Processes

Andre M. Maier, DHBW Ravensburg
dhbw@andre-maier.com

What is a process?

- A process is an instance of a program that is being executed
- A process may have one or more child processes and be a child of its parent process.
- Instructions within one process may be executed concurrently by multiple threads.

Andre M. Maier
Dept. of Computer Science

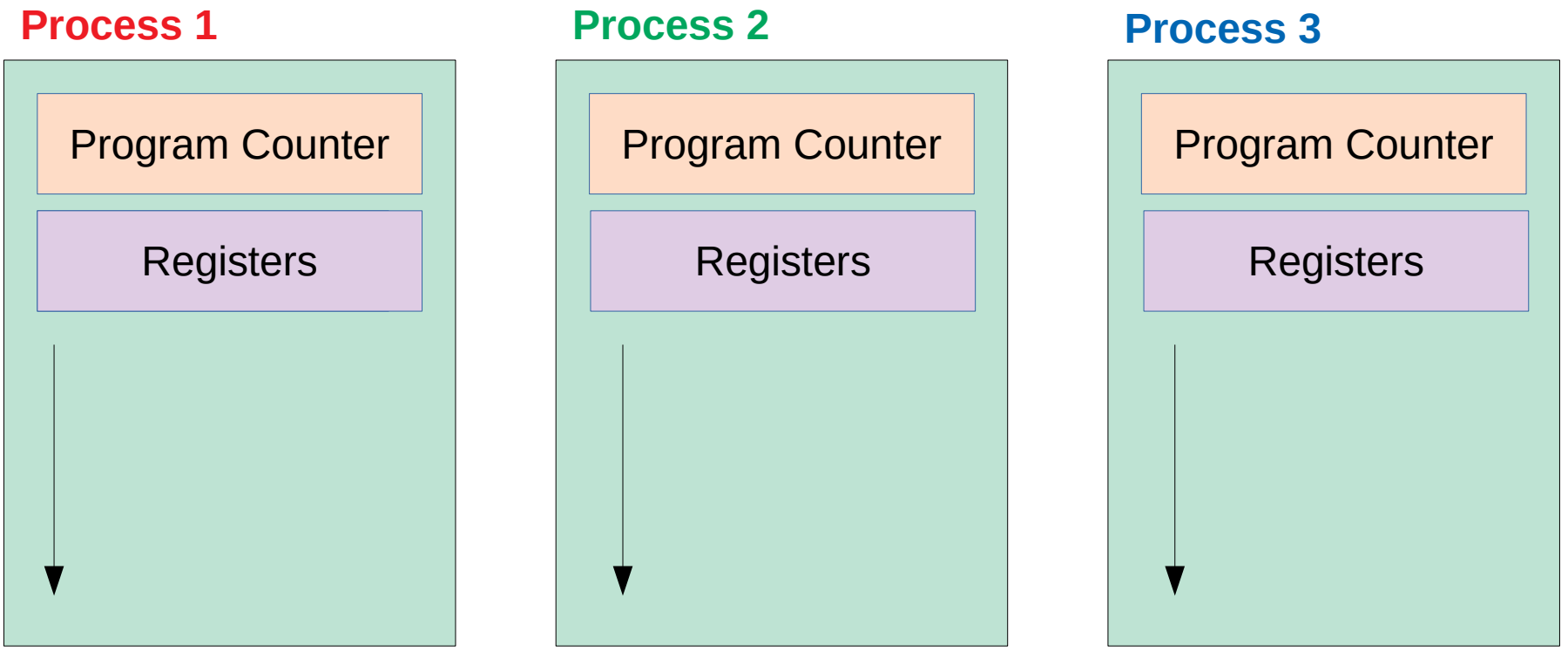
Program vs. Process

- Baking analogy
  - CPU → Baker
  - Input Data → Cake Ingredients
  - Program → Recipe
- “... a program is something that may be stored on disk, not doing anything.” [1]

[1] Andrew S. Tanenbaum, Modern Operating Systems, 4th Edition

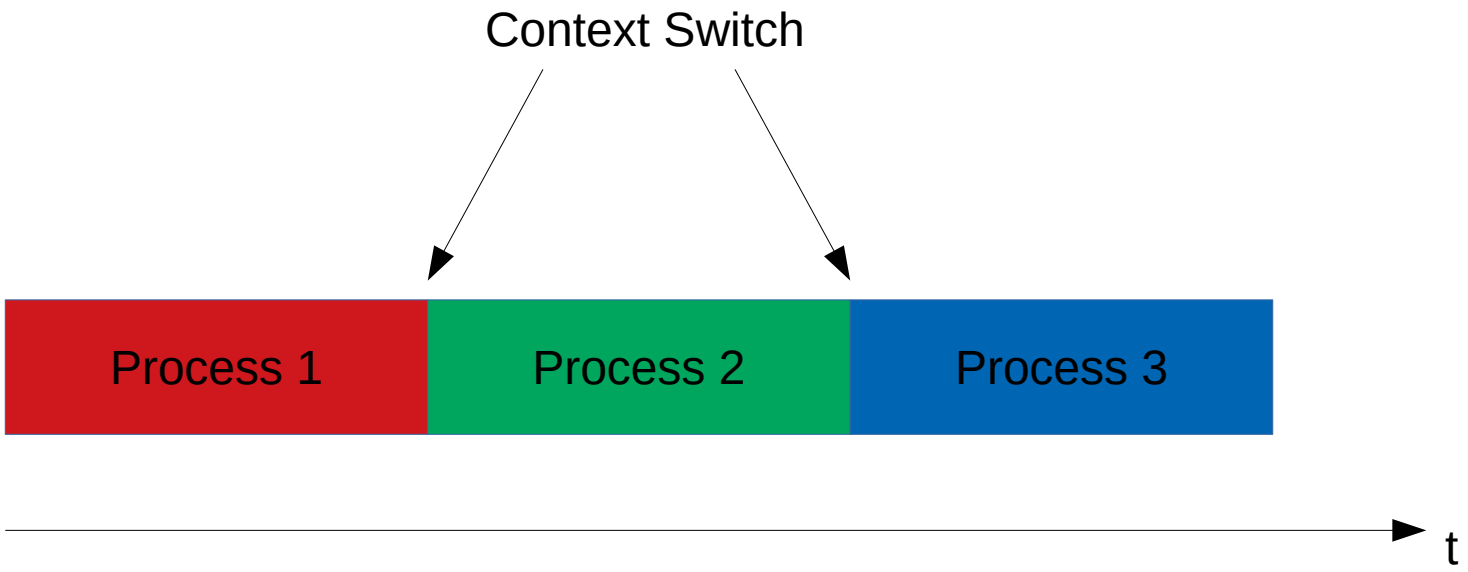
Andre M. Maier
Dept. of Computer Science

Process Model



Andre M. Maier
Dept. of Computer Science

Execution on a Single-Core CPU



Andre M. Maier
Dept. of Computer Science

Context Switch (1)

- A context completely describes a process's current state of execution.
  - Program counter
  - Registers
  - Virtual address space
- During a context switch ...
  - the current process's execution is suspended
  - the context information for that process is stored
  - the context of the next process is retrieved from memory and restored in the CPU's registers
  - the execution of the new process is resumed at the location indicated by the program counter

Andre M. Maier
Dept. of Computer Science

Context Switch (2)

- Hardware context switching
  - Context switch completely performed by CPU hardware
- Software context switching
  - Context switch performed by the operating system
  - Often preferred for performance and portability reasons
- Context switches are performed by a component called *dispatcher*, which is part of the scheduler. The dispatcher also alters the flow of execution that is required in a context switch.

Andre M. Maier
Dept. of Computer Science

Processes vs. Threads

- A single process can have multiple threads.
- A thread represents a separate, independent, path of execution within a process.
- All threads of a process share the process's address space (memory).
- Each thread has its own execution stack
  - Multiple threads can call the execute the same function concurrently
- Advantages of threads over processes
  - less context switching time
  - easier, more efficient, communication between threads
  - concurrency within a process

Andre M. Maier
Dept. of Computer Science

Process-related OS Objectives

- Maximize processor utilization
- Allocate resources to processes
- Prioritize execution
- Allow for user creation of processes
- Provide mechanisms for synchronization and inter-process communication

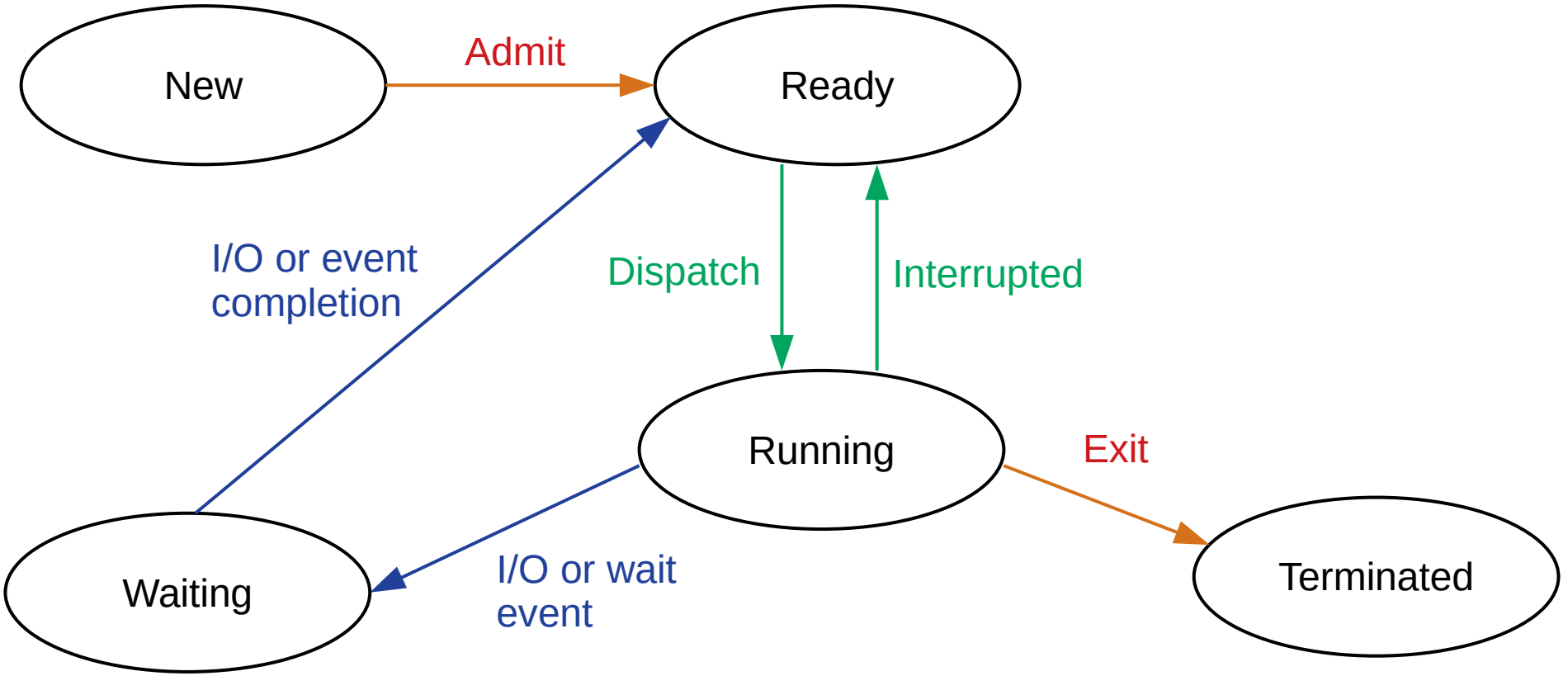
Andre M. Maier
Dept. of Computer Science

Process Control Block (PCB)

Process ID
State
Priority
Owner
Memory Pointers
Context data
I/O status information
Accounting information
...
...

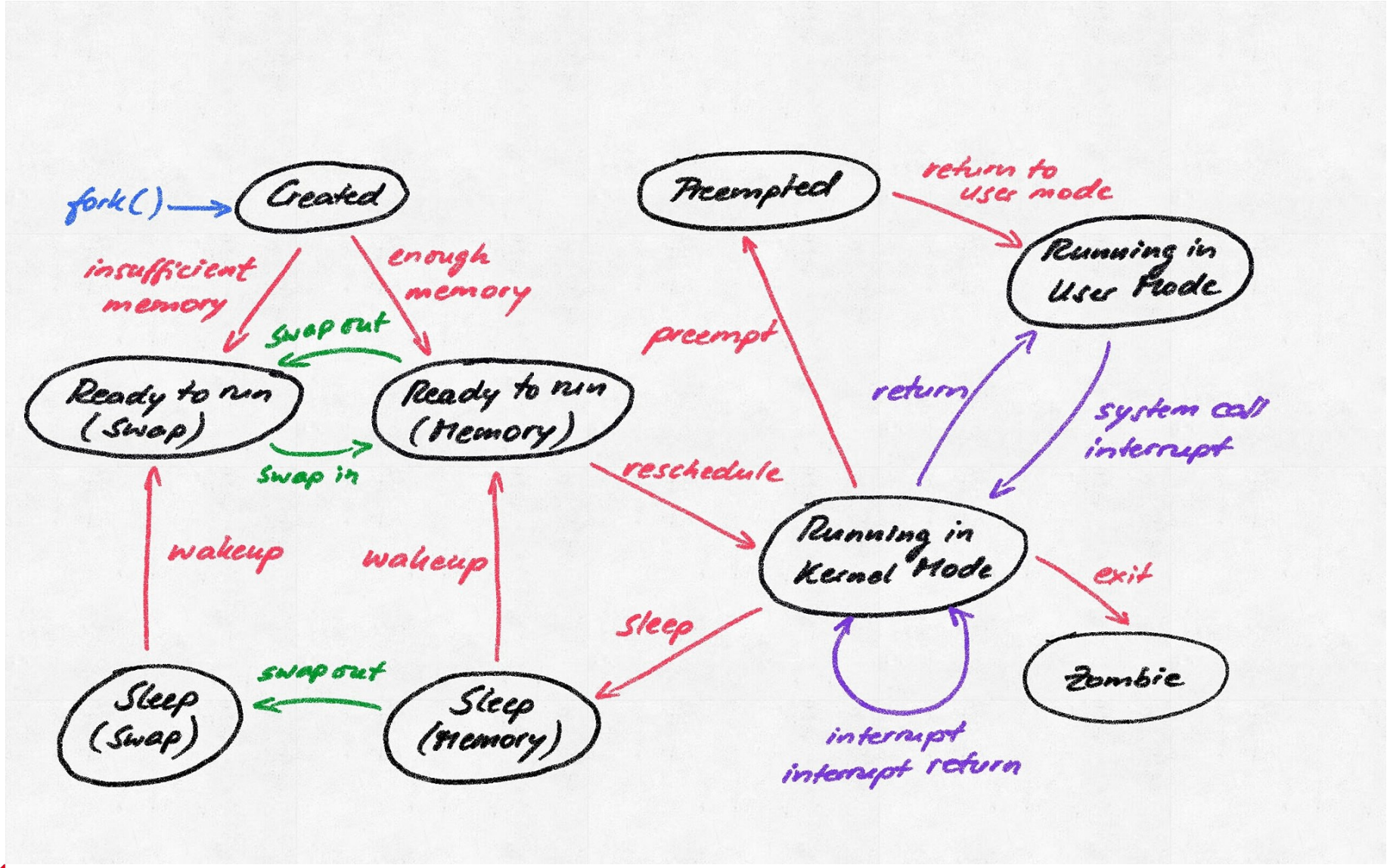
Andre M. Maier
Dept. of Computer Science

Process States



Andre M. Maier
Dept. of Computer Science

On Linux ...



Andre M. Maier
Dept. of Computer Science

Lab Exercises (1)

- Make yourself familiar with the following Unix/Linux tools.
  - `ps`
  - `pstree`
  - `top`, `htop`
  - GUI system resource monitor
- Find out how you can use `ps` to identify and display
  - the PID of a process with a certain name
  - the process state of a process with a certain PID or name
- Write and run a C program that contains an empty infinite loop. Examine its execution by using the tools mentioned above.

Andre M. Maier
Dept. of Computer Science

Lab Exercises (2)

- Change the program to put the process to sleep for 1 second in the endless loop. Explain in detail how this change affects process execution.
- Make yourself familiar with the following bash features:
  - `jobs`
  - `bg`, `fg`
  - `&`
- What do *bash jobs* have to do with processes?

Andre M. Maier
Dept. of Computer Science

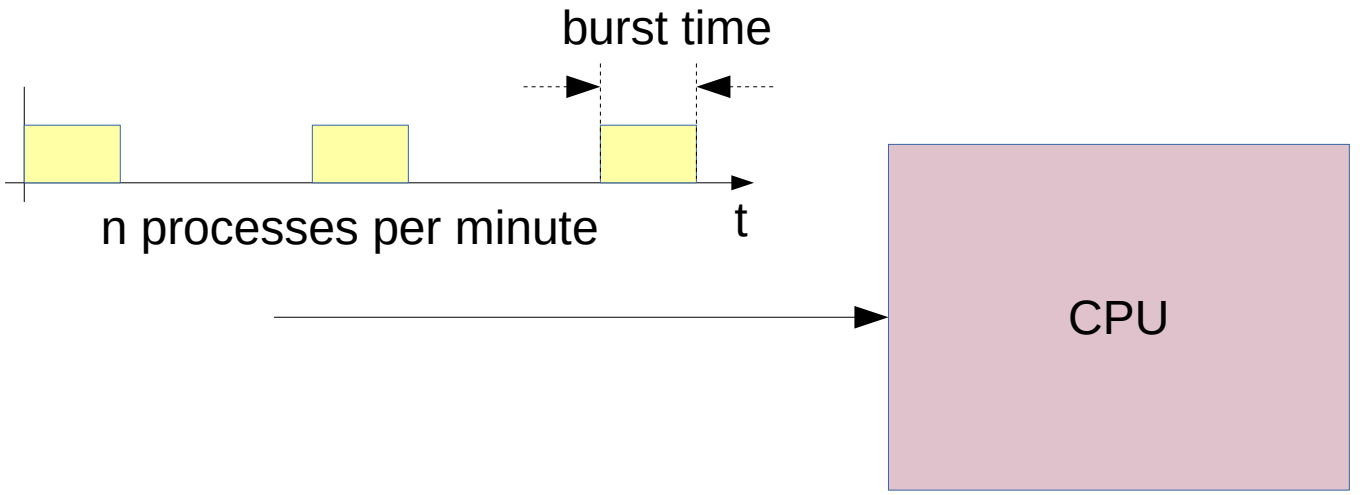
Lab Exercises (3)

- Analyze the following C program and explain what it does.

```
#include <unistd.h>
int main(int argc, char** argv)
{
   while(1)
      fork();
   return 0;
}
```

- Open your GUI system resource monitor, compile and run the program.
- What security mechanism does Linux provide to prevent such code from causing harm?

CPU Utilization (1)



$$CPU\ Utilization = \frac{Burst\ Time}{Period\ Time}$$

Example:
Processes arrive at a rate of 12 processes per minute. The average burst time is 4 seconds. Determine the average CPU utilization (in percentage).

CPU Utilization (2)

- In Linux, *top* or *uptime* shows the “load average” on a system.
- Three numbers: System load average over the last 1min, 5min, 15min.
- The number is the number of processes in either a *runnable* or *uninterruptable* (e.g. waiting for some I/O) state.
- Interpretation
  - 100% utilization on a single core CPU -> 1.0
  - 100% utilization on a quad core CPU -> 4.0

Questions for Review

- What is the difference between a process and a program?
- What are the steps that are involved in a context switch?
- Explain the five-state process model.
- Explain how a fork bomb works and how it can potentially compromise a computer system.
- The following load average values are shown on a Linux system running on a quad core processor:
  7.12, 2.02, 0.83
  What conclusions can you draw on the workload of this system?