Operating Systems

Lecture 2: The Process

Nipun Batra
Aug 3, 2018

OS Design Goals

OS Design Goals

1. High performance -> Minimize OS overheads

OS Design Goals

1. High performance -> Minimize OS overheads
    1. Extra memory

OS Design Goals

1. High performance -> Minimize OS overheads
    1. Extra memory
    2. Extra CPU

OS Design Goals

1. High performance -> Minimize OS overheads
    1. Extra memory
    2. Extra CPU
    3. Extra disk

OS Design Goals

1. High performance -> Minimize OS overheads
   1. Extra memory
   2. Extra CPU
   3. Extra disk
2. Protecting applications from one harming another and the OS -> Isolation

OS Design Goals

1. High performance -> Minimize OS overheads
    1. Extra memory
    2. Extra CPU
    3. Extra disk
2. Protecting applications from one harming another and the OS -> Isolation
3. Reliability

OS Design Goals

1. High performance -> Minimize OS overheads
    1. Extra memory
    2. Extra CPU
    3. Extra disk
2. Protecting applications from one harming another and the OS -> Isolation
3. Reliability
    1. Imagine sitting in a flight and the OS crashing!

OS Design Goals

1. High performance -> Minimize OS overheads
    1. Extra memory
    2. Extra CPU
    3. Extra disk
2. Protecting applications from one harming another and the OS -> Isolation
3. Reliability
    1. Imagine sitting in a flight and the OS crashing!
    2. Or, dispensing cash in an ATM and the OS crashing!

OS Design Goals

1. High performance -> Minimize OS overheads
    1. Extra memory
    2. Extra CPU
    3. Extra disk
2. Protecting applications from one harming another and the OS -> Isolation
3. Reliability
    1. Imagine sitting in a flight and the OS crashing!
    2. Or, dispensing cash in an ATM and the OS crashing!
    3. Or, the MRI scan machine OS reboots on its own!

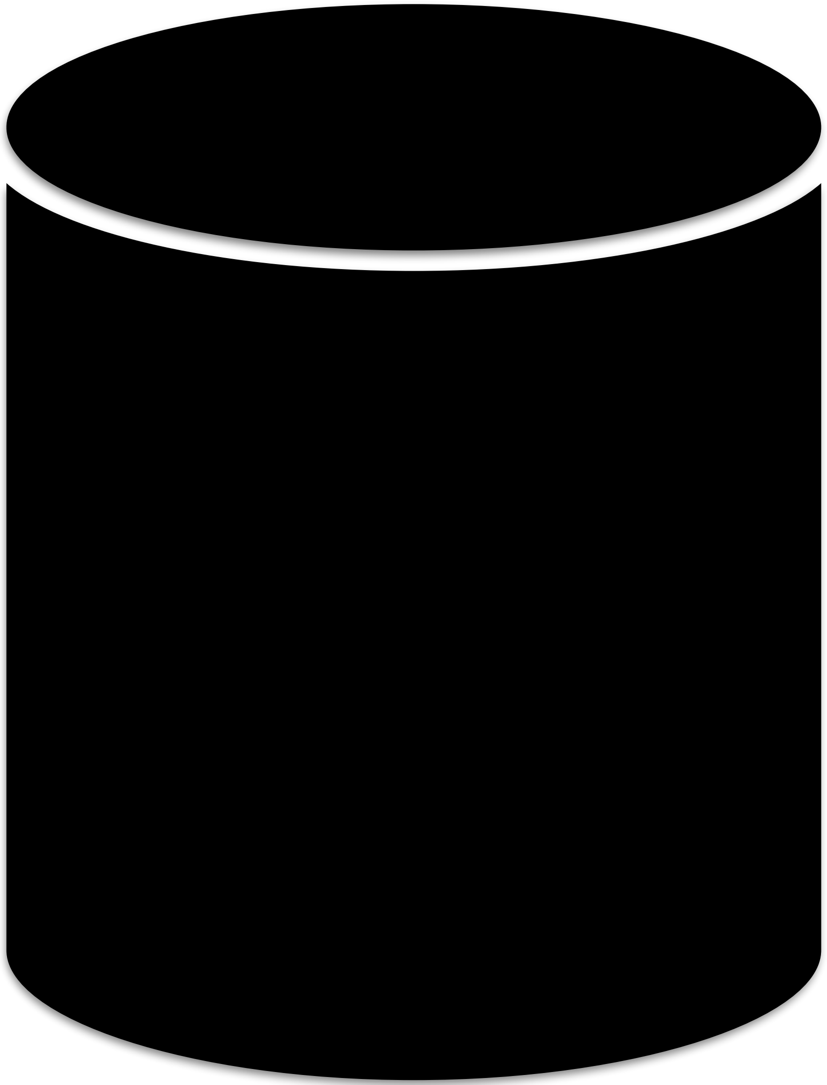
OS Design Goals

1. High performance -> Minimize OS overheads
    1. Extra memory
    2. Extra CPU
    3. Extra disk
2. Protecting applications from one harming another and the OS -> Isolation
3. Reliability
    1. Imagine sitting in a flight and the OS crashing!
    2. Or, dispensing cash in an ATM and the OS crashing!
    3. Or, the MRI scan machine OS reboots on its own!
4. Energy efficiency (esp. for mobile systems!)

Process

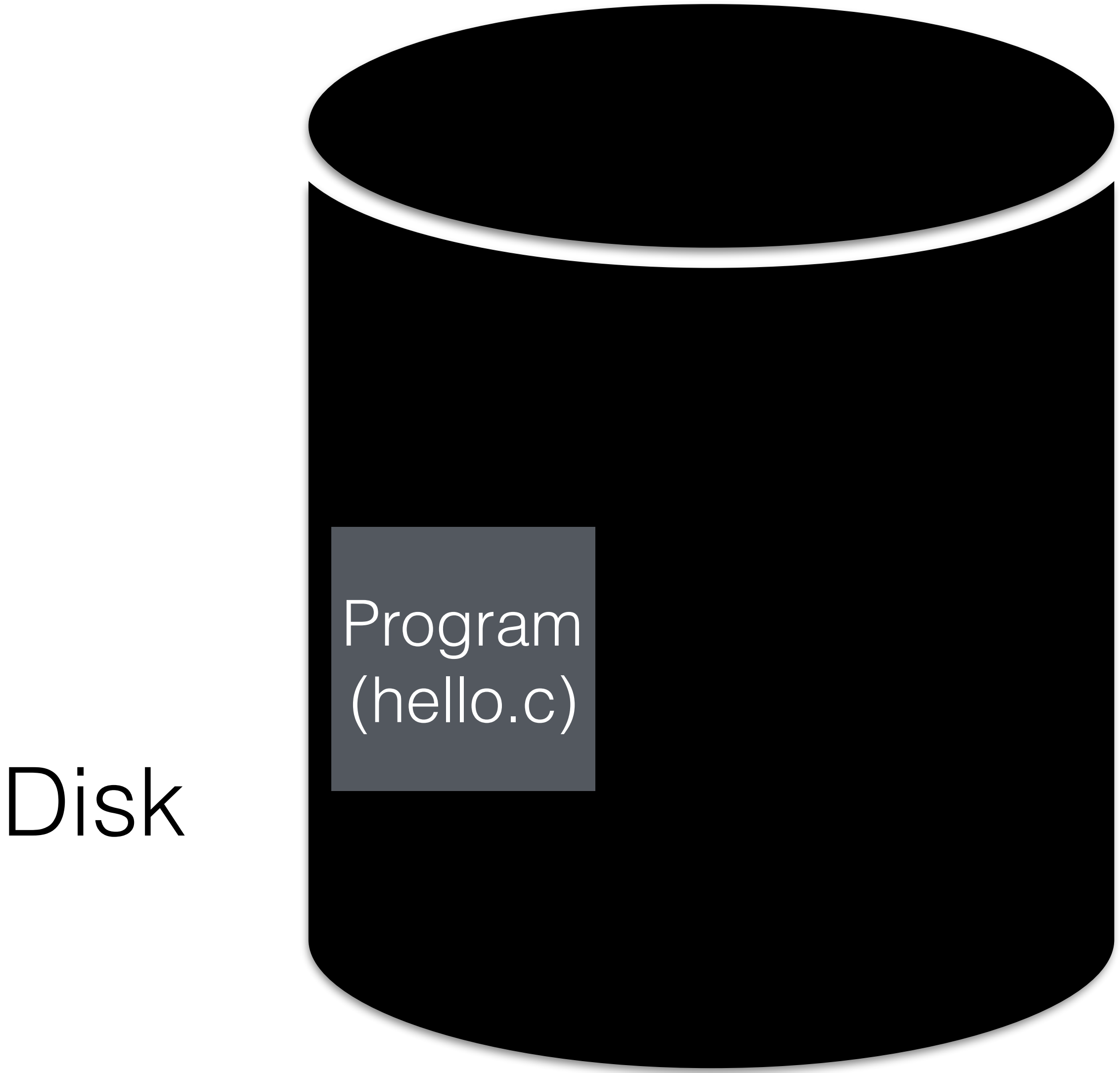
- Process = Running program
- Review example from previous lecture
  - Output of top
  - Activity monitor

Process Execution

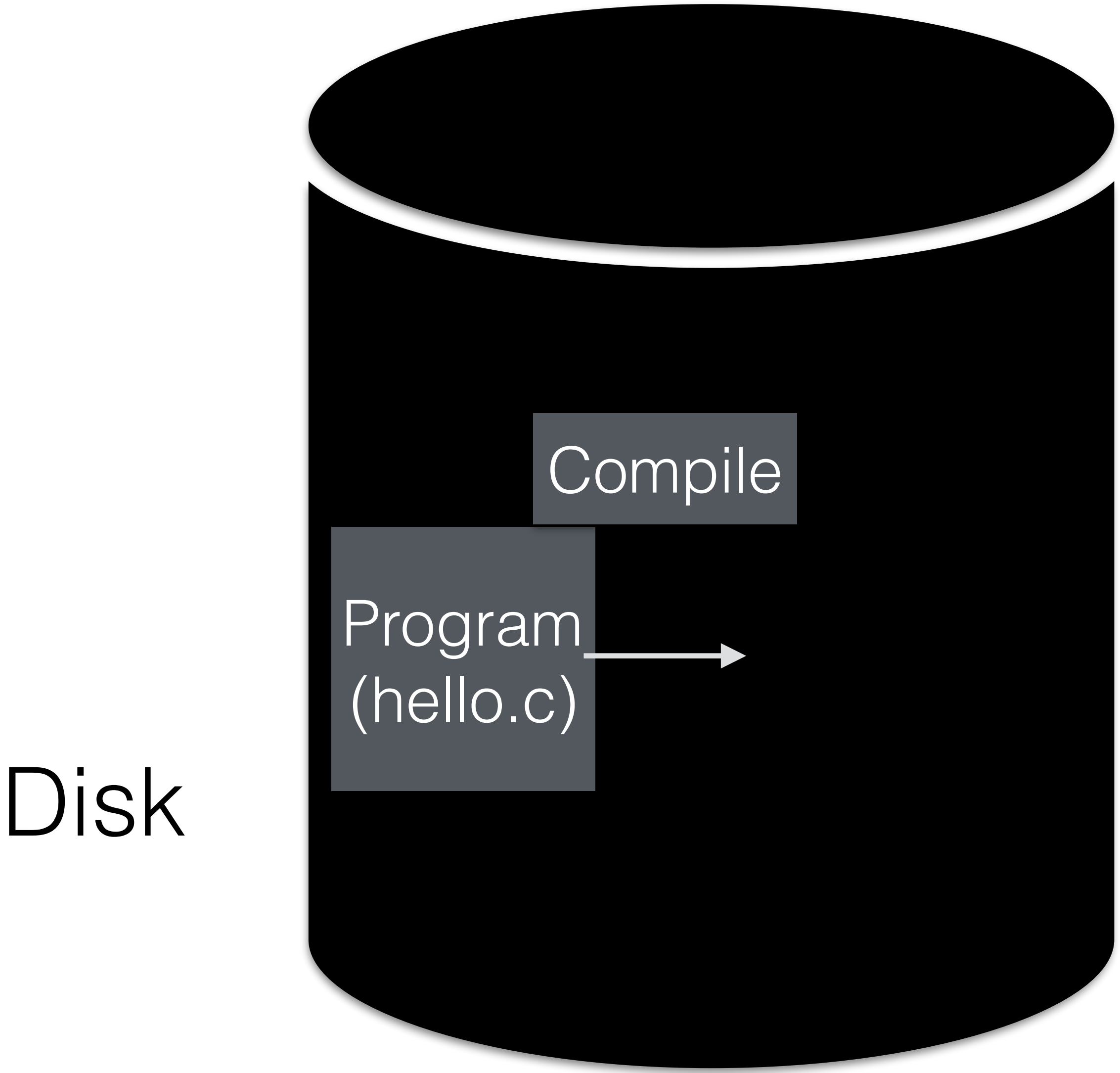


Disk

Process Execution



Process Execution

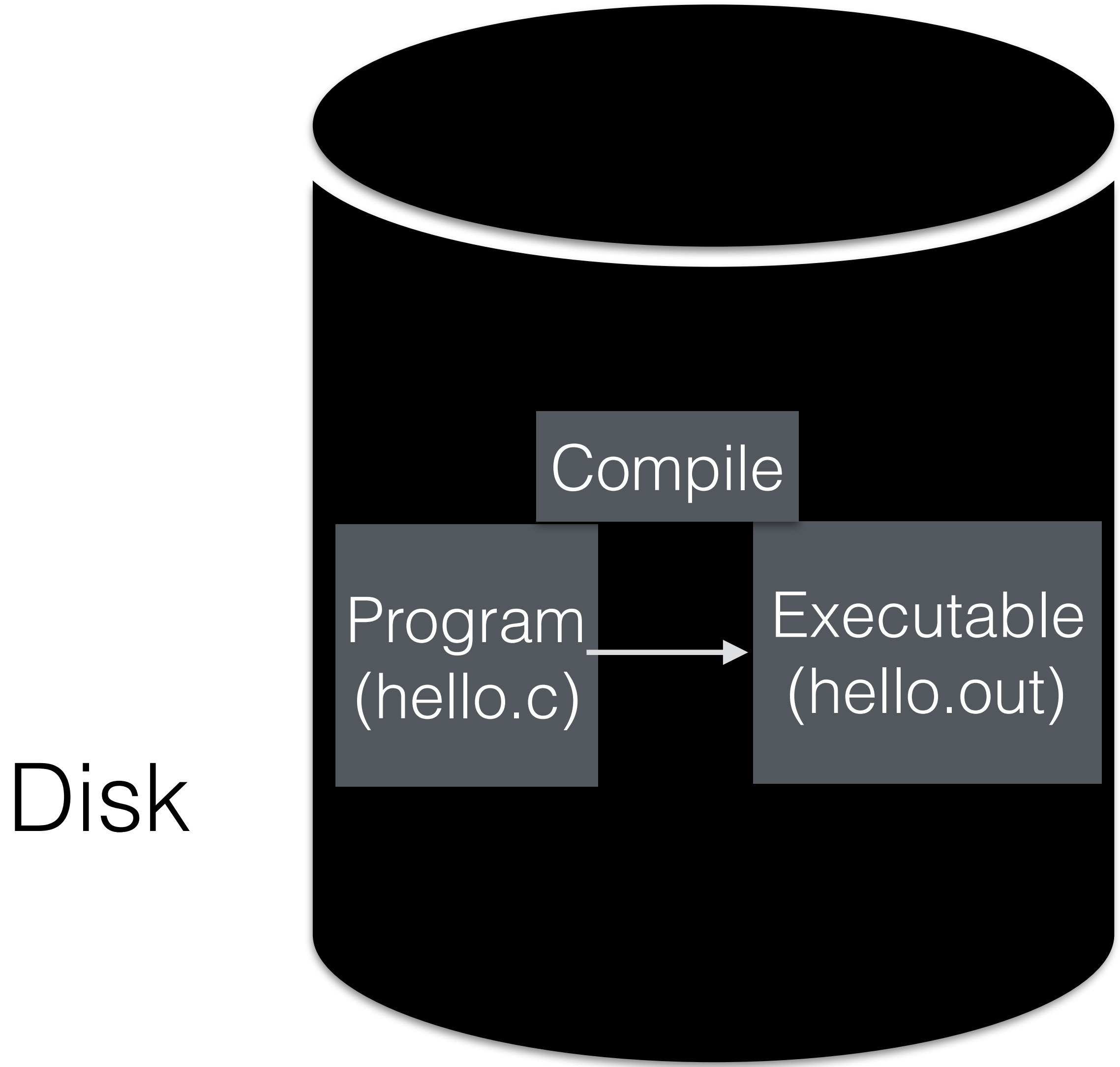


Disk

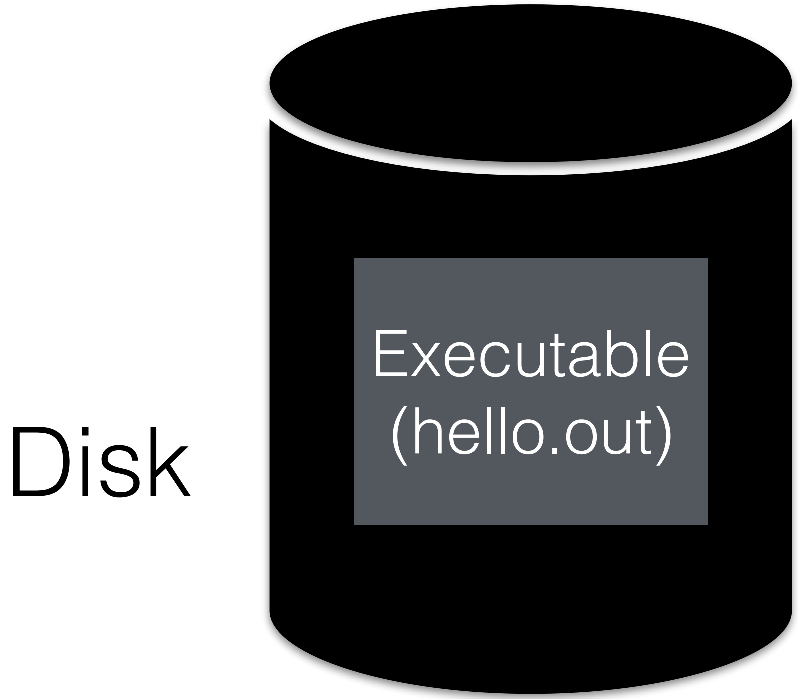
Compile

Program
(hello.c)

Process Execution

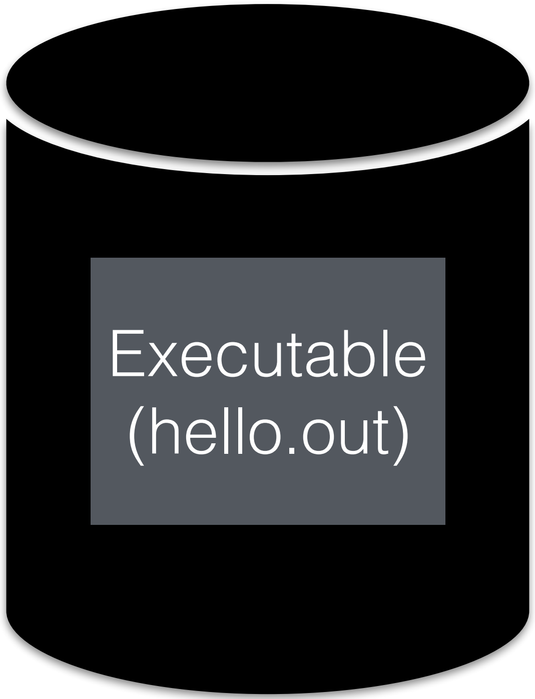


Process Execution

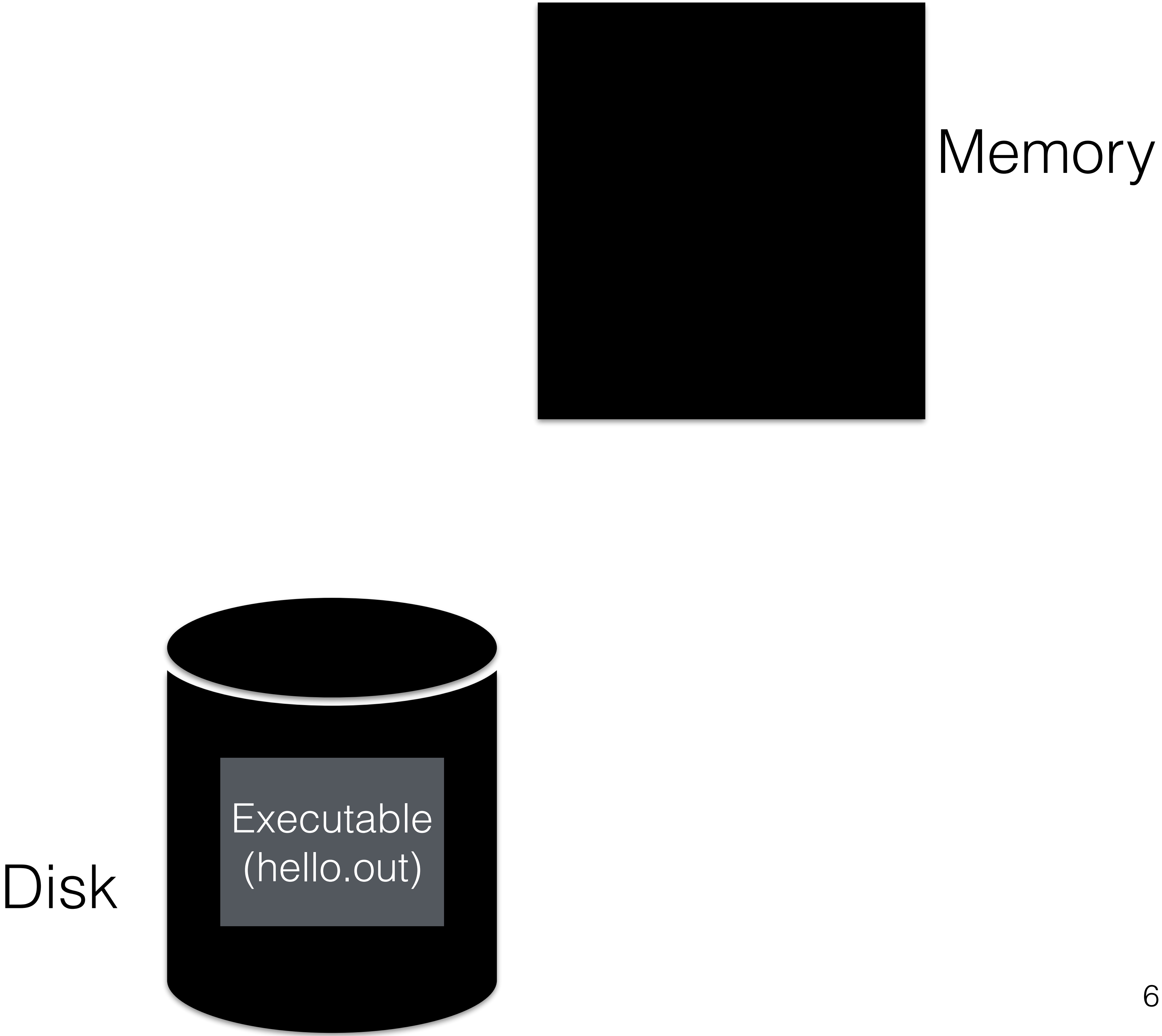


Process Execution

Disk



Process Execution

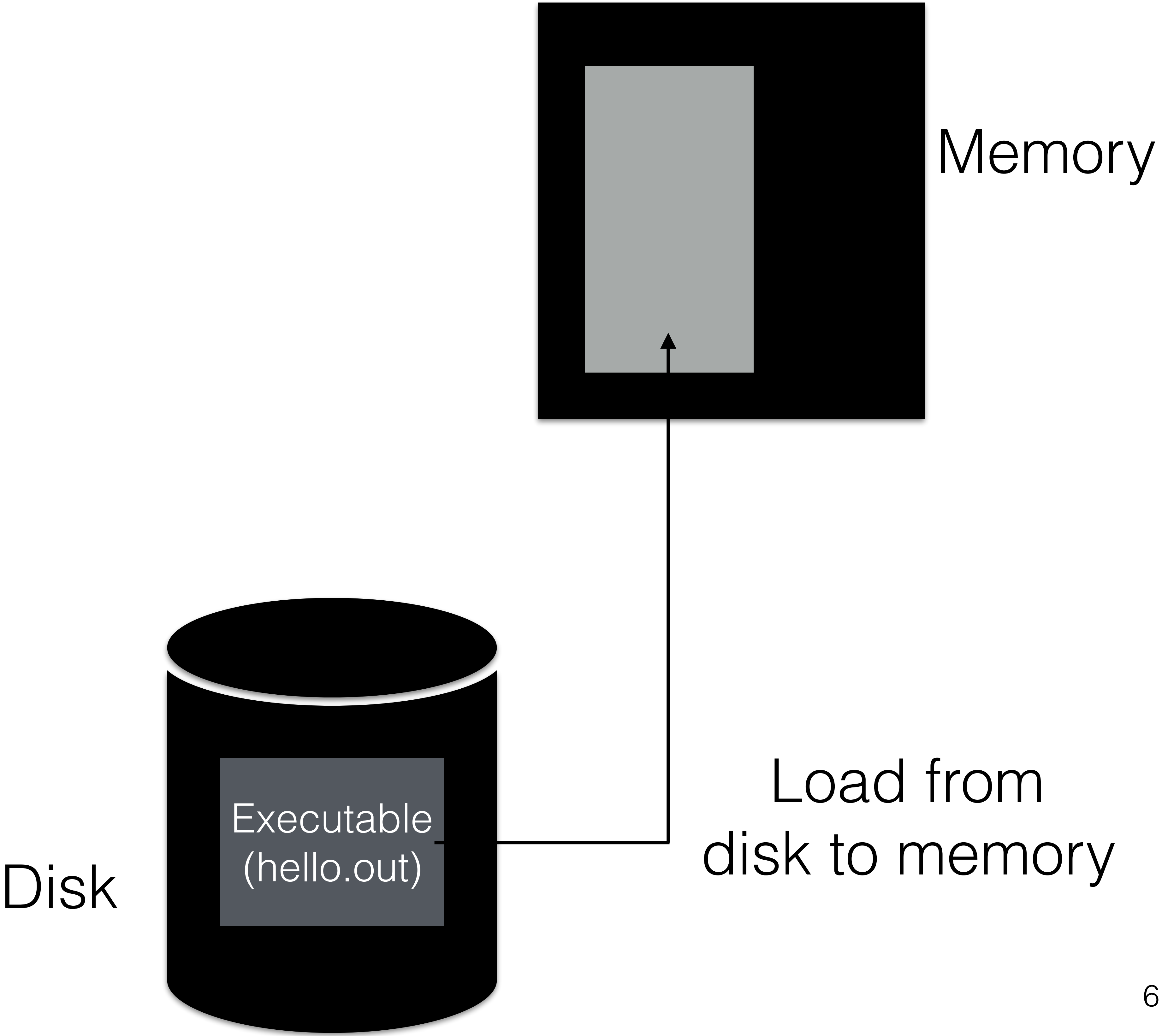


Memory

Disk

Executable (hello.out)

Process Execution



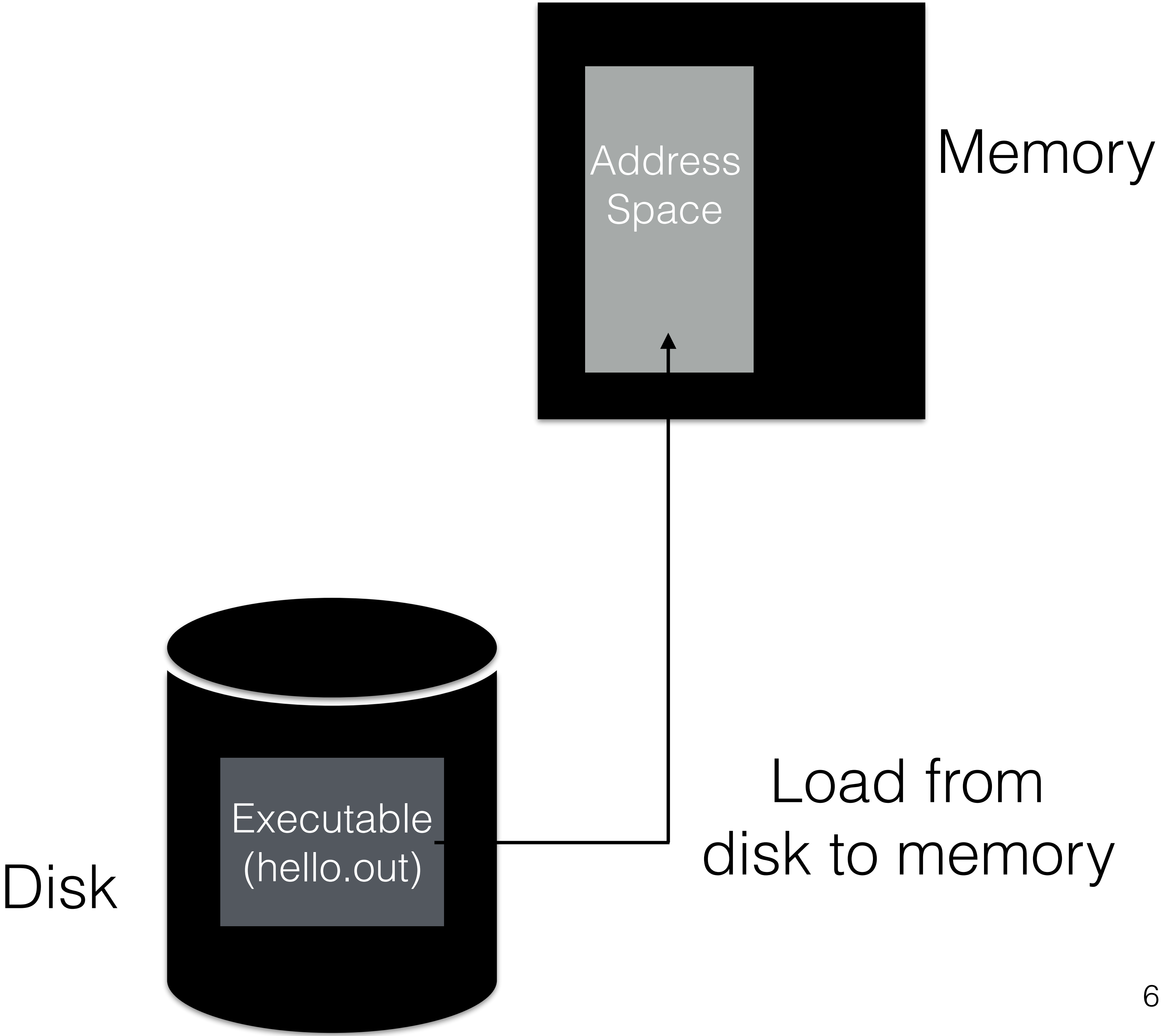
Memory

Executable (hello.out)

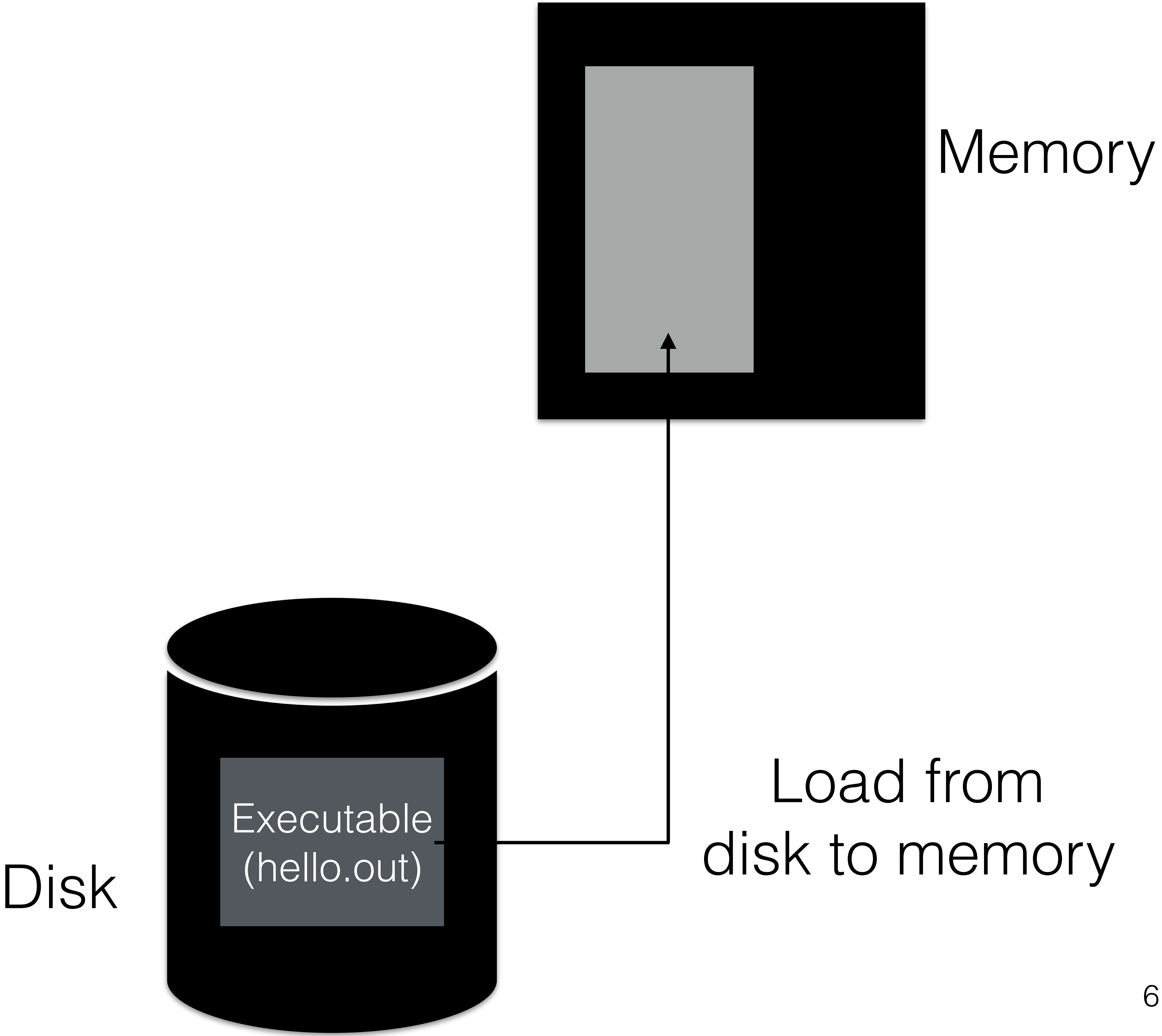
Disk

Load from disk to memory

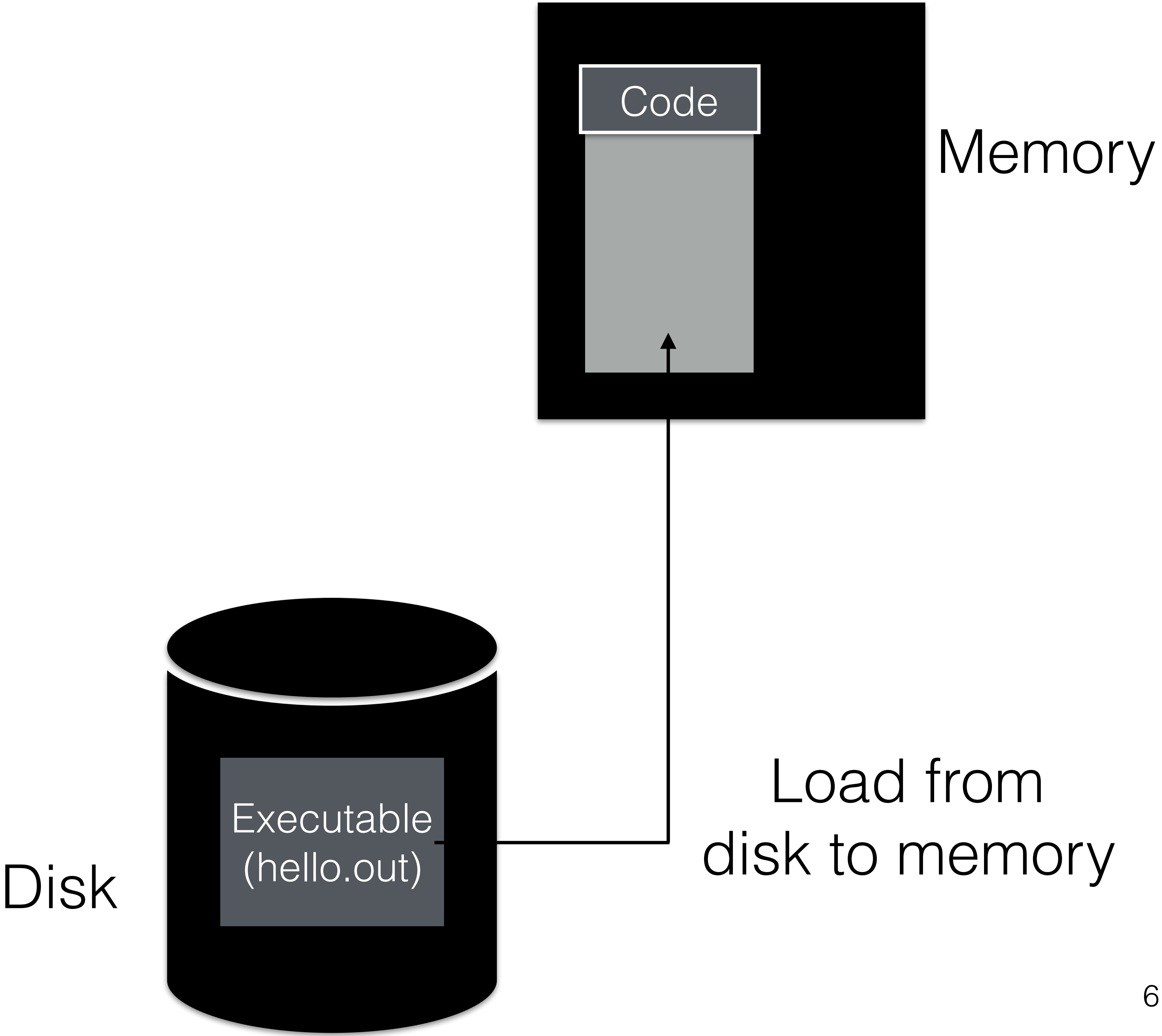
Process Execution



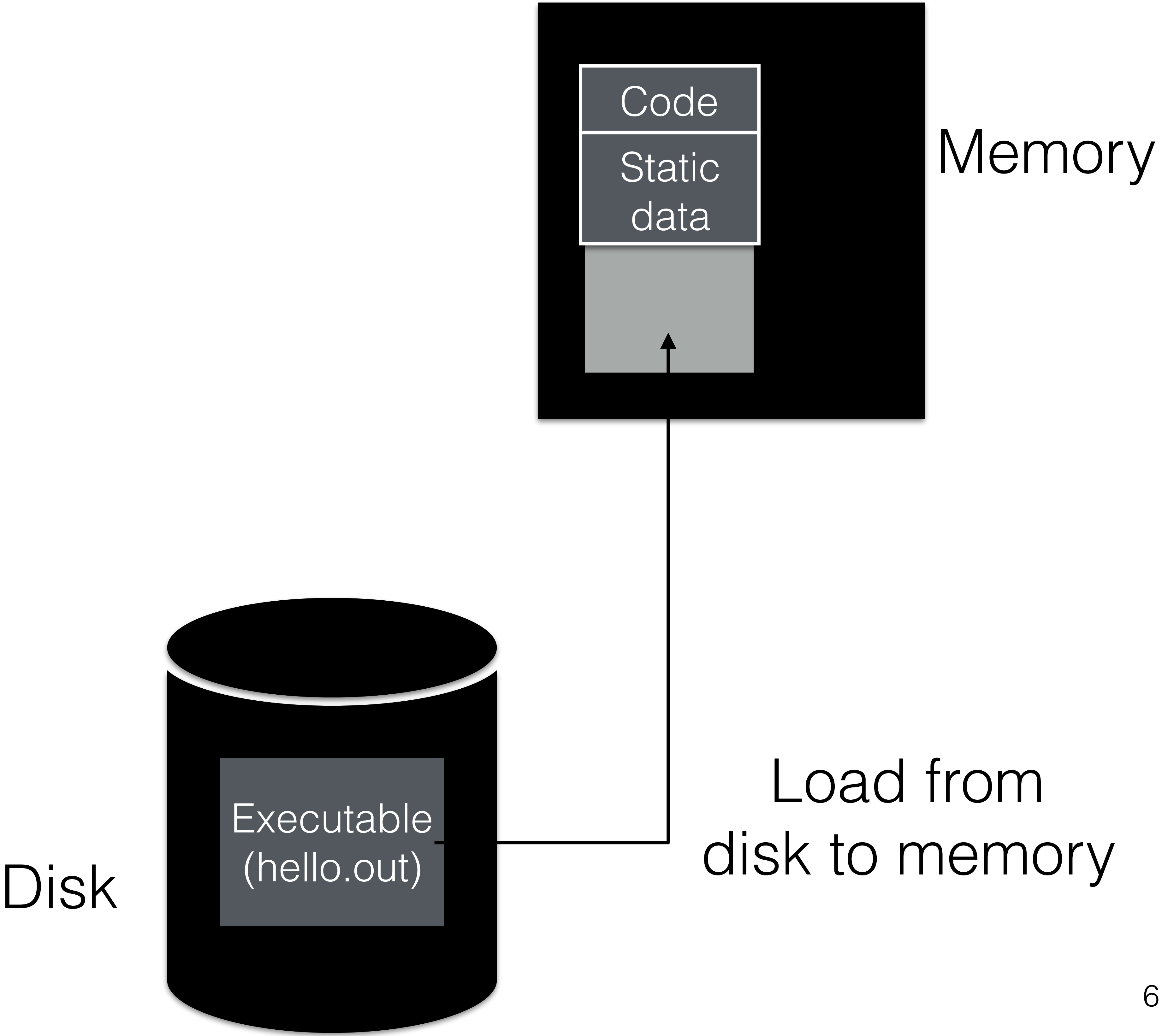
Process Execution



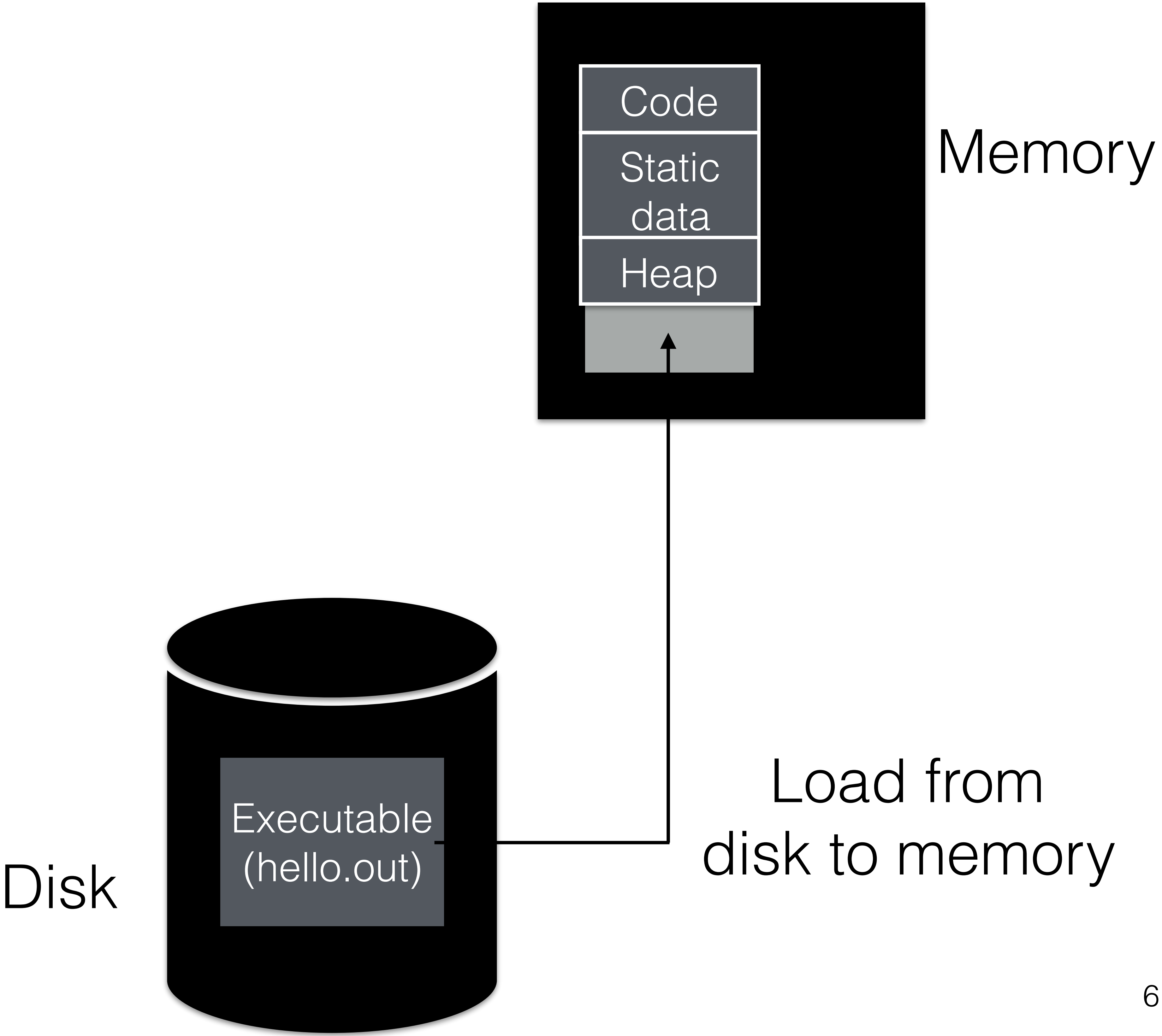
Process Execution



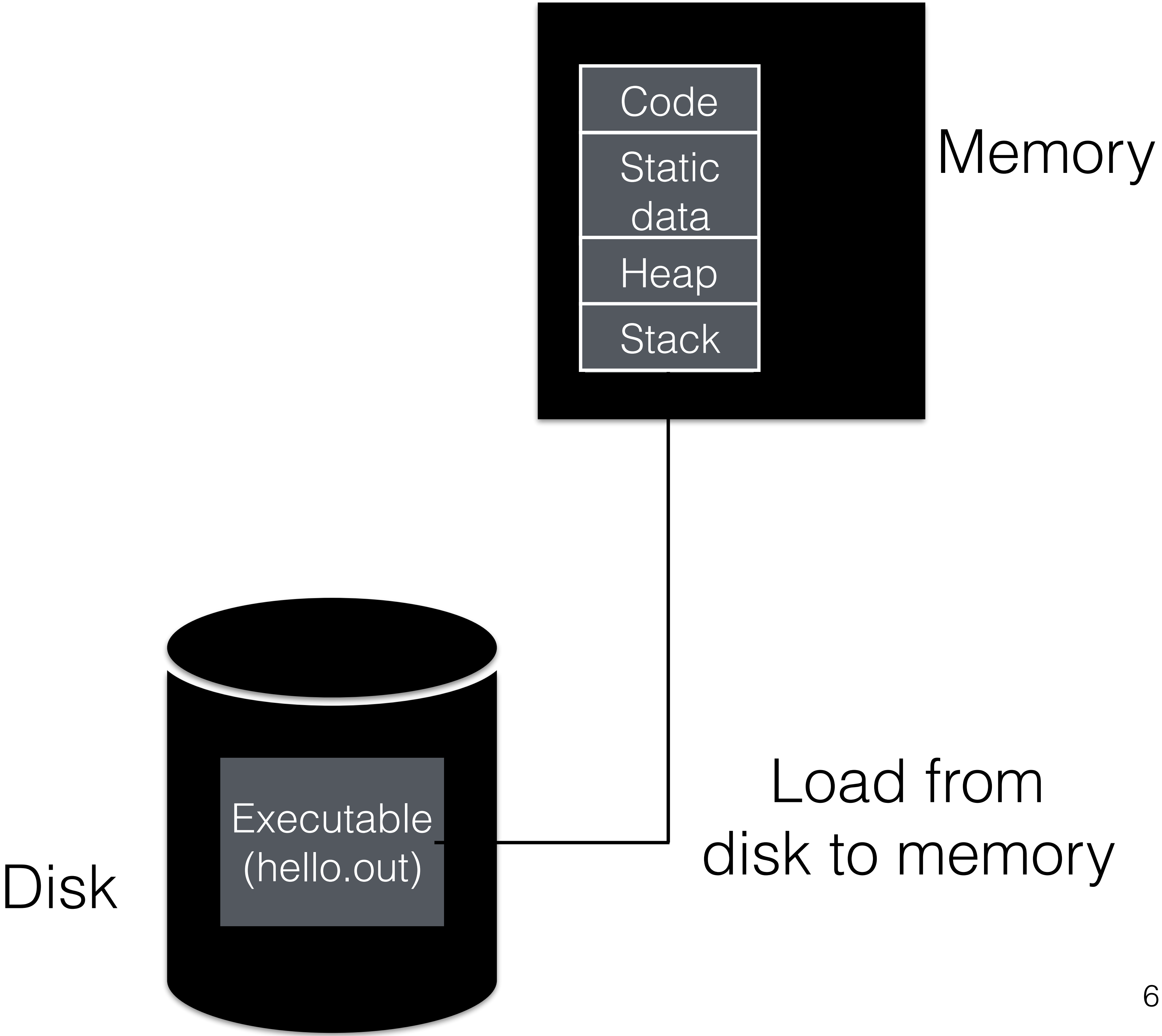
Process Execution



Process Execution

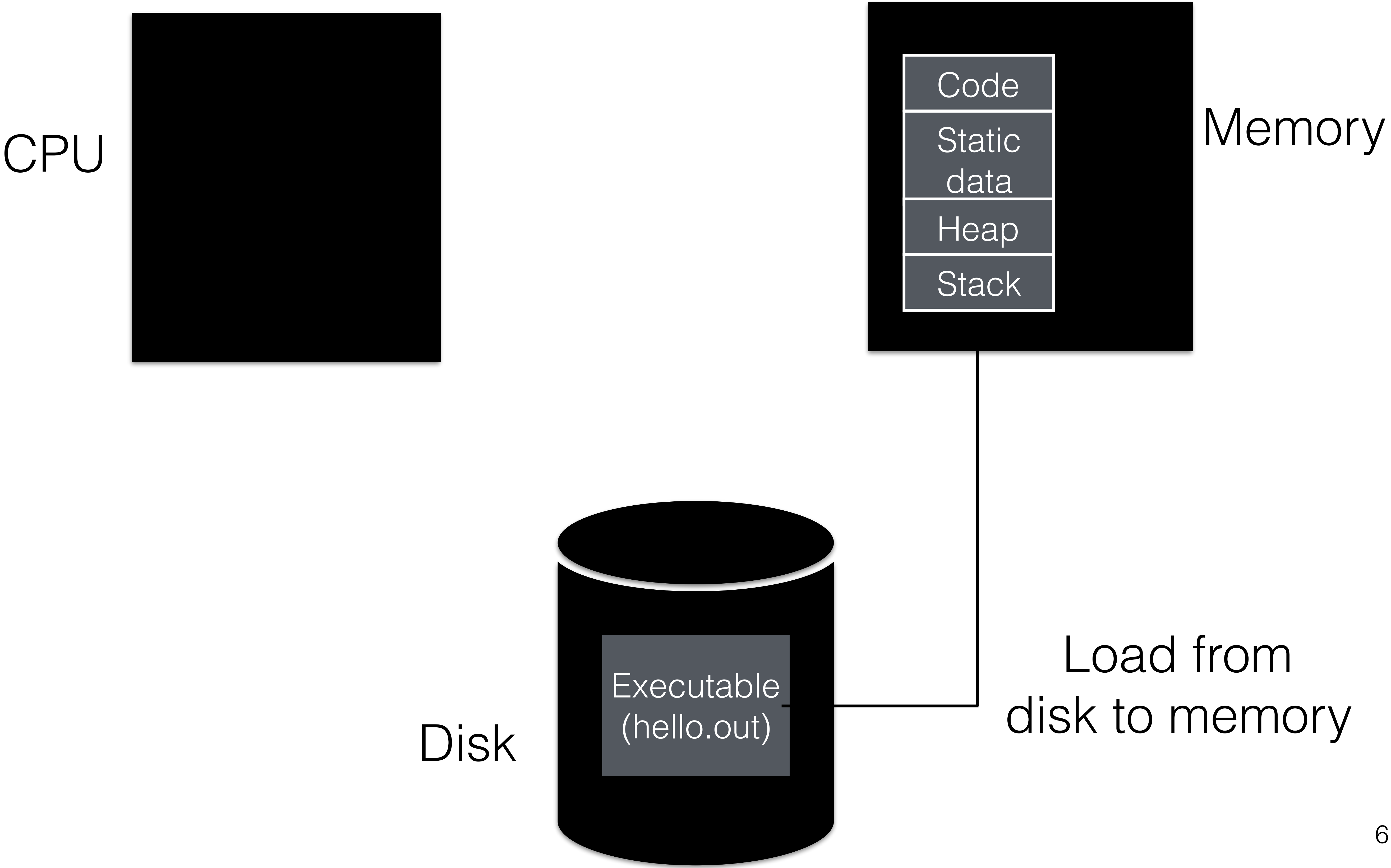


Process Execution



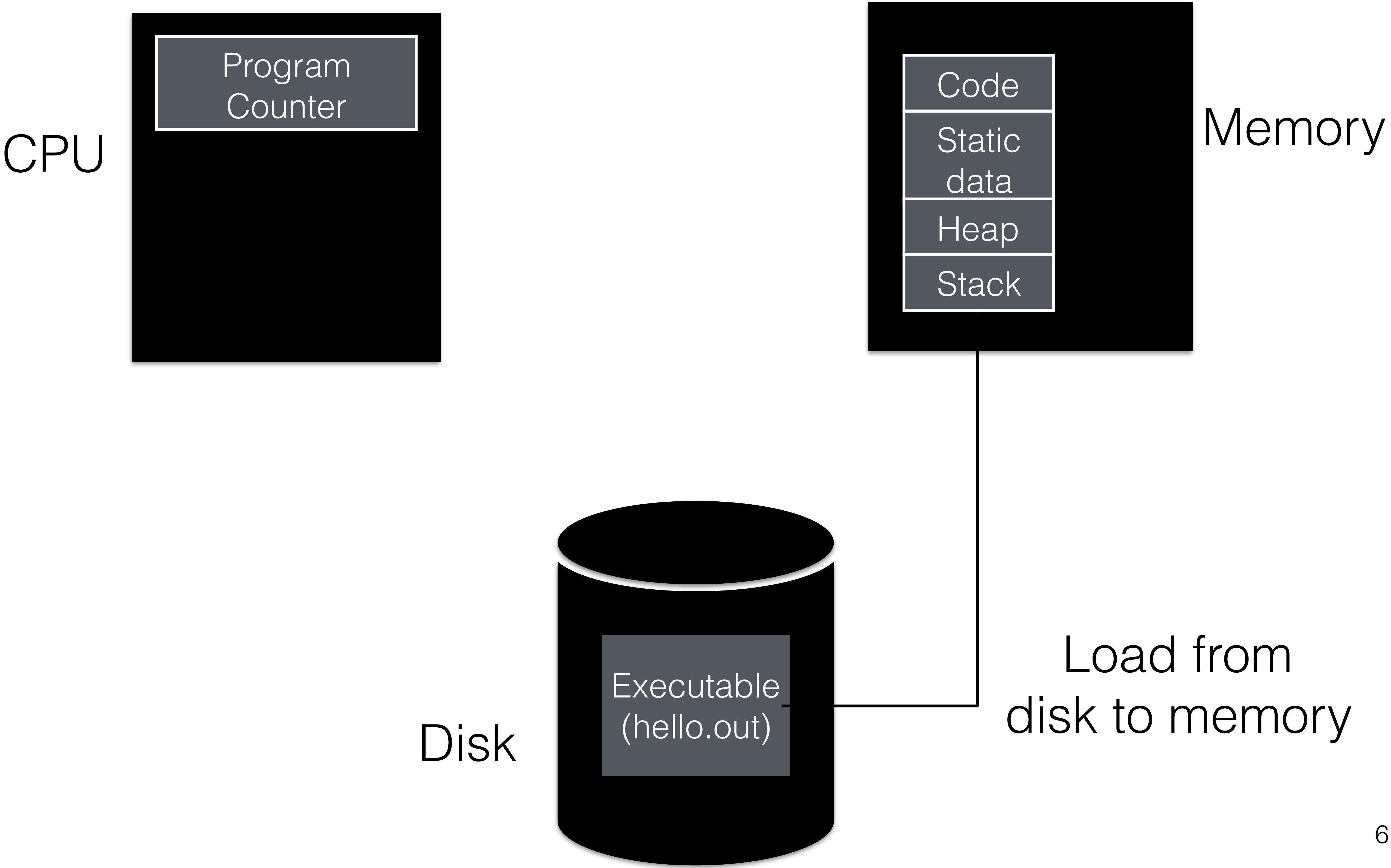
Load from disk to memory

Process Execution



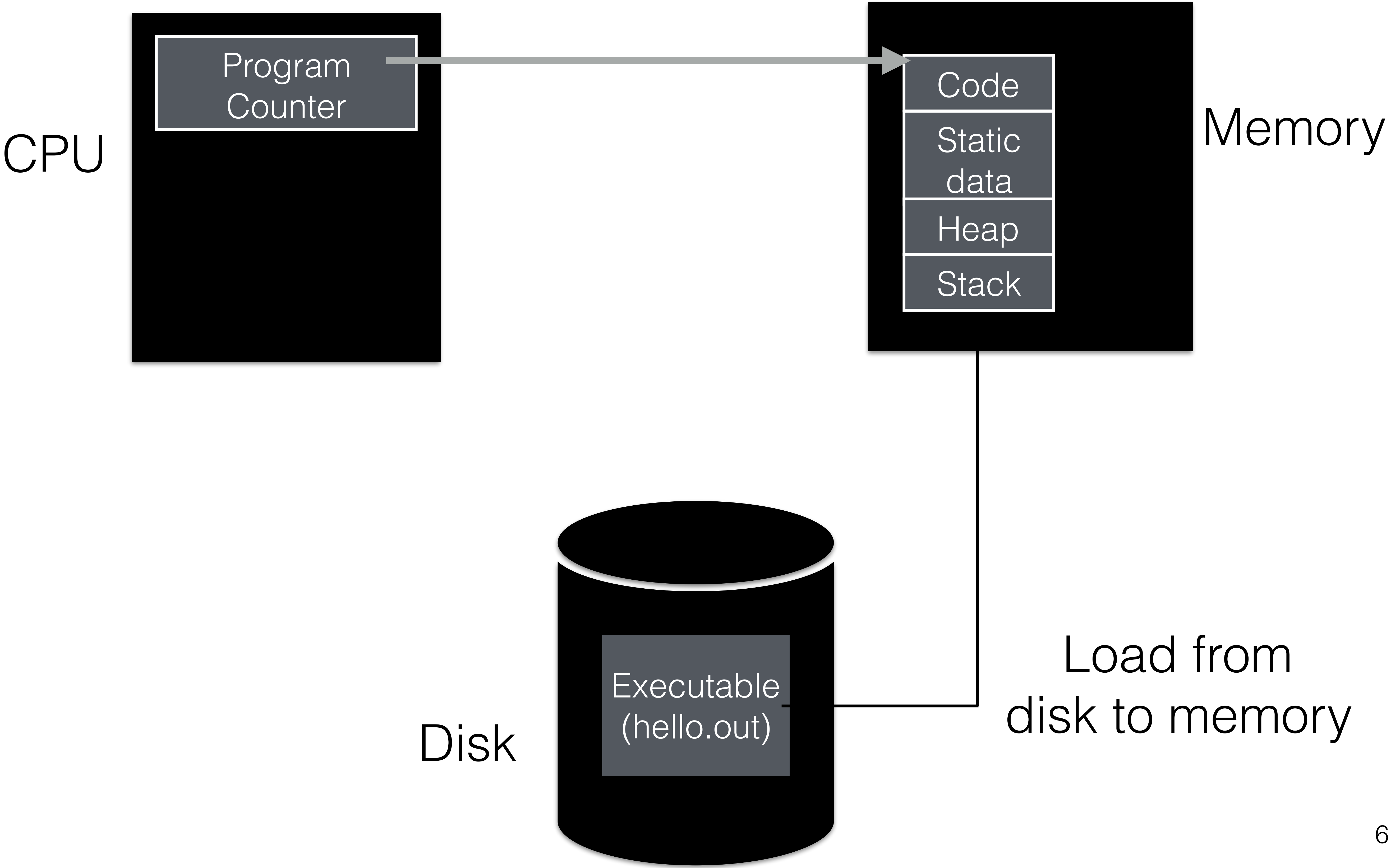
Load from disk to memory

Process Execution



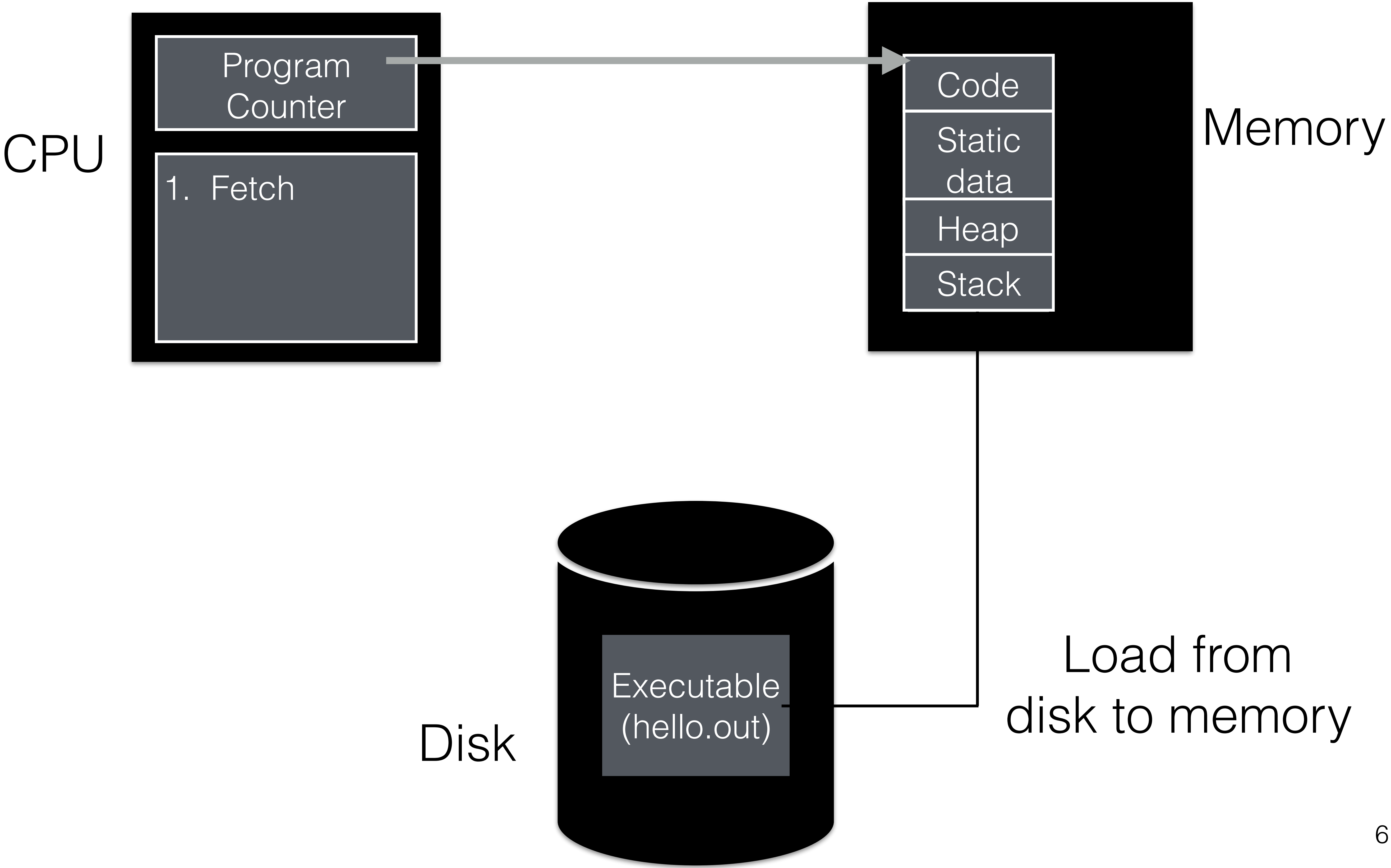
Load from disk to memory

Process Execution



Load from disk to memory

Process Execution



CPU

Program Counter

1. Fetch

Code

Static data

Heap

Stack

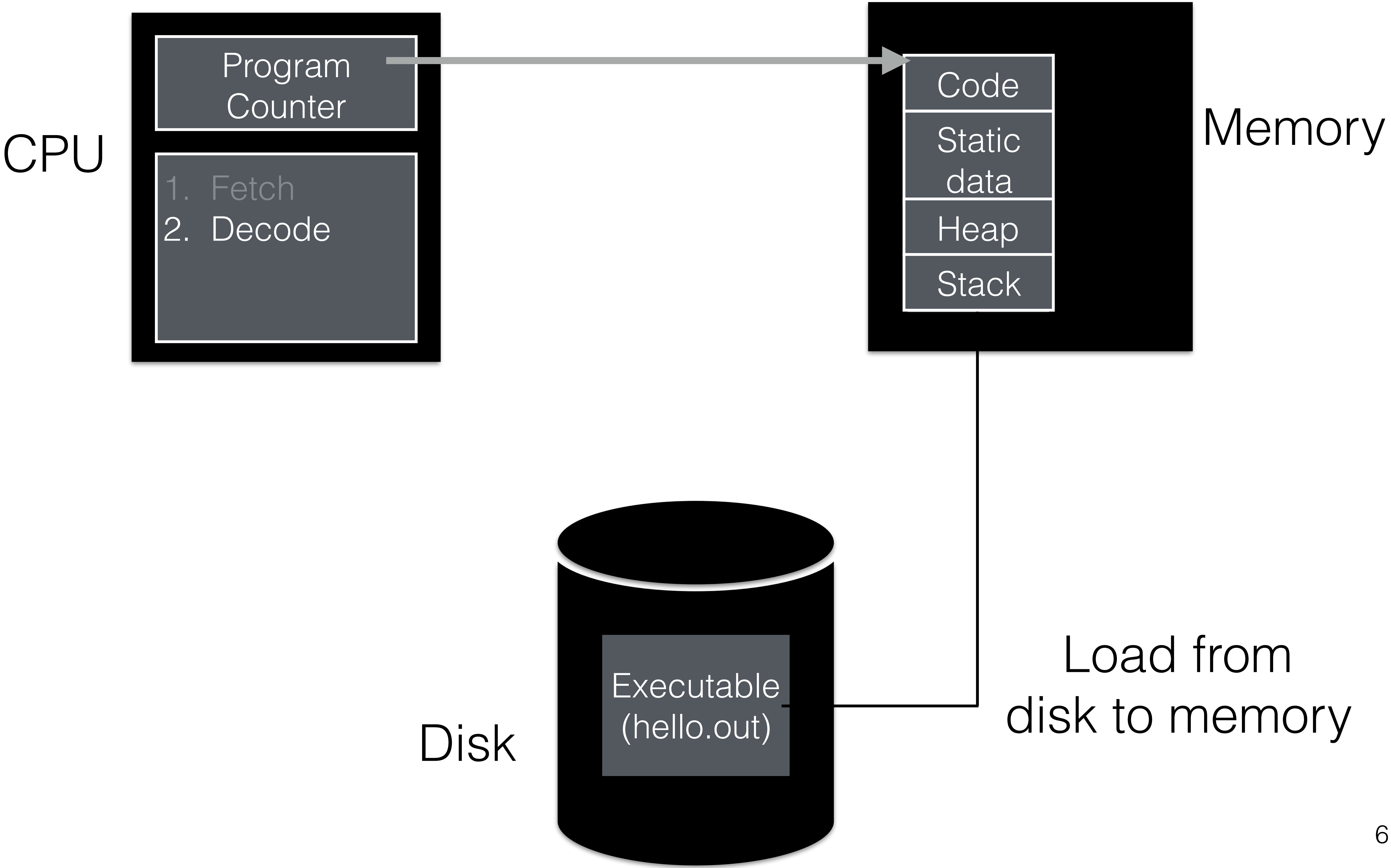
Memory

Disk

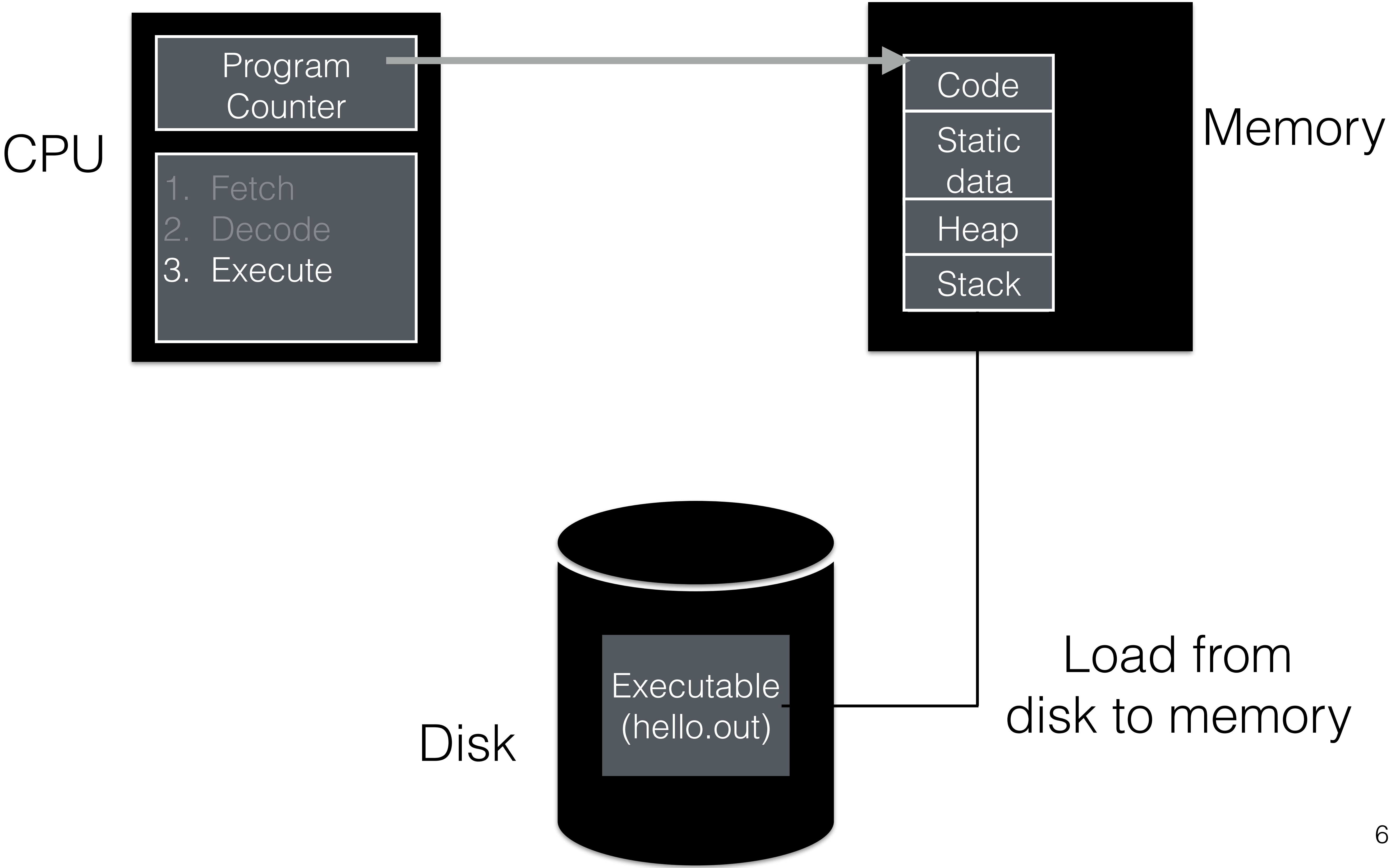
Executable (hello.out)

Load from disk to memory

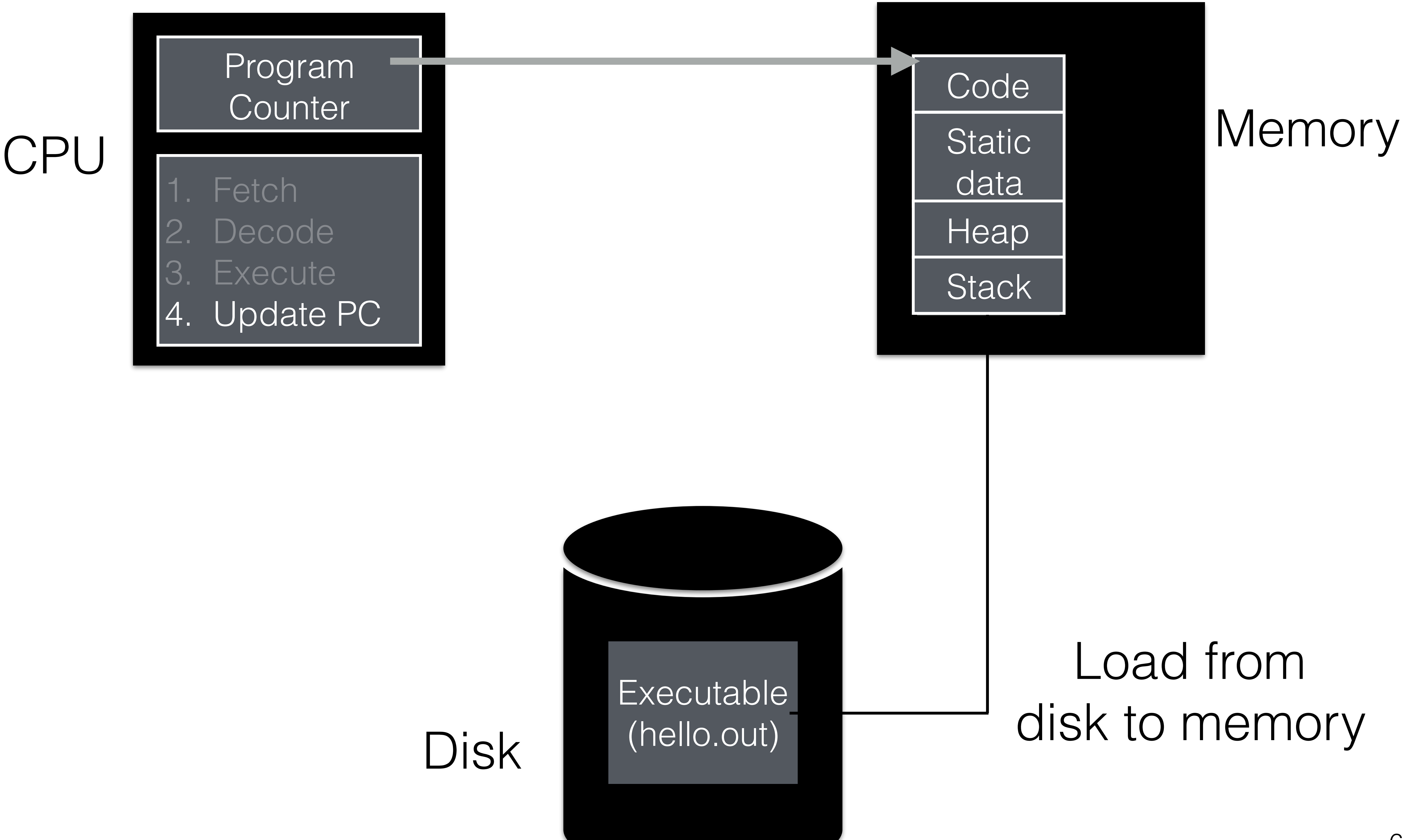
Process Execution



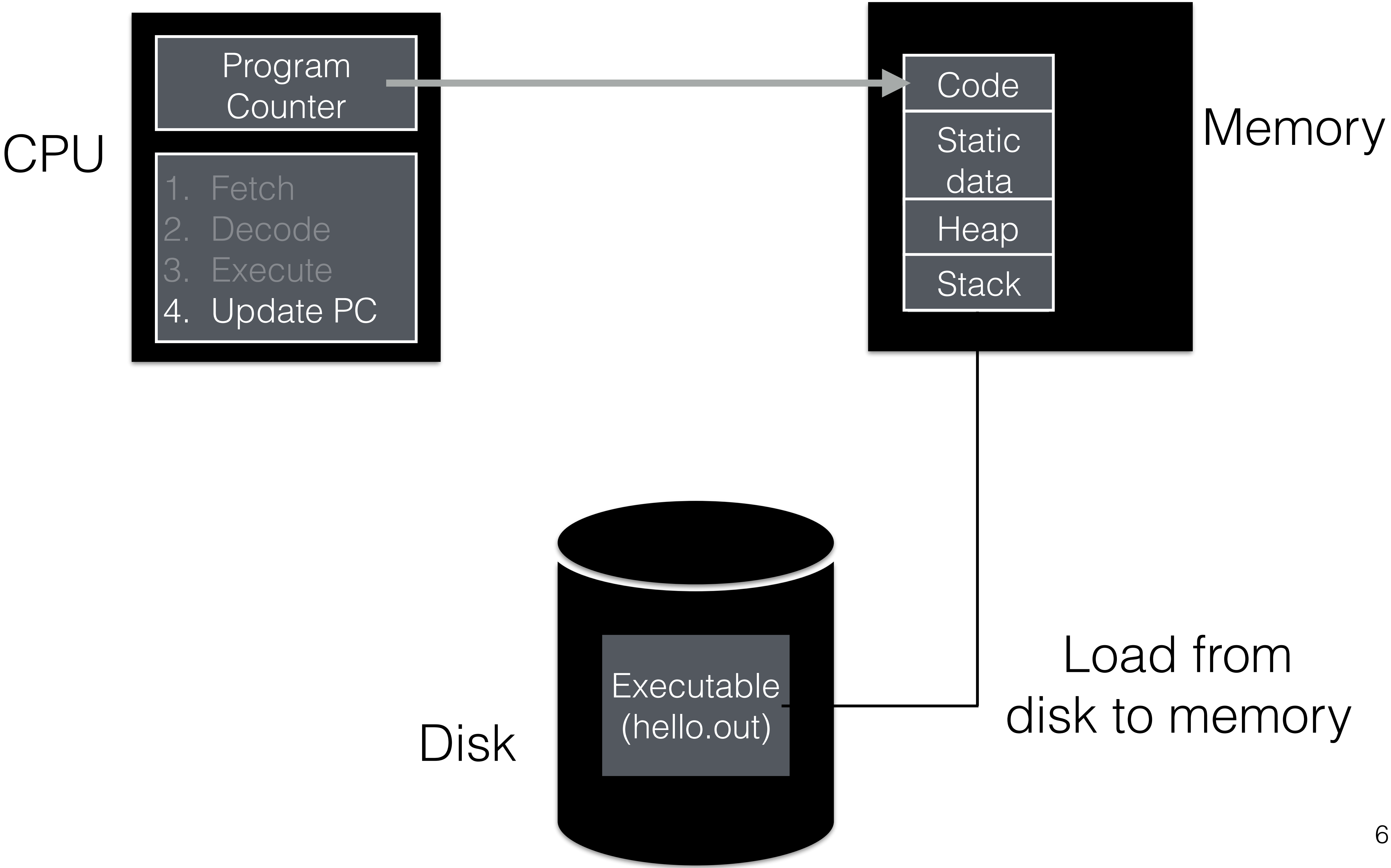
Process Execution



Process Execution



Process Execution



CPU Virtualisation

CPU Virtualisation

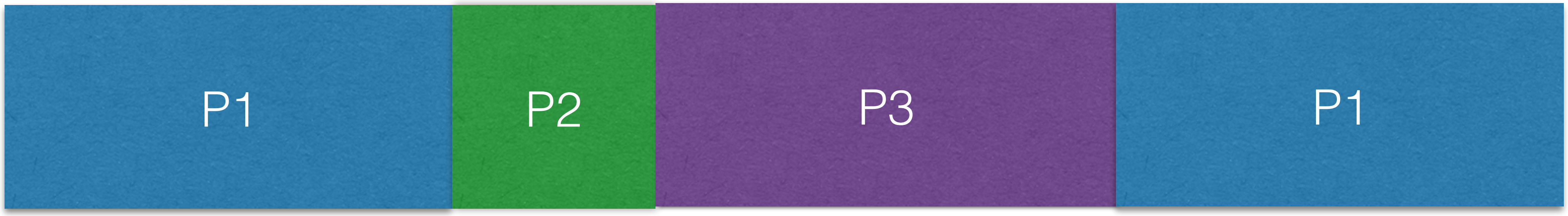
- Goal: Provide an illusion of many CPUs

CPU Virtualisation

- Goal: Provide an illusion of many CPUs
- How: Time sharing between processes

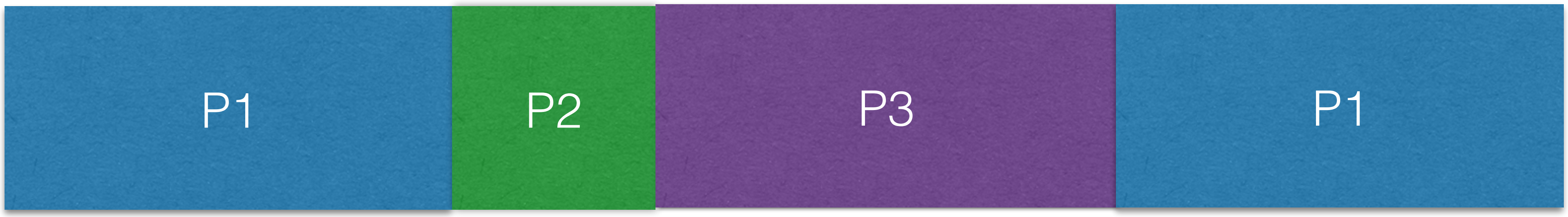
CPU Virtualisation

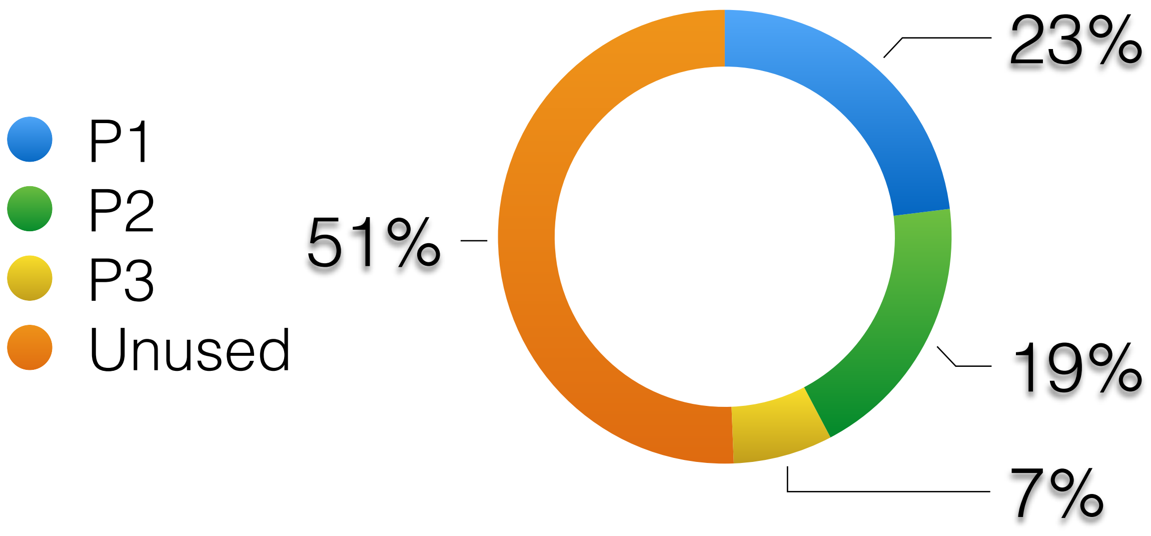
- Goal: Provide an illusion of many CPUs
- How: Time sharing between processes



CPU Virtualisation

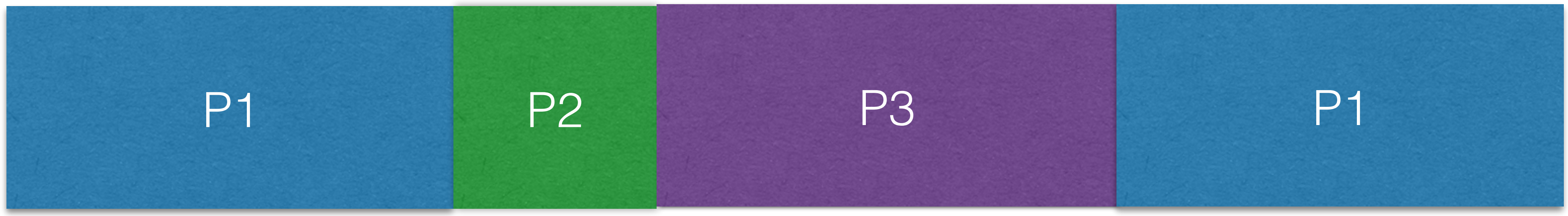
- Goal: Provide an illusion of many CPUs
- How: Time sharing between processes

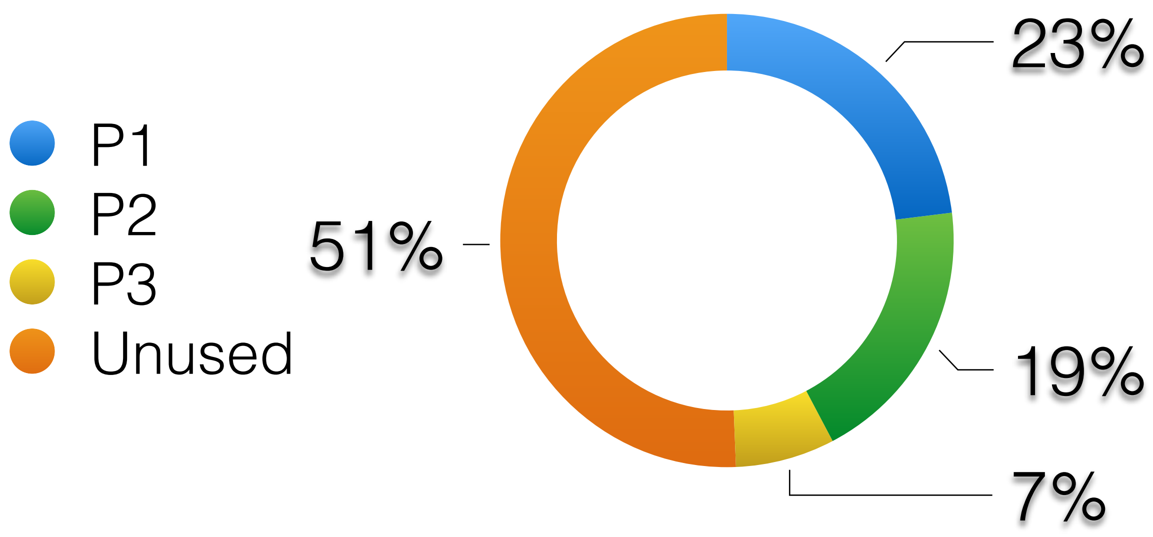




CPU Virtualisation

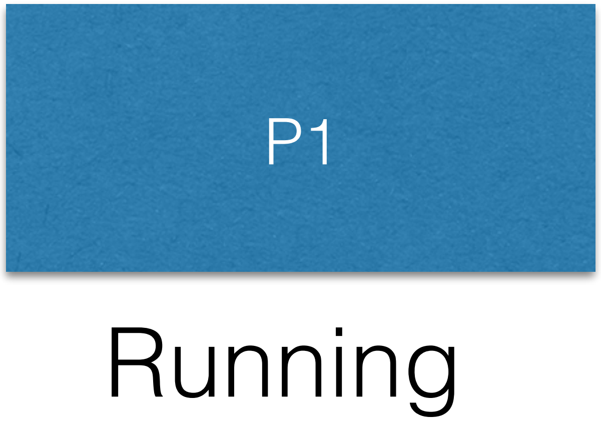
- Goal: Provide an illusion of many CPUs
- How: Time sharing between processes





Space sharing for memory

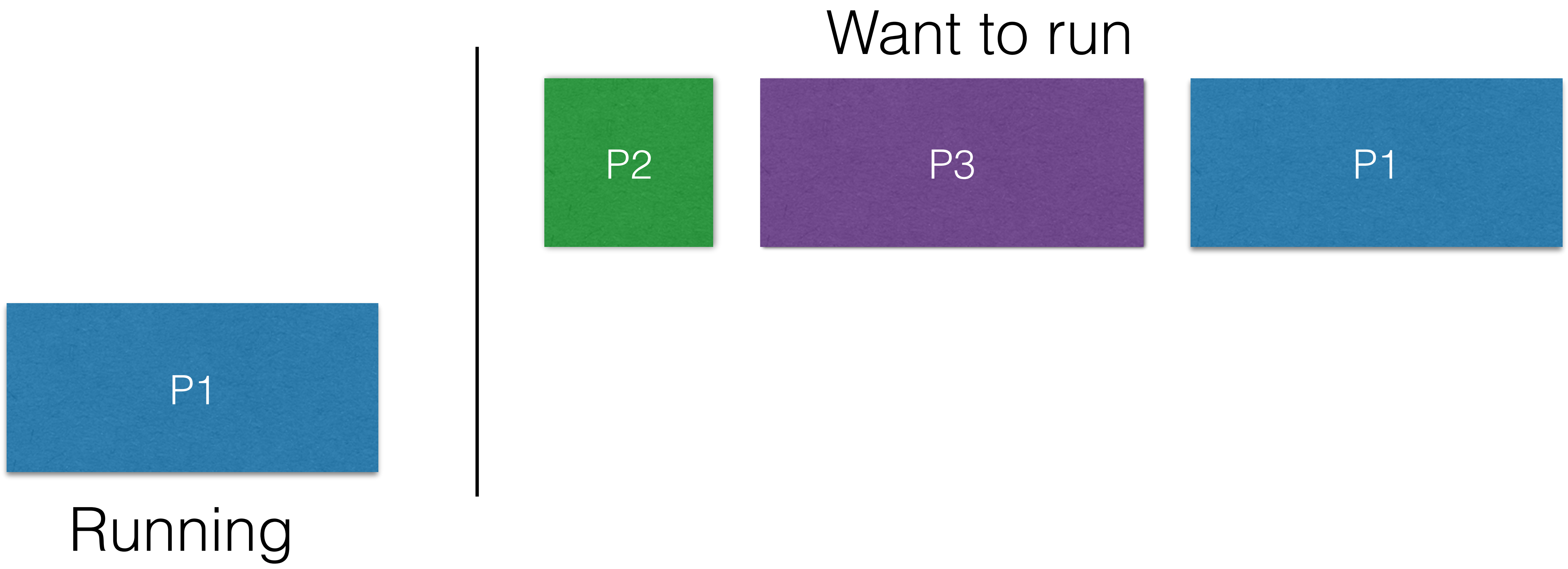
CPU Virtualisation II



P1

Running

CPU Virtualisation II



Want to run

P2

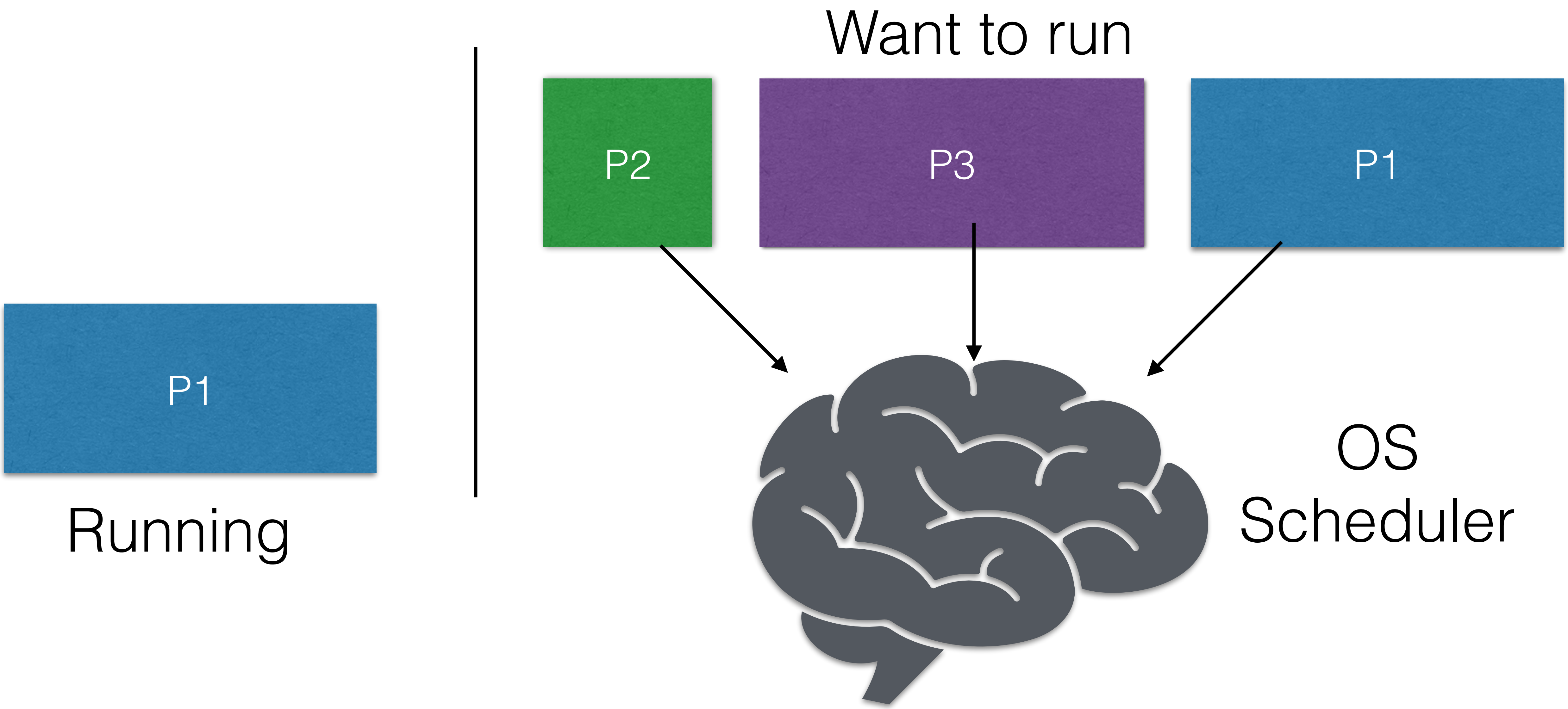
P3

P1

P1

Running

CPU Virtualisation II



P1

Running

Want to run

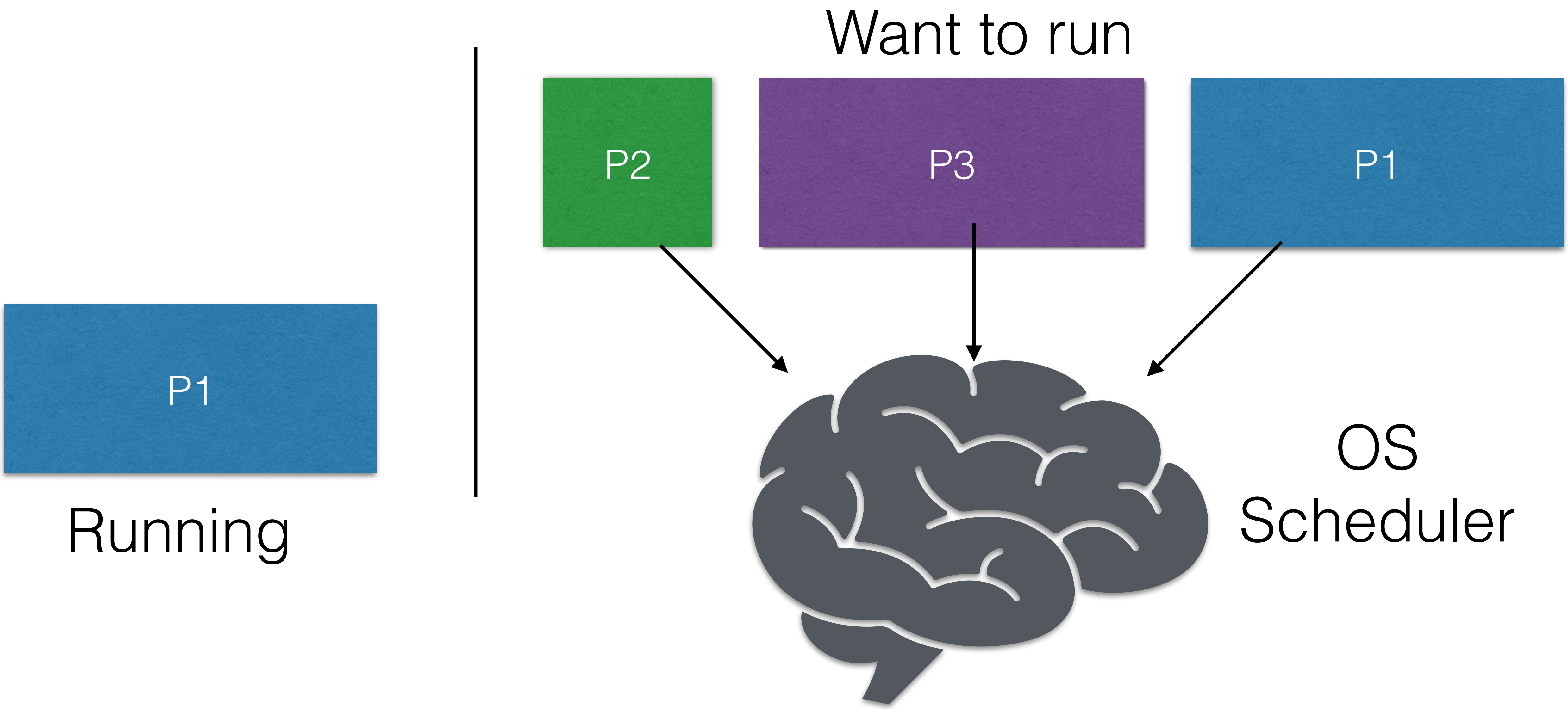
P2

P3

P1

OS Scheduler

CPU Virtualisation II



P1

Running

Want to run

P2

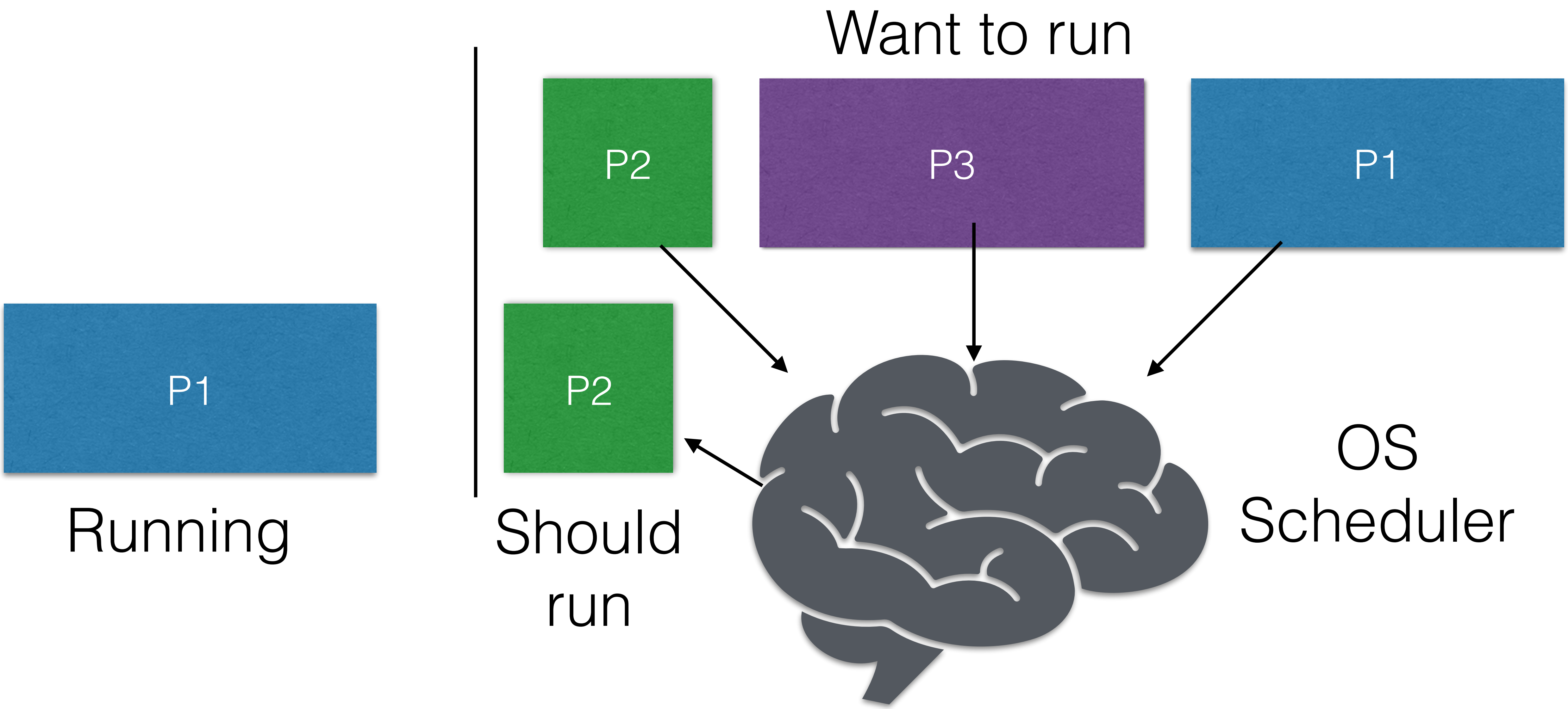
P3

P1

OS Scheduler

Next P = f(run time, metric, type of process, …)

CPU Virtualisation II



Want to run

P2

P3

P1

P1

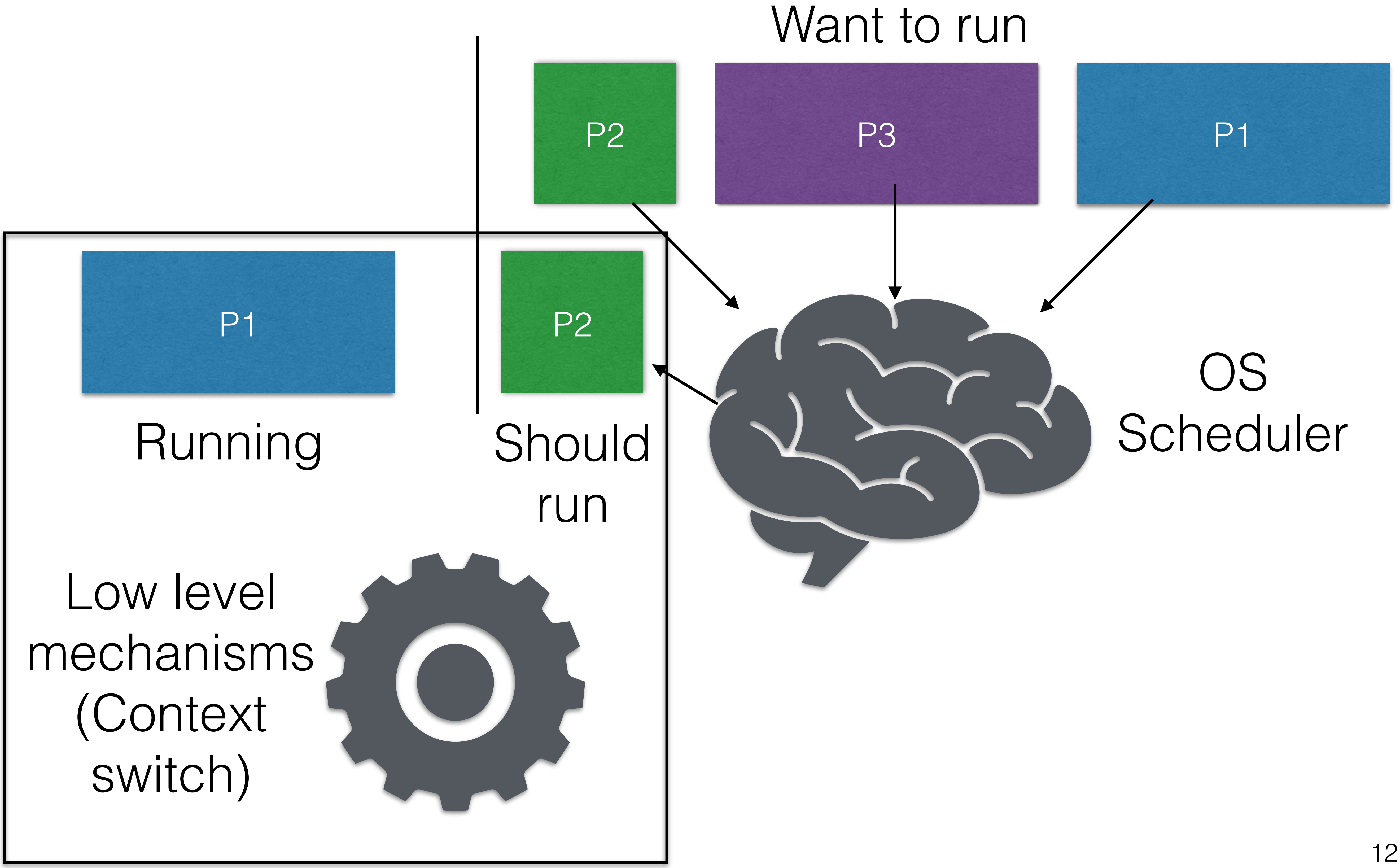
Running

P2

Should run

OS Scheduler

CPU Virtualisation II



Want to run

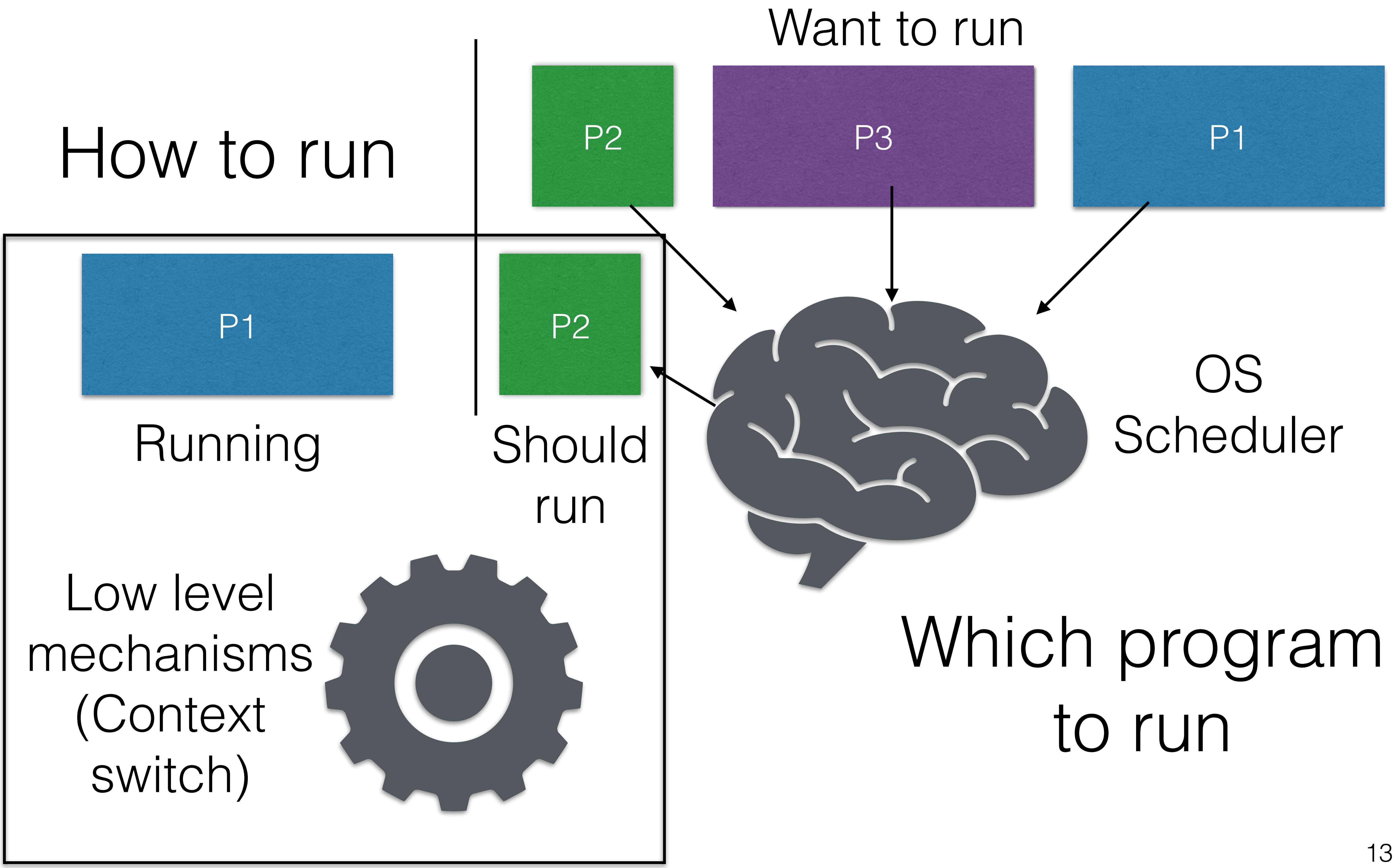
Running

Should run

OS Scheduler

Low level mechanisms (Context switch)

CPU Virtualisation II



Want to run

P2

P3

P1

How to run

P1

Running

P2

Should run

OS Scheduler

Low level mechanisms (Context switch)

Which program to run

Process API

Process API

- Create process:

Process API

- Create process:
  - Double click

Process API

- Create process:
    - Double click
    - Run on command line

Process API

- Create process:
  - Double click
  - Run on command line
- Destroy processes:

Process API

- Create process:
  - Double click
  - Run on command line
- Destroy processes:
  - Task manager

Process API

- Create process:
  - Double click
  - Run on command line
- Destroy processes:
  - Task manager
  - Command line

Process API

- Create process:
  - Double click
  - Run on command line
- Destroy processes:
  - Task manager
  - Command line
- Wait:

Process API

- Create process:
  - Double click
  - Run on command line
- Destroy processes:
  - Task manager
  - Command line
- Wait:
  - Don’t run process till other process completes

Process API

- Create process:
  - Double click
  - Run on command line
- Destroy processes:
  - Task manager
  - Command line
- Wait:
  - Don’t run process till other process completes
- Status:

Process API

- Create process:
  - Double click
  - Run on command line
- Destroy processes:
  - Task manager
  - Command line
- Wait:
  - Don’t run process till other process completes
- Status:
  - How long run, what state it is in

Process API

- Create process:
  - Double click
  - Run on command line
- Destroy processes:
  - Task manager
  - Command line
- Wait:
  - Don’t run process till other process completes
- Status:
  - How long run, what state it is in
  - Does **top, ps** give us this info?

Process API

- Create process:
  - Double click
  - Run on command line
- Destroy processes:
  - Task manager
  - Command line
- Wait:
  - Don’t run process till other process completes
- Status:
  - How long run, what state it is in
  - Does **top, ps** give us this info?
- Misc.:

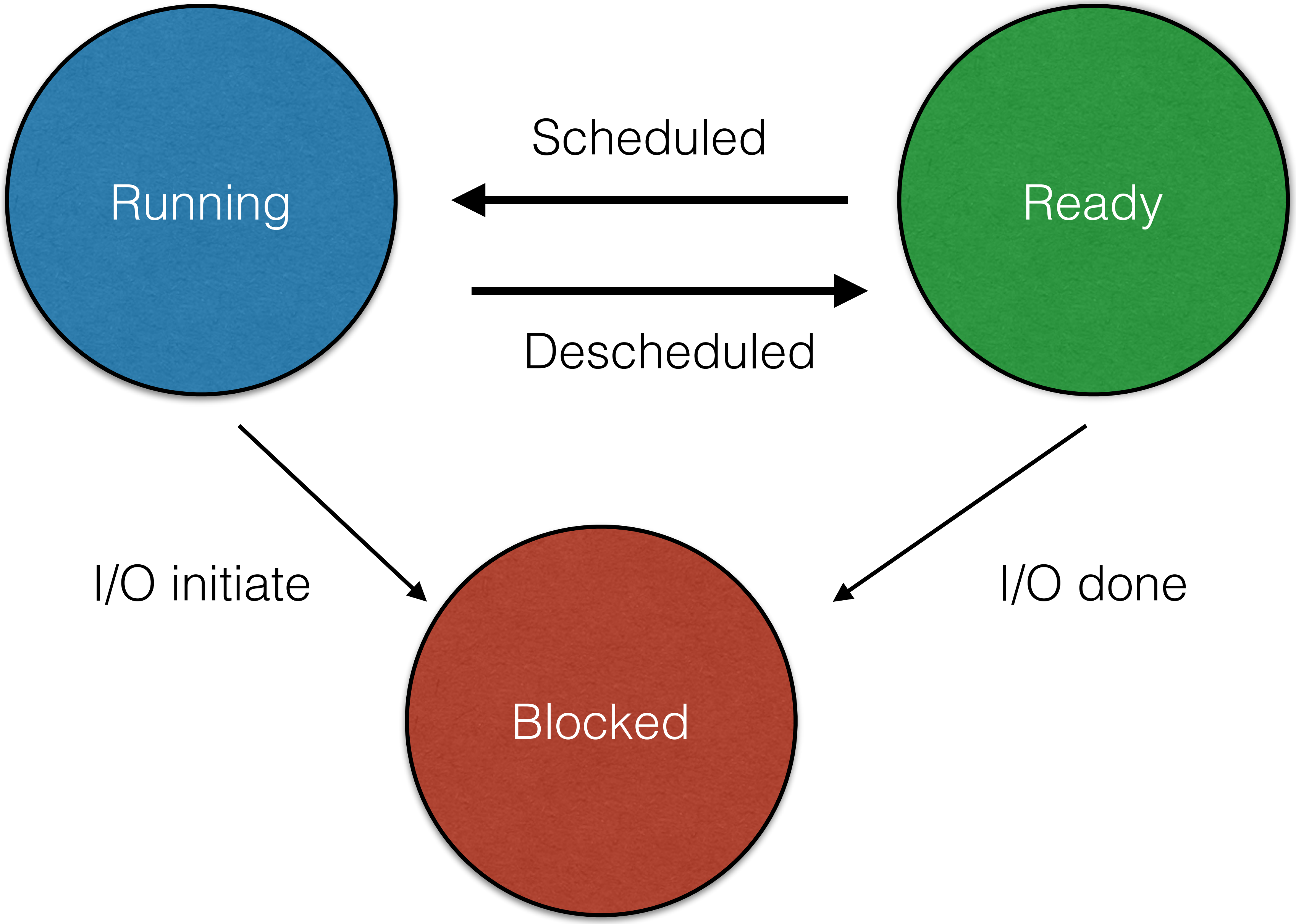
Process API

- Create process:
  - Double click
  - Run on command line
- Destroy processes:
  - Task manager
  - Command line
- Wait:
  - Don’t run process till other process completes
- Status:
  - How long run, what state it is in
  - Does **top, ps** give us this info?
- Misc.:
  - Suspend

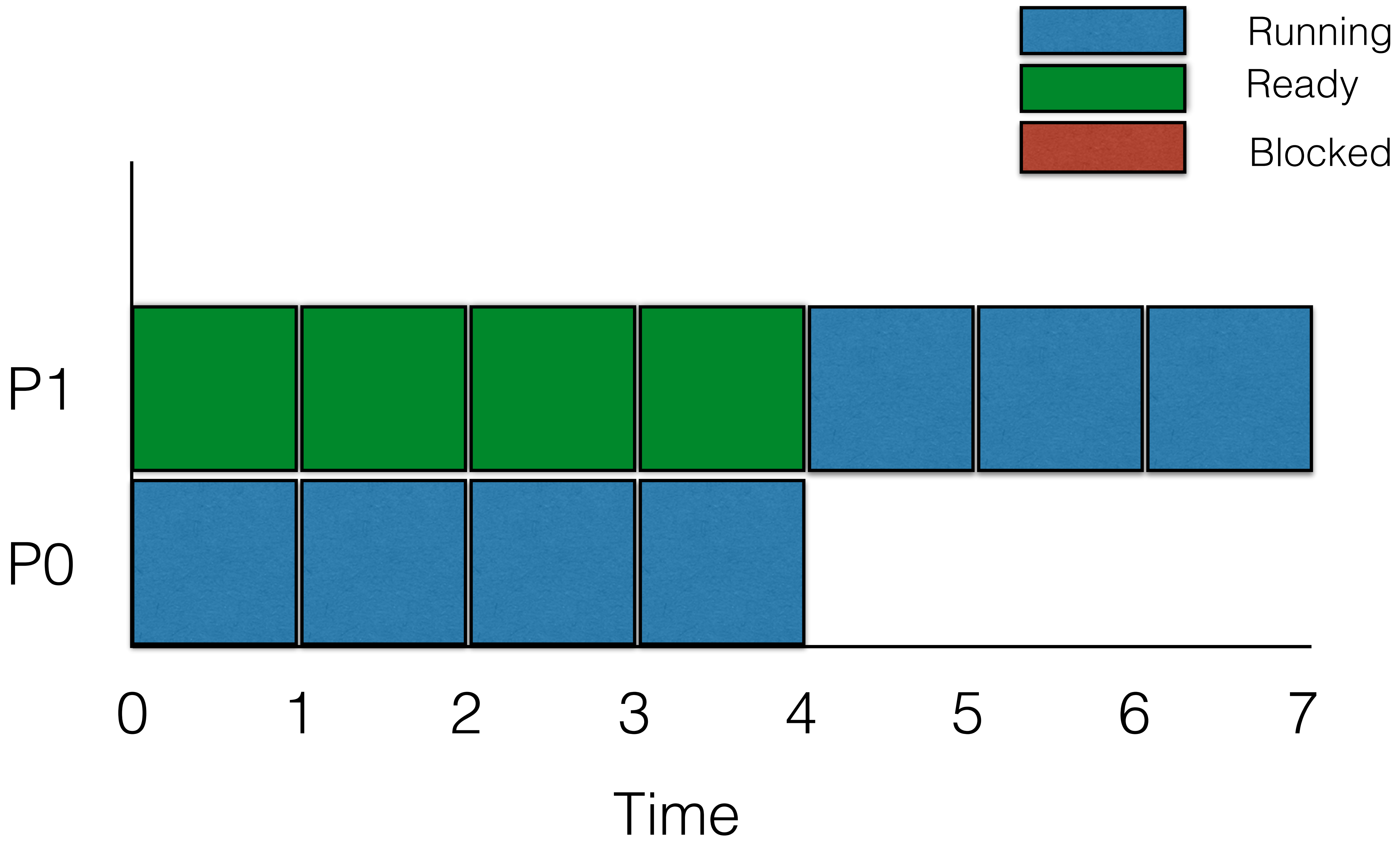
Process API

- Create process:
  - Double click
  - Run on command line
- Destroy processes:
  - Task manager
  - Command line
- Wait:
  - Don’t run process till other process completes
- Status:
  - How long run, what state it is in
  - Does **top, ps** give us this info?
- Misc.:
  - Suspend

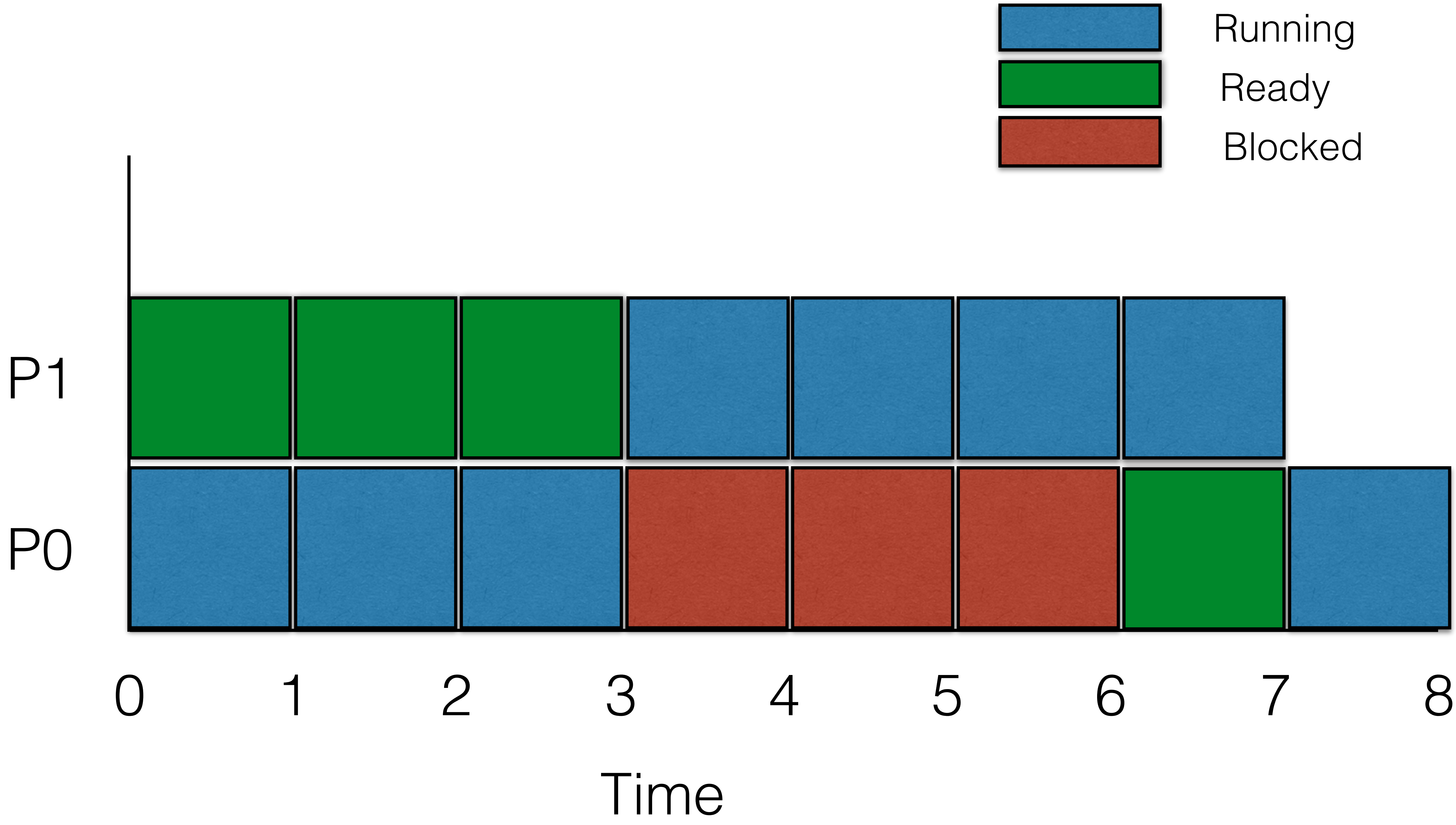
Process States



Process States



Process States


Running
Ready
Blocked
P1
P0
Time