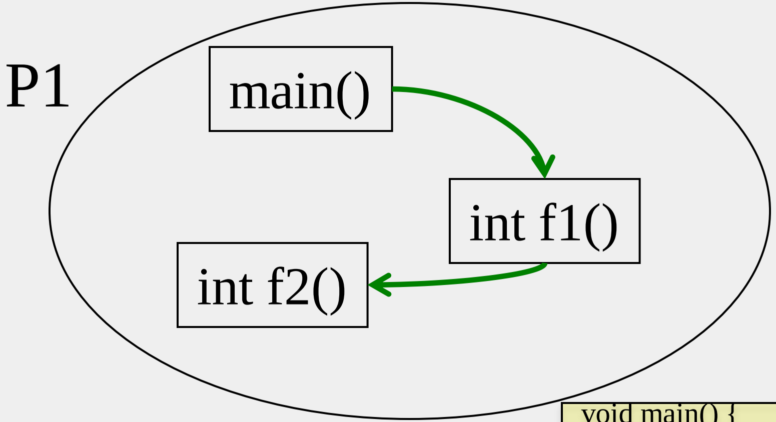
- Stack: First-in *Last-out* datastructure

Remove (Pop) from top ← ← **Insert (Push)** at top



- Insert (Push) 9: goes to top
- Remove (Pop): gets 9
- Pop: gets 3
- Next pop: gets 5 (and so on)

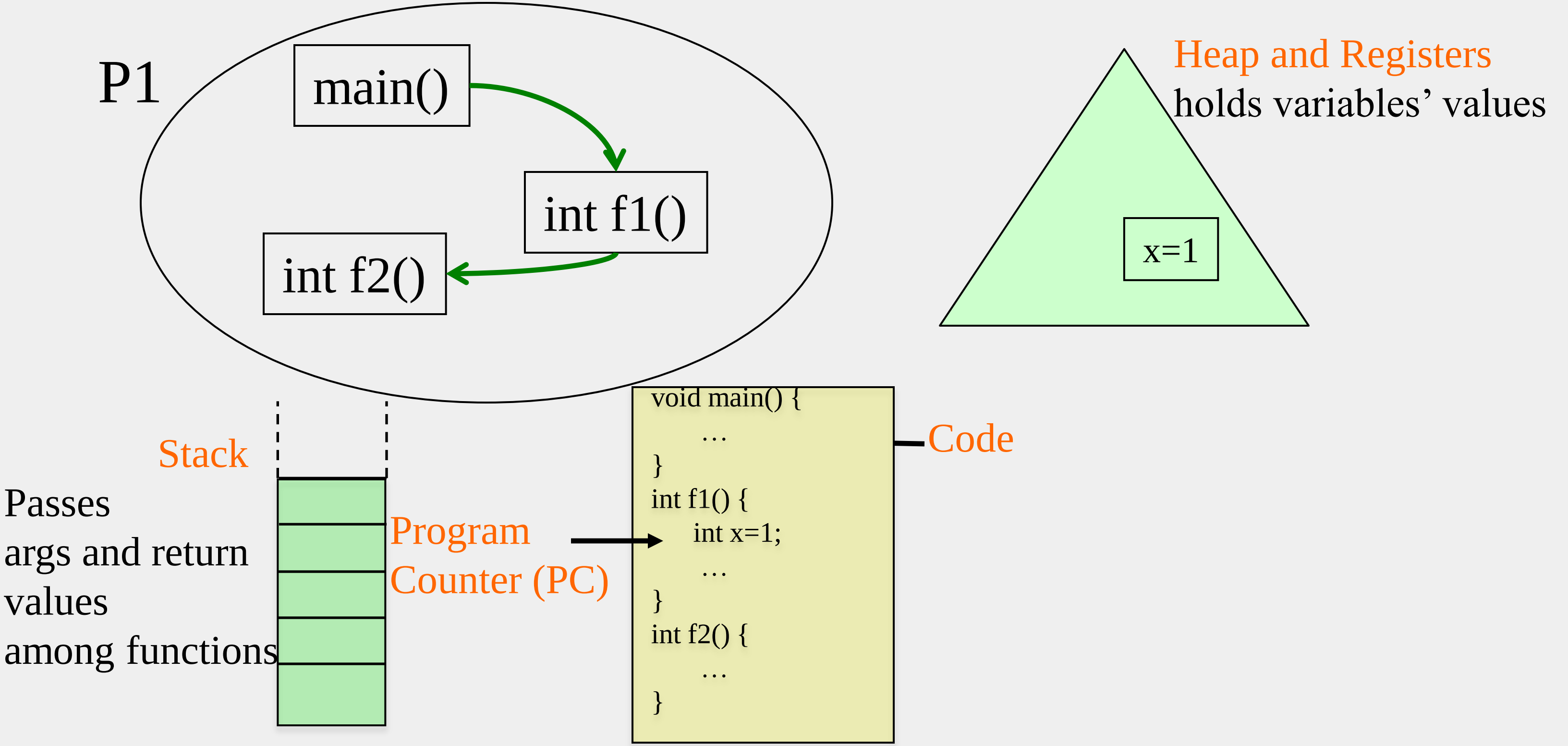
II. Process = a Program in Action



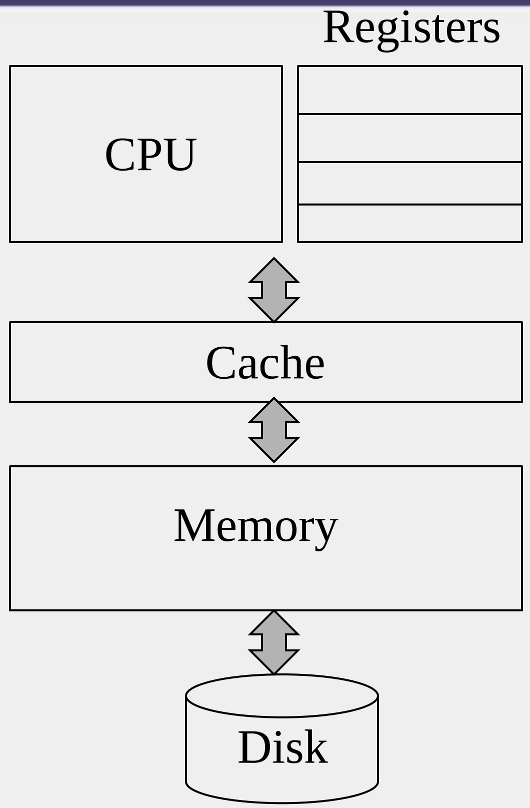

```
void main() {
      …
}
int f1() {
      int x=1;
      …
}
int f2() {
      …
}
```

Code

INSIDE A PROCESS



III. Computer Architecture (Simplified)



Computer Architecture (2)

- A program you write (C++, Java, etc.) gets compiled to low-level machine instructions
  - Stored in file system on disk
- CPU loads instructions in batches into memory (and cache, and registers)
- As it executes each instruction, CPU loads data for instruction into memory (and cache, and registers)
  - And does any necessary stores into memory
- Memory can also be flushed to disk
- This is a highly simplified picture!
  - (but works for now)

IV. Big O() Notation

- One of the most basic ways of analyzing algorithms
- Describes *upper bound* on behavior of algorithm as some variable is scaled (increased) to infinity
- Analyzes run-time (or another performance metric)
- Worst-case performance

Big O() Notation: Informal Definition

- "An algorithm A is O(foo)"

  Means

- "Algorithm A takes < c * foo time to complete, for some constant c, beyond some input size N"
- Usually, foo is a function of input size N
  - e.g., an algorithm is O(N)
  - e.g., an algorithm is O(N^2)
- We don't state the constants in Big O() notation

Big O() Notation: Example 1

- “Searching for an element in an unsorted list is O(N), where N = size of list”
- Have to iterate through list
- Worst-case performance is when that element is not there in the list, or is the last one in the list
- Thus involves N operations
- Number of operations < c* N, where c=2.

Big O() Notation: Example 2

- "Insertion sorting of an unsorted list is $O(N^2)$, where N = size of list"
- Insertion sort Algorithm:
  - Create new empty list
  - For each element in unsorted list
    - Insert element into sorted list at appropriate position
- First element takes 1 operation to insert
- Second element takes (in worst case) 2 operations to insert
- i-th element takes i operations to insert
- Total time = $1+2+3+\ldots+N=N(N+1)/2 < 1*N^2$

V. Basic Probability

- Set=collection of things
  - S="Set of all humans who live in the world"
- Subset=collection of things that is part of a larger set
  - S2="Set of all humans who live in Europe"
  - S2 is a subset of S

Basic Probability

- Any event has a probability of happening
- If you wake up at a random hour of the day, what is the probability of the event that the time is between 10 am and 11 am?
- There are 24 hours in a day
  - Set of hours contains 24 elements: 12 am, 1 am, 2 am, … 10 am, 11 am, …11 pm
- You pick one hour at random
- Probability you pick 10 am = 1/24

Multiplying Probabilities

- E1 is an event
- E2 is an event
- E1 and E2 are <u>independent</u> of each other
- Then: Prob(E1 AND E2) = Prob(E1) * Prob(E2)
- You have three shirts: blue, green, red
- You wake up at a random hour and blindly pick a shirt
- Prob(You woke up between 10 am and 11 am AND that you're wearing a green shirt) = (1/24) * (1/3) = 1/72
- But beware: can't multiply probabilities if events are dependent (i.e., influence each other)!

Adding Probabilities

- E1 is an event
- E2 is an event
- Then:

Prob(E1 OR E2) = Prob(E1) + Prob(E2) – Prob(E1 AND E2)

- If you don't know Prob(E1 AND E2), then you can write

Prob(E1 OR E2) ≤ Prob(E1) + Prob(E2)

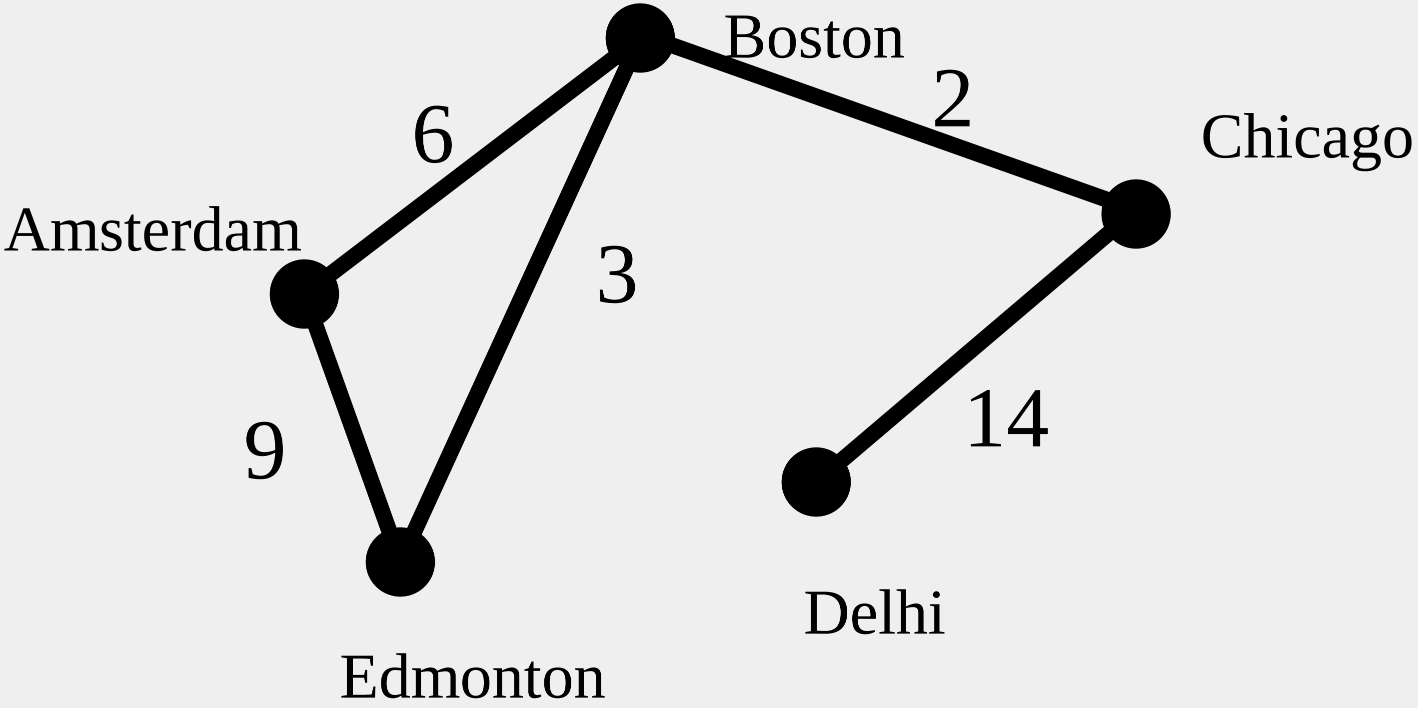
VI. DNS

- DNS = Domain Name System
- Collection of servers, throughout the world
- Input to DNS: a URL, e.g., coursera.org
  - URL is a name, a human-readable string that uniquely identifies the object

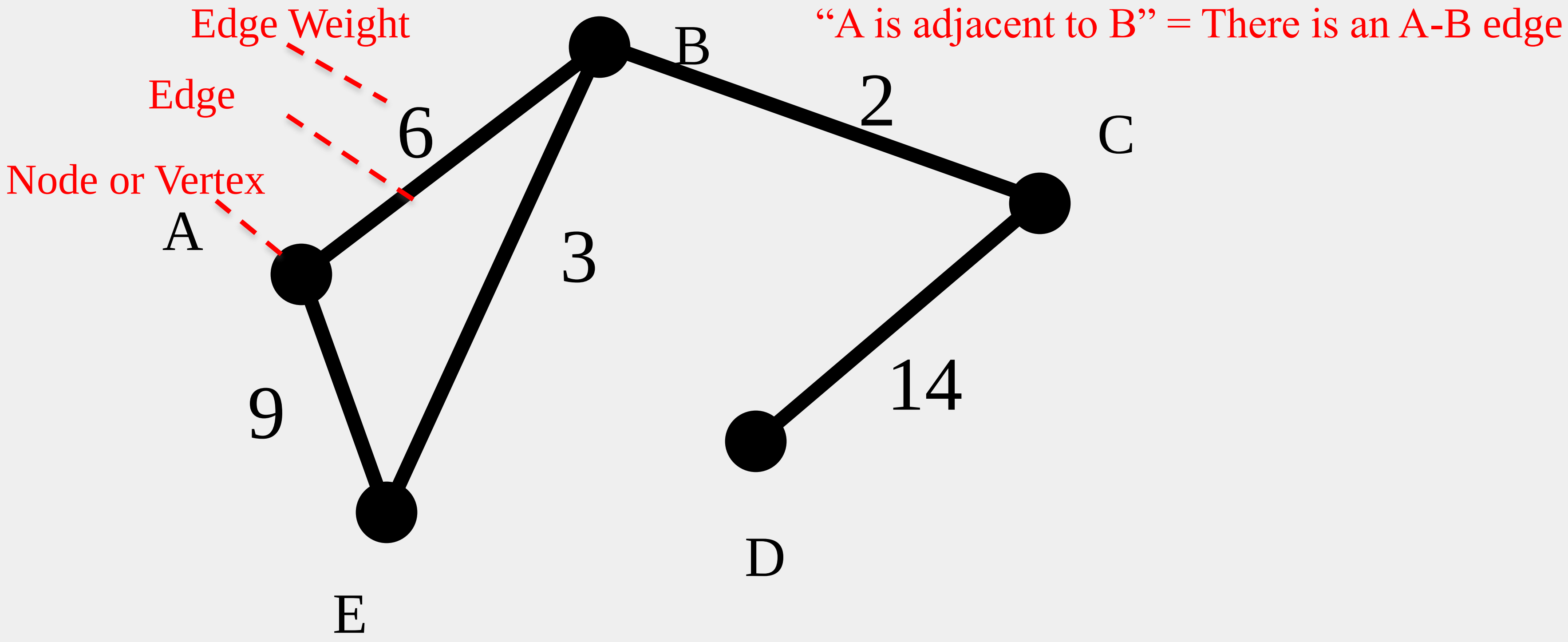
DNS (2)

- Output from DNS: IP address of a web server that hosts that content
  - IP address is an ID, a unique string pointing to the object. May not be human readable.
- IP address may refer to either
  - Web server actually hosting that content, or
  - An indirect server, e.g., a CDN (content distribution network) server, e.g., from Akamai

VII. Graphs



Graphs (2)



Edge Weight
Edge
Node or Vertex
"A is adjacent to B" = There is an A-B edge
A
B
C
D
E
6
2
3
9
14

What We Covered

I. Basic datastructures
II. Processes
III. Computer architecture
IV. O() notation
V. Basic probability
VI. Miscellaneous