DATA STRUCTURES, ALGORITHM DESIGN AND C CONCEPTS

DURATION: 48 HRS

BASIC CONCEPTS: Pointers and Dynamic Memory Allocation, Algorithm Specification, Data Abstraction, Performance Analysis, Performance and Measurement

ARRAYS and STRUCTURES: Arrays, Dynamically Allocated Arrays, Structures and Unions, Polynomials, Sparse Matrices, Representation of Multidimensional Arrays

STACKS AND QUEUES: Stacks, Stacks Using Dynamic Arrays, Queues, Circular Queues Using Dynamic Arrays, Evaluation of Expressions, Multiple Stacks and Queues

LINKED LISTS: Singly Linked lists and Chains, Representing Chains in C, Linked Stacks and Queues, Polynomials, Additional List operations, Sparse Matrices, Doubly Linked Lists

TREES – 1: Introduction, Binary Trees, Binary Tree Traversals, Threaded Binary Trees, Heaps**.**
TREES – 2: **GRAPHS:** Binary Search Trees, Selection Trees, Forests, Representation of Disjoint Sets, Counting Binary Trees, the Graph Abstract Data Type.

PRIORITY QUEUES: Single- and Double-Ended Priority Queues, Leftist, Binomial Heaps, Fibonacci Heaps, Pairing Heaps.

 INTRODUCTION TO ALOGORITHM DESIGN: Notion of Algorithm, Review of Asymptotic Notations, Mathematical Analysis of Non-Recursive and Recursive Algorithms

Brute Force Approaches: Introduction, Selection Sort and Bubble Sort, Sequential Search and Brute Force String Matching.

COMMANDS AND SHELL PROGRAMMING

DURATION: 40 HRS

The Linux Operating System, the LINUX architecture and Command Usage, The File System, Basic File Attributes, the VI Editor, the Shell, the Process, Customizing the environment, Essential Shell Programming

- ♣ Quick Introduction to Linux
- ♣ What Linux is?
- ♣ Who developed the Linux?
- ♣ How to get Linux?
- ♣ How to Install Linux
- ♣ Where I can use Linux?
- ♣ What Kernel Is?
- ♣ What is Linux Shell?
- ♣ How to use Shell
- ♣ What is Shell Script?
- ♣ Why to Write Shell Script?
- ♣ More on Shell...
- ♣ Getting started with Shell Programming
- ♣ How to write shell script
- ♣ Variables in shell
- ♣ How to define User defined variables (UDV)
- ♣ Rules for Naming variable name (Both UDV and System Variable)
- ♣ How to print or access value of UDV (User defined variables)
- ♣ Echo Command
- ♣ Shell Arithmetic
- ♣ More about Quotes
- ♣ Exit Status
- ♣ The read Statement
- ♣ Wild cards (Filename Shorthand or meta Characters)
- ♣ More commands on one command line
- ♣ Command Line Processing
- ♣ Why Command Line arguments required
- ♣ Redirection of Standard output/input i.e. Input - Output redirection
- ♣ Pipes
- ♣ Filter
- ♣ What is Processes?
- ♣ Why Process required
- ♣ Linux Command(s) Related with Process

Example Scripts:

INTRODUCTION TO OPERATING SYSTEMS AND SYSTEM STRUCTURES

DURATION: 25 HRS

What operating systems do; Computer System organization; Computer System architecture; Operating System structure; Operating System operations; Process management; Memory management; Storage management; Protection and security;

Process Management: Process concept; Process scheduling; Inter-process communication. Multi-Threaded Programming: Overview; Multithreading models; Process Scheduling: Basic concepts; Scheduling criteria; Scheduling algorithms; Multiple-Processor scheduling; Thread scheduling.

Process Synchronization: Synchronization: The Critical section problem; Peterson's solution; Synchronization hardware; Semaphores;

Deadlocks: System model; Deadlock characterization; Methods for handling deadlocks; Deadlock prevention; Deadlock avoidance; Deadlock detection and recovery from deadlock.

Memory Management: Memory Management Strategies; Background; Swapping; Contiguous memory allocation; Paging; Structure of page table; Segmentation.

Virtual Memory Management: Background; Demand paging; Copy-on-write; Page replacement; Allocation of frames; Thrashing.

File System: File concept; Access methods; Directory structure; File system mounting; file sharing; Protection.

Implementing File System: File system structure; File system implementation; Directory implementation; Allocation methods; Free space management

LINUX SYSTEM PROGRAMMING

DURATION: 40 HRS

Introduction and Essential Concepts

System Programming, APIs and ABIs, Standards, Concepts of Linux Programming, Getting Started with System Programming, File I/O, Opening Files, Reading via read( ), Writing with write( ), Synchronized I/O, Direct I/O, Closing Files, Seeking with lseek( ), Positional Reads and Writes, Truncating Files, Multiplexed I/O, Kernel Internals, Buffered I/O User-Buffered I/O, Standard I/O, Opening Files, Opening a Stream via File Descriptor, Closing Streams, Reading from a Stream, Writing to a Stream, Sample Program Using Buffered I/O, Seeking a Stream, Flushing a Stream, Errors and End-of-File, Obtaining the Associated File Descriptor, Controlling the Buffering, Thread Safety , Critiques of Standard I/O

Advanced File I/O: Scatter/Gather I/O, The Event Poll Interface, Mapping Files into Memory, Advice for Normal File I/O, Synchronized, Synchronous, and Asynchronous Operations , I/O Schedulers and I/O Performance

Process Management: The Process ID, Running a New Process, Terminating a Process, Waiting for Terminated Child Processes, Users and Groups , Sessions and Process Groups and Daemons

Advanced Process Management: Process Scheduling, Yielding the Processor, Process Priorities, Processor Affinity, Real-Time Systems Resource Limits

File and Directory Management: Files and Their Metadata, Directories, Links, Copying and Moving Files, Device Nodes, Out-of-Band Communication, Monitoring, File Events

Memory Management: The Process Address Space, Allocating Dynamic Memory, Managing the Data Segment, Anonymous Memory Mappings, Advanced Memory Allocation, Debugging Memory Allocations Stack-Based Allocations, Choosing a Memory Allocation Mechanism, Manipulating Memory, Locking Memory, Opportunistic Allocation

Signals: Signal Concepts, Basic Signal Management, Sending a Signal, Reentrancy, Signal Sets, Blocking Signals Advanced Signal Management, Sending a Signal with a Payload

Introduction to Linux Internals

DURATION: 50 HRS

Introduction and environmental Setup, Main characteristics of the Linux, Operating system, Linux distributions, Kernel configuration facilities, Building the kernel, Location of components, Compiling

Kernel Overview

structures, Processes and tasks, Files and in nodes, Dynamic memory management, Queues and semaphores, System time and timers, Primary algorithms, Signals, Interrupts, System booting, Timer interrupt Scheduler, System call implementation, Description of system calls

Practical examples: adding new system calls

Memory Management

Architecture-dependent memory, Model Pages, Virtual address space, linear address conversion, Page table and page directory, Middle page directory, Virtual address space model, User segment, Virtual memory, the bark system call, Mapping functions, Kernel segment, Static and dynamic memory, Allocation in the kernel, Block device caching, Block buffering, Update and bdflush, List structures for the buffer cache, How to use the buffer cache, Paging in Linux, Page cache management, Finding free pages, Page exception sinter-Process Communication, Synchronization, communication via files, Pipes , Debugging using ptrace, System V IPC, Socket-based communications

File System

Basic aspects, VFS, mouting a file system, Superblock, the I node concept and operations, File operations, Directory cache, Proc file system, Ext2 file system, Structure, Directories, Block allocation, Extensions, System Calls, Initialization, Process management, Memory management, File system

Kernel-Related Commands

Ps, Top, Free, Init, Shutdown, Strace, trace route, mount

Device Drivers

Character vs. block devices, polling and interrupts, polling and interrupt mode, Interrupt sharing, Bottom halves, Task queues, Implementation, Setup, Open and release, Read and write, IOCtl, Select, lseek, mmap, readdir, fsync and fasync, Module Management, Interfaces to modules, Adding/removing modules to the kernel

Insmod, Modprobe, Rmmod, Implementation details, Network Layer model, Network communications, Data structures, Socket, sk_buff, Inet socket, Proto, Devices, Ethernet, SLIP and PPP, Loopback, Dummy device Protocols, Arp, IP, Functions, Routing, Multicasting, Packet filters, Accounting, Firewalls, UDP, Standard and extra functions, TCP Standard functions, Communication details SCSI Subsystem, Architecture overview, Names and conventions, Upper level, Block devices (hard disks, CD-ROM), Character devices (tape), Generic drivers, mid level (boot parameters, proc Interface), Lower (hardware) level and pseudo

Drivers

Boot Process, Booting details, LILO, Started by MBRand Started by a boot manager, Structure in the MBR,Files,Parameters,Start-up messages, Error messages

Debugging Tools

Ptrace, SysRq, KDB, User Mode Linux, Kgdb

DEVICE DRIVERS

DURATION: 50 HRS

The Role of the Device Driver, Splitting the Kernel, Classes of Devices and Modules, Security Issues Version Numbering, License Terms, Joining the Kernel Development Community

Building and Running Modules

Setting Up Your Test System, the Hello World Module, Kernel Modules Versus Applications, Compiling and Loading, The Kernel Symbol Table, Preliminaries, Initialization and Shutdown, Module Parameters, Doing It in User Space

Char Drivers

The Design of scull, Major and Minor Numbers, Some Important Data Structures, v Char Device Registration, open and release, Scull's Memory Usage, read and write, Playing with the New Devices

Debugging Techniques

Debugging Support in the Kernel, Debugging by Printing, Debugging by Querying, Debugging by Watching, Debugging System Faults, Debuggers and Related Tools

Concurrency and Race Conditions

Pitfalls in scull, Concurrency and Its Management, Semaphores and Mutexes, Completions Spinlocks, Locking Traps, Alternatives to Locking

Advanced Char Driver Operations

Ioctl, Blocking I/O, poll and select, Asynchronous Notification, Seeking a Device, Access Control on a Device File

Time, Delays, and Deferred Work

Measuring Time Lapses, Knowing the Current Time, Delaying Execution, Kernel Timers, Tasklets

Allocating Memory

The Real Story of kmalloc, Lookaside Caches, get_free_page and Friends, vmalloc and Friends 224 Per-CPU Variables, Obtaining Large Buffers

Communicating with Hardware

I/O Ports and I/O Memory, Using I/O Ports, An I/O Port Example, Using I/O Memory

Interrupt Handling

Preparing the Parallel Port, Installing an Interrupt Handler, Implementing a Handler, Top and Bottom Halves, Interrupt Sharing, Interrupt-Driven I/O

Data Types in the Kernel

Use of Standard C Types, Assigning an Explicit Size to Data Items, Interface-Specific Types, Other Portability Issues, Linked Lists

PCI Drivers

The PCI Interface, A Look Back: ISA, PC/104 and PC/104+, Other PC Buses, SBus, NuBus,External Buses

USB Drivers

USB Device Basics, USB and Sysfs, USB Urbs, Writing a USB Driver, USB Transfers Without Urbs

The Linux Device Model

Kobjects, Ksets, and Subsystems, Low-Level Sysfs Operations, Hotplug Event Generation, Buses, Devices, and Drivers, Classes, Putting It All Together, Hotplug, Dealing with Firmware

Memory Mapping and DMA

Memory Management in Linux, The mmap Device Operation, Performing Direct I/O, Direct Memory Access

Block Drivers

Registration, The Block Device Operations, Request Processing, Some Other Details 491

Network Drivers

How snull Is Designed, Connecting to the Kernel, The net_device Structure in Detail, Opening and Closing, Packet Transmission, Packet Reception, The Interrupt Handler, Receive Interrupt, Mitigation, Changes in Link State, The Socket Buffers, MAC Address Resolution, Custom ioctl Commands, Statistical Information, Multicast

TTY Drivers

A Small TTY Driver, tty_driver Function Pointers, TTY Line Settings, ioctls, proc and sysfs Handling of TTY Devices, The tty_driver Structure in Detail, The tty_operations Structure in Detail, The tty_struct Structure in Detail