

بسم الله الرحمن الرحيم

# «سیستم عامل»

۱

جلسه ۱: مقدمه

# معرفی سیستم عامل

# What is an operating system?

3

# What is an operating system?

3

- Operating system --“a program that controls the execution of application programs and implements an interface between the user of a computer and the computer hardware”
  - Narrow view of a computer and OS
    - Traditional computer with applications running on it (e.g. PCs, Workstations, Servers)
  - Broad view of a computer and OS
    - Anything that needs to manage resources (e.g. router OS, embedded system, cell phone OS ...)

# Two key OS functions

4

# Two key OS functions

4

- **Abstract Machine**
  - Hides complex details of the underlying hardware
  - Provides common API to applications and services
  - Simplifies application writing

# Two key OS functions

4

- **Abstract Machine**
  - Hides complex details of the underlying hardware
  - Provides common API to applications and services
  - Simplifies application writing
- **Resource Manager**
  - Controls accesses to shared resources
    - CPU, memory, disks, network, ...
  - Allows for global policies to be implemented

# Why is abstraction important?

5



# Why is abstraction important?

5

- **Without OSs and abstract interfaces, application writers must program all device access directly**
  - load device command codes into device registers
  - handle initialization, recalibration, sensing, timing etc for physical devices
  - understand physical characteristics and layout
  - control motors
  - interpret return codes ... etc

# Why is abstraction important?

5

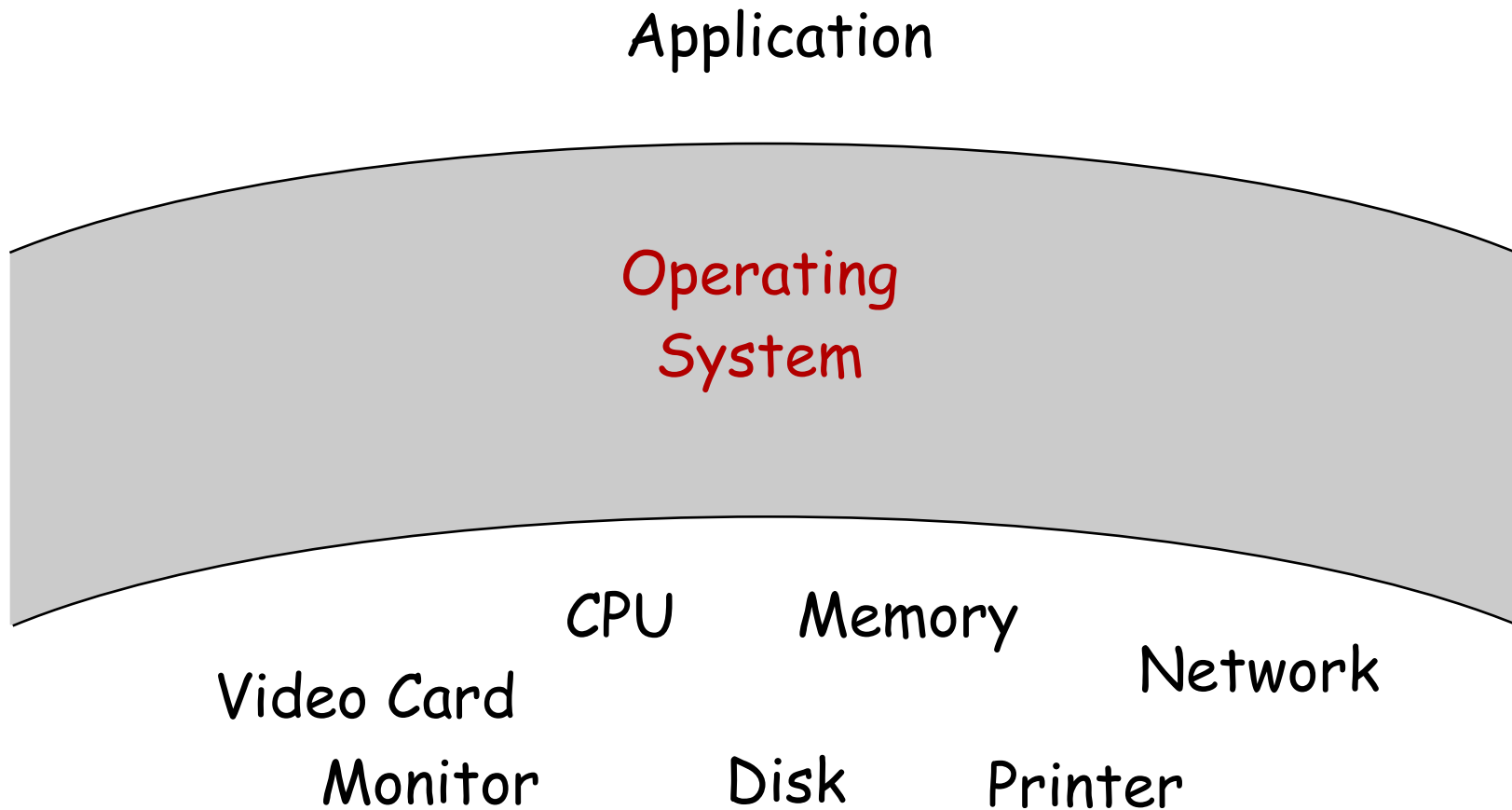
- **Without OSs and abstract interfaces, application writers must program all device access directly**
  - load device command codes into device registers
  - handle initialization, recalibration, sensing, timing etc for physical devices
  - understand physical characteristics and layout
  - control motors
  - interpret return codes ... etc
- **Applications suffer severe code bloat!**
  - very complicated maintenance and upgrading
  - no portability
  - writing this code once, and sharing it, is how OS began!

Application

CPU      Memory      Network  
Video Card      Disk      Printer  
Monitor

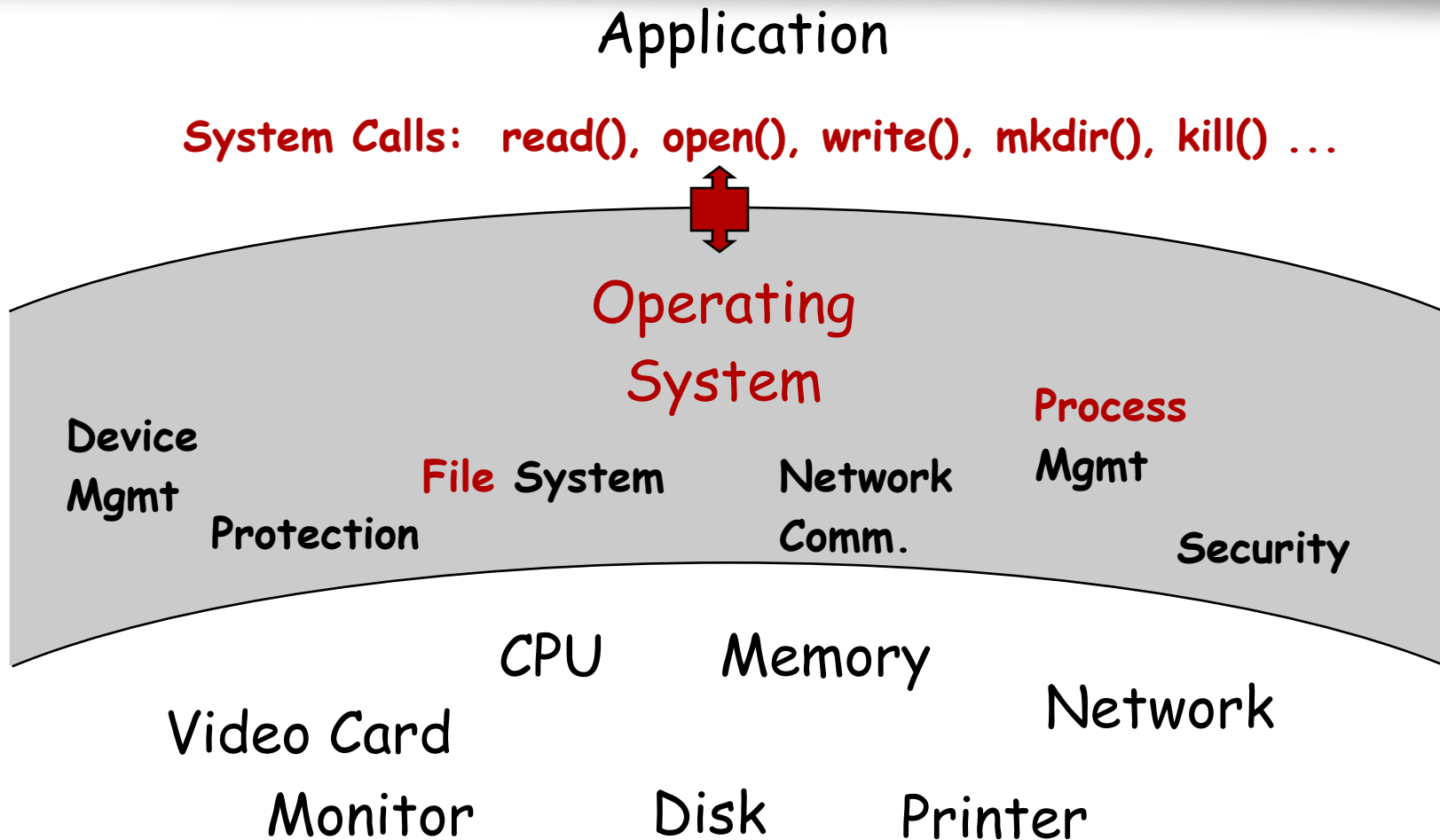
# Providing abstraction via system calls

6



# Providing abstraction via system calls

7



# OS as a resource manager

8

# OS as a resource manager

8

- **Allocating resources to applications across space and time**
  - time sharing a resource (scheduling)
  - space sharing a resource (allocation)

- **Allocating resources to applications across space and time**
  - time sharing a resource (scheduling)
  - space sharing a resource (allocation)
- **Making efficient use of limited resources**
  - improving utilization
  - minimizing overhead
  - improving throughput/good put



# OS as a resource manager

8

- **Allocating resources to applications across space and time**
  - time sharing a resource (scheduling)
  - space sharing a resource (allocation)
- **Making efficient use of limited resources**
  - improving utilization
  - minimizing overhead
  - improving throughput/good put
- **Protecting applications from each other**
  - enforcement of boundaries

# Problems an OS must solve

9

- Time sharing the CPU among applications
- Space sharing the memory among applications
- Space sharing the disk among users
- Time sharing access to the disk
- Time sharing access to the network

- **Protection**

- of applications from each other
- of user data from other users
- of hardware/devices
- of the OS itself!

- **The OS is just a program! It needs help from the hardware to accomplish these tasks!**

- When an application is running, the OS is not running!
- When the OS is not running, it can't do anything!

# Operating System Concepts

TENTH EDITION

ABRAHAM SILBERSCHATZ • PETER BAER GALVIN • GREG GAGNE



WILEY

کتاب درس

# PART ONE ■ OVERVIEW

## Chapter 1 Introduction

- 1.1 What Operating Systems Do 4
- 1.2 Computer-System Organization 7
- 1.3 Computer-System Architecture 15
- 1.4 Operating-System Operations 21
- 1.5 Resource Management 27
- 1.6 Security and Protection 33
- 1.7 Virtualization 34
- 1.8 Distributed Systems 35
- 1.9 Kernel Data Structures 36
- 1.10 Computing Environments 40
- 1.11 Free and Open-Source Operating Systems 46
- Practice Exercises 53
- Further Reading 54

## Chapter 2 Operating-System Structures

- 2.1 Operating-System Services 55
- 2.2 User and Operating-System Interface 58
- 2.3 System Calls 62
- 2.4 System Services 74
- 2.5 Linkers and Loaders 75
- 2.6 Why Applications Are Operating-System Specific 77
- 2.7 Operating-System Design and Implementation 79
- 2.8 Operating-System Structure 81
- 2.9 Building and Booting an Operating System 92
- 2.10 Operating-System Debugging 95
- 2.11 Summary 100
- Practice Exercises 101
- Further Reading 101

# **PART TWO ■ PROCESS MANAGEMENT**

## **Chapter 3 Processes**

3.1 Process Concept	106	3.7 Examples of IPC Systems	132
3.2 Process Scheduling	110	3.8 Communication in Client–	
3.3 Operations on Processes	116	Server Systems	145
3.4 Interprocess Communication	123	3.9 Summary	153
3.5 IPC in Shared-Memory Systems	125	Practice Exercises	154
3.6 IPC in Message-Passing Systems	127	Further Reading	156

## **Chapter 4 Threads & Concurrency**

4.1 Overview	160	4.6 Threading Issues	188
4.2 Multicore Programming	162	4.7 Operating-System Examples	194
4.3 Multithreading Models	166	4.8 Summary	196
4.4 Thread Libraries	168	Practice Exercises	197
4.5 Implicit Threading	176	Further Reading	198

## **Chapter 5 CPU Scheduling**

5.1 Basic Concepts	200	5.7 Operating-System Examples	234
5.2 Scheduling Criteria	204	5.8 Algorithm Evaluation	244
5.3 Scheduling Algorithms	205	5.9 Summary	250
5.4 Thread Scheduling	217	Practice Exercises	251
5.5 Multi-Processor Scheduling	220	Further Reading	254
5.6 Real-Time CPU Scheduling	227		

# PART THREE ■ PROCESS SYNCHRONIZATION

## Chapter 6 Synchronization Tools

6.1 Background	257	6.7 Monitors	276
6.2 The Critical-Section Problem	260	6.8 Liveness	283
6.3 Peterson's Solution	262	6.9 Evaluation	284
6.4 Hardware Support for Synchronization	265	6.10 Summary	286
6.5 Mutex Locks	270	Practice Exercises	287
6.6 Semaphores	272	Further Reading	288

## Chapter 7 Synchronization Examples

7.1 Classic Problems of Synchronization	289	7.5 Alternative Approaches	311
7.2 Synchronization within the Kernel	295	7.6 Summary	314
7.3 POSIX Synchronization	299	Practice Exercises	314
7.4 Synchronization in Java	303	Further Reading	315

## Chapter 8 Deadlocks

8.1 System Model	318	8.6 Deadlock Avoidance	330
8.2 Deadlock in Multithreaded Applications	319	8.7 Deadlock Detection	337
8.3 Deadlock Characterization	321	8.8 Recovery from Deadlock	341
8.4 Methods for Handling Deadlocks	326	8.9 Summary	343
8.5 Deadlock Prevention	327	Practice Exercises	344
		Further Reading	346



# PART FOUR ■ MEMORY MANAGEMENT

## Chapter 9 Main Memory

- 9.1 Background 349
- 9.2 Contiguous Memory Allocation 356
- 9.3 Paging 360
- 9.4 Structure of the Page Table 371
- 9.5 Swapping 376
- 9.6 Example: Intel 32- and 64-bit Architectures 379
- 9.7 Example: ARMv8 Architecture 383
- 9.8 Summary 384
  - Practice Exercises 385
  - Further Reading 387

## Chapter 10 Virtual Memory

- 10.1 Background 389
- 10.2 Demand Paging 392
- 10.3 Copy-on-Write 399
- 10.4 Page Replacement 401
- 10.5 Allocation of Frames 413
- 10.6 Thrashing 419
- 10.7 Memory Compression 425
- 10.8 Allocating Kernel Memory 426
- 10.9 Other Considerations 430
- 10.10 Operating-System Examples 436
- 10.11 Summary 440
  - Practice Exercises 441
  - Further Reading 444



# PART FIVE ■ STORAGE MANAGEMENT

## Chapter 11 Mass-Storage Structure

- |   |     |                            |     |
|---|-----|----------------------------|-----|
| 11.1 Overview of Mass-Storage Structure | 449 | 11.6 Swap-Space Management | 467 |
| 11.2 HDD Scheduling                     | 457 | 11.7 Storage Attachment    | 469 |
| 11.3 NVM Scheduling                     | 461 | 11.8 RAID Structure        | 473 |
| 11.4 Error Detection and Correction     | 462 | 11.9 Summary               | 485 |
| 11.5 Storage Device Management          | 463 | Practice Exercises         | 486 |
|   |     | Further Reading            | 487 |

## Chapter 12 I/O Systems

- |   |     |                    |     |
|---|-----|--------------------|-----|
| 12.1 Overview   | 489 | 12.6 STREAMS       | 519 |
| 12.2 I/O Hardware                                     | 490 | 12.7 Performance   | 521 |
| 12.3 Application I/O Interface                        | 500 | 12.8 Summary       | 524 |
| 12.4 Kernel I/O Subsystem                             | 508 | Practice Exercises | 525 |
| 12.5 Transforming I/O Requests to Hardware Operations | 516 | Further Reading    | 526 |

# PART SIX ■ FILE SYSTEM

## Chapter 13 File-System Interface

- |                          |     |                          |     |
|--------------------------|-----|--------------------------|-----|
| 13.1 File Concept        | 529 | 13.5 Memory-Mapped Files | 555 |
| 13.2 Access Methods      | 539 | 13.6 Summary             | 560 |
| 13.3 Directory Structure | 541 | Practice Exercises       | 560 |
| 13.4 Protection          | 550 | Further Reading          | 561 |

## Chapter 14 File-System Implementation

- |                                 |     |                                    |     |
|---------------------------------|-----|------------------------------------|-----|
| 14.1 File-System Structure      | 564 | 14.7 Recovery                      | 586 |
| 14.2 File-System Operations     | 566 | 14.8 Example: The WAFL File System | 589 |
| 14.3 Directory Implementation   | 568 | 14.9 Summary                       | 593 |
| 14.4 Allocation Methods         | 570 | Practice Exercises                 | 594 |
| 14.5 Free-Space Management      | 578 | Further Reading                    | 594 |
| 14.6 Efficiency and Performance | 582 |                                    |     |

## Chapter 15 File-System Internals

- |                              |     |                            |     |
|------------------------------|-----|----------------------------|-----|
| 15.1 File Systems            | 597 | 15.7 Consistency Semantics | 608 |
| 15.2 File-System Mounting    | 598 | 15.8 NFS                   | 610 |
| 15.3 Partitions and Mounting | 601 | 15.9 Summary               | 615 |
| 15.4 File Sharing            | 602 | Practice Exercises         | 616 |
| 15.5 Virtual File Systems    | 603 | Further Reading            | 617 |
| 15.6 Remote File Systems     | 605 |                            |     |

## **PART SEVEN ■ SECURITY AND PROTECTION**

### **Chapter 16 Security**

- 16.1 The Security Problem 621
- 16.2 Program Threats 625
- 16.3 System and Network Threats 634
- 16.4 Cryptography as a Security Tool 637
- 16.5 User Authentication 648
- 16.6 Implementing Security Defenses 653
- 16.7 An Example: Windows 10 662
- 16.8 Summary 664
- Further Reading 665

### **Chapter 17 Protection**

- 17.1 Goals of Protection 667
- 17.2 Principles of Protection 668
- 17.3 Protection Rings 669
- 17.4 Domain of Protection 671
- 17.5 Access Matrix 675
- 17.6 Implementation of the Access Matrix 679
- 17.7 Revocation of Access Rights 682
- 17.8 Role-Based Access Control 683
- 17.9 Mandatory Access Control (MAC) 684
- 17.10 Capability-Based Systems 685
- 17.11 Other Protection Improvement Methods 687
- 17.12 Language-Based Protection 690
- 17.13 Summary 696
- Further Reading 697

## **PART EIGHT ■ ADVANCED TOPICS**

### **Chapter 18 Virtual Machines**

- 18.1 Overview 701
- 18.2 History 703
- 18.3 Benefits and Features 704
- 18.4 Building Blocks 707
- 18.5 Types of VMs and Their Implementations 713
- 18.6 Virtualization and Operating-System Components 719
- 18.7 Examples 726
- 18.8 Virtualization Research 728
- 18.9 Summary 729
- Further Reading 730

### **Chapter 19 Networks and Distributed Systems**

- 19.1 Advantages of Distributed Systems 733
- 19.2 Network Structure 735
- 19.3 Communication Structure 738
- 19.4 Network and Distributed Operating Systems 749
- 19.5 Design Issues in Distributed Systems 753
- 19.6 Distributed File Systems 757
- 19.7 DFS Naming and Transparency 761
- 19.8 Remote File Access 764
- 19.9 Final Thoughts on Distributed File Systems 767
- 19.10 Summary 768
- Practice Exercises 769
- Further Reading 770

## **PART NINE ■ CASE STUDIES**

### **Chapter 20 The Linux System**

- 20.1 Linux History 775
- 20.2 Design Principles 780
- 20.3 Kernel Modules 783
- 20.4 Process Management 786
- 20.5 Scheduling 790
- 20.6 Memory Management 795
- 20.7 File Systems 803
- 20.8 Input and Output 810
- 20.9 Interprocess Communication 812
- 20.10 Network Structure 813
- 20.11 Security 816
- 20.12 Summary 818
- Practice Exercises 819
- Further Reading 819

### **Chapter 21 Windows 10**

- 21.1 History 821
- 21.2 Design Principles 826
- 21.3 System Components 838
- 21.4 Terminal Services and Fast User Switching 874
- 21.5 File System 875
- 21.6 Networking 880
- 21.7 Programmer Interface 884
- 21.8 Summary 895
- Practice Exercises 896
- Further Reading 897

# PART TEN ■ APPENDICES

## Chapter A Influentia Operating Systems

- A.1 Feature Migration 1
- A.2 Early Systems 2
- A.3 Atlas 9
- A.4 XDS-940 10
- A.5 THE 11
- A.6 RC 4000 11
- A.7 CTSS 12
- A.8 MULTICS 13
- A.9 IBM OS/360 13
- A.10 TOPS-20 15
- A.11 CP/M and MS/DOS 15
- A.12 Macintosh Operating System and Windows 16
- A.13 Mach 16
- A.14 Capability-based Systems—Hydra and CAP 18
- A.15 Other Systems 20
- Further Reading 21

## Chapter B Windows 7

- B.1 History 1
- B.2 Design Principles 3
- B.3 System Components 10
- B.4 Terminal Services and Fast User Switching 34
- B.5 File System 35
- B.6 Networking 41
- B.7 Programmer Interface 46
- B.8 Summary 55
- Practice Exercises 55
- Further Reading 56

## Chapter C BSD UNIX

- C.1 UNIX History 1
- C.2 Design Principles 6
- C.3 Programmer Interface 8
- C.4 User Interface 15
- C.5 Process Management 18
- C.6 Memory Management 22
- C.7 File System 25
- C.8 I/O System 33
- C.9 Interprocess Communication 36
- C.10 Summary 41
- Further Reading 42



# Introduction



An **operating system** is software that manages a computer's hardware. It also provides a basis for application programs and acts as an intermediary between the computer user and the computer hardware. An amazing aspect of operating systems is how they vary in accomplishing these tasks in a wide variety of computing environments. Operating systems are everywhere, from cars and home appliances that include “Internet of Things” devices, to smart phones, personal computers, enterprise computers, and cloud computing environments.

In order to explore the role of an operating system in a modern computing environment, it is important first to understand the organization and architecture of computer hardware. This includes the CPU, memory, and I/O devices, as well as storage. A fundamental responsibility of an operating system is to allocate these resources to programs.

Because an operating system is large and complex, it must be created piece by piece. Each of these pieces should be a well-delineated portion of the system, with carefully defined inputs, outputs, and functions. In this chapter, we provide a general overview of the major components of a contemporary computer system as well as the functions provided by the operating system. Additionally, we cover several topics to help set the stage for the remainder of the text: data structures used in operating systems, computing environments, and open-source and free operating systems.

## CHAPTER OBJECTIVES

- Describe the general organization of a computer system and the role of interrupts.
- Describe the components in a modern multiprocessor computer system.
- Illustrate the transition from user mode to kernel mode.
- Discuss how operating systems are used in various computing environments.
- Provide examples of free and open-source operating systems.

## 1.12 Summary

- An operating system is software that manages the computer hardware, as well as providing an environment for application programs to run.
- Interrupts are a key way in which hardware interacts with the operating system. A hardware device triggers an interrupt by sending a signal to the CPU to alert the CPU that some event requires attention. The interrupt is managed by the interrupt handler.
- For a computer to do its job of executing programs, the programs must be in main memory, which is the only large storage area that the processor can access directly.
- The main memory is usually a volatile storage device that loses its contents when power is turned off or lost.



## Practice Exercises

- 1.1 What are the three main purposes of an operating system?
- 1.2 We have stressed the need for an operating system to make efficient use of the computing hardware. When is it appropriate for the operating system to forsake this principle and to “waste” resources? Why is such a system not really wasteful?
- 1.3 What is the main difficulty that a programmer must overcome in writing an operating system for a real-time environment?
- 1.4 Keeping in mind the various definitions of *operating system*, consider whether the operating system should include applications such as web browsers and mail programs. Argue both that it should and that it should not, and support your answers.

## Programming Problems

- 2.24** In Section 2.3, we described a program that copies the contents of one file to a destination file. This program works by first prompting the user for the name of the source and destination files. Write this program using either the POSIX or Windows API. Be sure to include all necessary error checking, including ensuring that the source file exists.

Once you have correctly designed and tested the program, if you used a system that supports it, run the program using a utility that traces system calls. Linux systems provide the `strace` utility, and macOS systems use the `dtruss` command. (The `dtruss` command, which actually is a front end to `dtrace`, requires admin privileges, so it must be run using `sudo`.) These tools can be used as follows (assume that the name of the executable file is `FileCopy`):

**Linux:**

```
strace ./FileCopy
```

**macOS:**

```
sudo dtruss ./FileCopy
```

Since Windows systems do not provide such a tool, you will have to trace through the Windows version of this program using a debugger.

## Programming Projects

### Introduction to Linux Kernel Modules

In this project, you will learn how to create a kernel module and load it into the Linux kernel. You will then modify the kernel module so that it creates an entry in the `/proc` file system. The project can be completed using the Linux virtual machine that is available with this text. Although you may use any text editor to write these C programs, you will have to use the *terminal* application to compile the programs, and you will have to enter commands on the command line to manage the modules in the kernel.

As you'll discover, the advantage of developing kernel modules is that it is a relatively easy method of interacting with the kernel, thus allowing you to write programs that directly invoke kernel functions. It is important for you to keep in mind that you are indeed writing *kernel code* that directly interacts with the kernel. That normally means that any errors in the code could crash the system! However, since you will be using a virtual machine, any failures will at worst only require rebooting the system.

- شناخت سخت افزار
- $\leq$  اصول سیستم
- برنامه نویسی زیاد

- پایان ترم: ۶ نمره.

- میان ترم: ۵ نمره.

- آزمونک‌ها: ۴ نمره (یکی از آزمونک‌ها برای هر فردی حذف می‌شود).

- پروژه: ۵ نمره.

- مجموعاً پروژه‌ها را می‌توانید ۷ روز با تاخیر تحویل بدهند بدون کم شدن

نمره.

# The BLITZ Home Page

<http://web.cecs.pdx.edu/~harry/Blitz/index.html>

# کلاس‌های حل تمرین

- چند کلاس حل تمرین برای پروژه‌ها
- اولین جلسه: راه‌اندازی پروژه
- زمان جلسه‌ها در یک نظر سنجی مشخص می‌شود.

• آقای توفیقی

• خانم قاسمی

• آقای رئیسی

• آقای کشیری



**CS 333**

**Introduction to Operating Systems**

**Class 1 - Introduction to OS-related  
Hardware and Software**

**Jonathan Walpole**

**Computer Science**

**Portland State University**

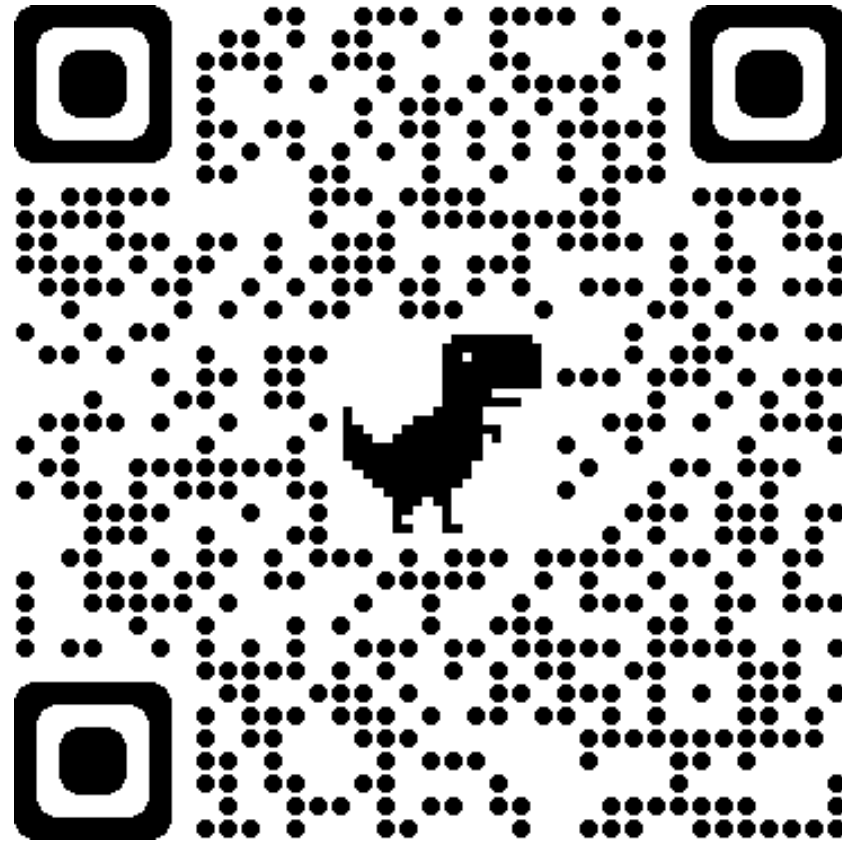
جلسه	تاریخ	موضوع جلسه	آزموز	تمرین
1	۹۹/۱۱/۲۶	Course Overview and Introduction to Operating Systems Reading: Chapters 1 and 2		
2	۹۹/۱۱/۲۸	The Process Concept Reading: Chapter 3		
3	۹۹/۱۲/۰۳	Threads and Concurrency Reading: Chapter 4		حل تمرین ۱ (این هفته)
4	۹۹/۱۲/۰۵			
5	۹۹/۱۲/۱۰	Synchronization Primitives Reading: Chapter 6	آزمونک ۱	تمرین سری ۱ (یک هفته)
6	۹۹/۱۲/۱۲			
7	۹۹/۱۲/۱۷	Classic Synchronization Problems Reading: Chapter 6		تمرین سری ۲ (دو هفته)
8	۹۹/۱۲/۱۹		آزمونک ۲	حل تمرین ۲ (این هفته)
9	۹۹/۱۲/۲۴	Monitors and Message Passing Reading: Chapter 6		
10	۹۹/۱۲/۲۶			

تمرین سری ۳ (سه هفته)		Deadlock Reading: Chapter 6	۰۰/۱/۱۵	11
حل تمرین ۳			۰۰/۱/۱۷	12
		Scheduling Reading: Chapter 5	۰۰/۱/۲۲	13
		میان‌ترم	۰۰/۱/۲۴	
			۰۰/۱/۲۹	14
	آزمونک ۳	Memory Management Reading: Chapter 7	۰۰/۱/۳۱	15
			۰۰/۲/۰۵	16
تمرین سری ۴ (دو هفته)		Virtual Memory 1 Reading: Chapter 8	۰۰/۲/۰۷	17
حل تمرین ۴	آزمونک ۴	Virtual Memory 2 Reading: Chapter 8	۰۰/۲/۱۲	18
		Virtual Memory 3 Reading: Chapter 8	۰۰/۲/۱۴	19
		Paging Algorithms Reading: Chapter 8	۰۰/۲/۱۹	20

			۰۰/۲/۲۱	21
تمرین سری ۵ (۷۵) (چهار هفته)	آزمونک ۵	Input/Output Reading: Chapter 12	۰۰/۲/۲۶	22
			۰۰/۲/۲۸	23
		Secondary Storage Management Reading: Chapter 11	۰۰/۳/۰۲	24
			۰۰/۳/۰۴	25
		File Systems 1 Reading: Chapters 9 and 10	۰۰/۳/۰۹	26
			۰۰/۳/۱۱	27
		تعطیل رسمی	۰۰/۳/۱۶	
		File Systems 2 Reading: Chapter 10	۰۰/۳/۱۸	28

# علم، مخصوصا کامپیوترش

اخبار و نظرات مرتبط با علوم کامپیوتر و قسمتی از زیست‌شناسی که به آن مرتبط است.



[http://foroughmand.ir/?page\\_id=699](http://foroughmand.ir/?page_id=699)

سوال؟