Lesson 1: Programming Fundamentals

Aniruddha Kudalkar

Objective

- Basics of Computer
- Linux Introduction
- Basics of Programming
- Compiler and Interpreter

Basics of Computer

VENGEANCE
CORSAIR

Computer

Lets us understand first, very basics of programming

- Hardware
- Software
- Firmware

Hardware

Computer hardware includes the physical parts of a computer, such as the case, central processing unit (CPU), monitor, mouse, keyboard, computer data storage, graphics card, sound card, speakers and motherboard.

Software

Software is a collection of instructions and data that tell a computer how to work. Software includes programs, libraries and related non-executable data, such as online documentation or digital media.

Firmware

Specific class of computer software that provides the low-level control for a device's specific hardware. Firmware, such as the BIOS of a personal computer, may contain basic functions of a device, and may provide hardware abstraction services to higher-level software such as operating systems.

Hardware

Physical parts of computer, which includes motherboard to IO devices

- Basic Components
- Motherboard
- Input/Output Devices

Basic Components

Monitor, Cabinet, Motherboard, CPU, GPU, Power Cables, Mouse, Keyboard, SMPS, Fans, RAM, ROM, HDD/SSD etc

Motherboard

A motherboard is the main printed circuit board (PCB) in general-purpose computers and other expandable systems. It holds and allows communication between many of the crucial electronic components of a system, such as the central processing unit (CPU) and memory, and provides connectors for other peripherals.

Motherboard

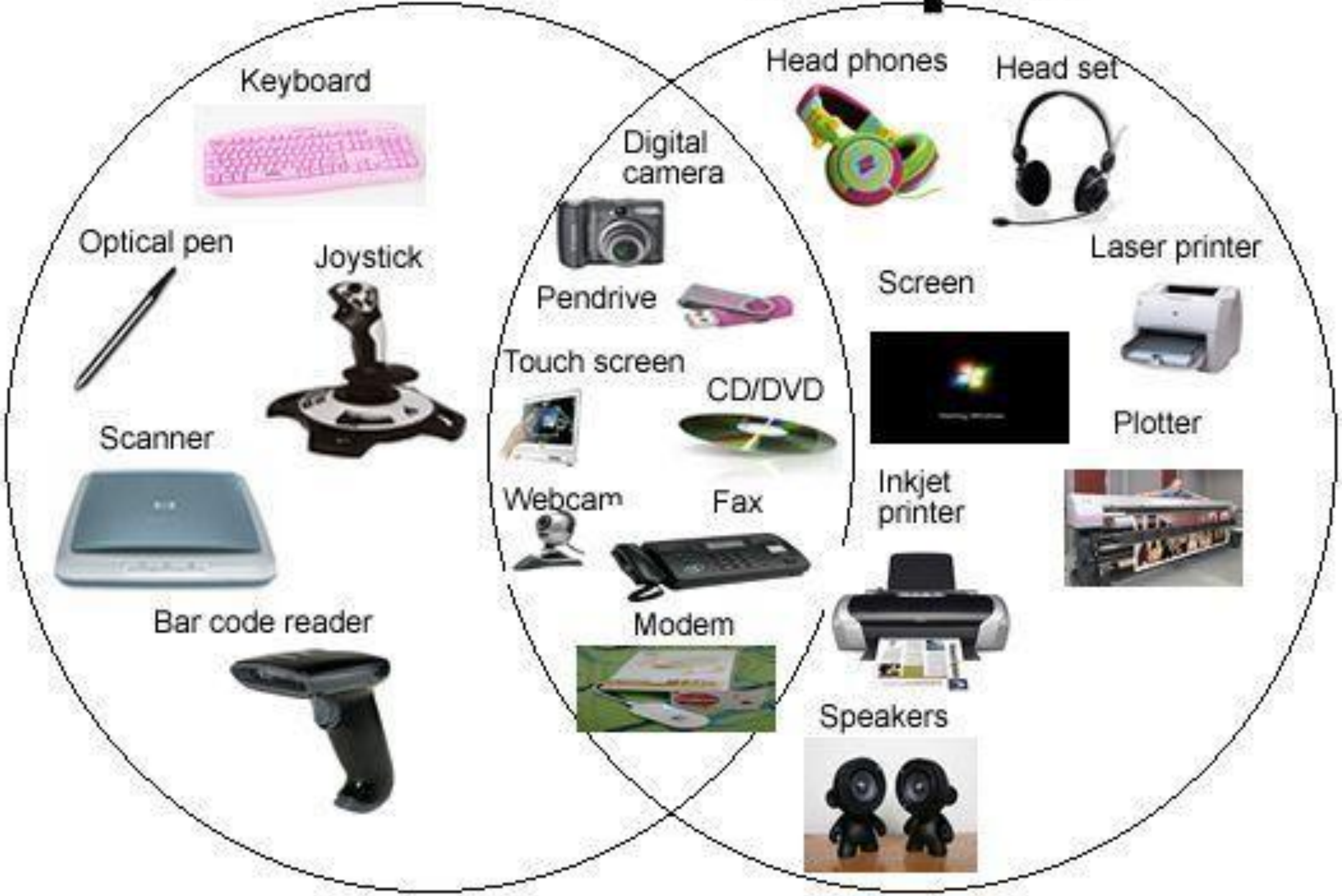
A motherboard usually contains significant sub-systems, such as the central processor, the chipset's input/output and memory controllers, interface connectors, and other components integrated for general use.

I/O Devices

I/O devices are the pieces of hardware used by a human (or other system) to communicate with a computer. For instance, a keyboard or computer mouse is an input device for a computer, while monitors and printers are output devices. Devices for communication between computers, such as modems and network cards, typically perform both input and output operations.

input

output



Software

Includes program, applications, data which can not be touched

- Types of Software
- Operating System
- Writing Softwares

Types Of Software

Softwares are divided into many sub categories as per their usages

Types of Software

Based on the goal, computer software can be divided into: Application Software, System Software and Malicious Software.

Application Software

uses the computer system to perform special functions beyond the basic operation of the computer itself. There are many different types of application software because the range of tasks that can be performed with a modern computer is so large

Application Software

- Image Editors
- Document Processors
- Video Players
- Calculators
- Calendar

System Software

manages hardware behaviour, as to provide basic functionalities that are required by users, or for other software to run properly, if at all. System software is also designed for providing a platform for running application software,

System Software

- *Operating System*, essential collections of software that manage resources
- *Device Drivers*, operate or control a particular type of device
- *Utilities*, programs designed to assist users

Malicious Software

or malware, is software that is developed to harm or disrupt computers. Malware is closely associated with computer-related crimes, though some malicious programs may have been designed as practical jokes.

Malicious Software

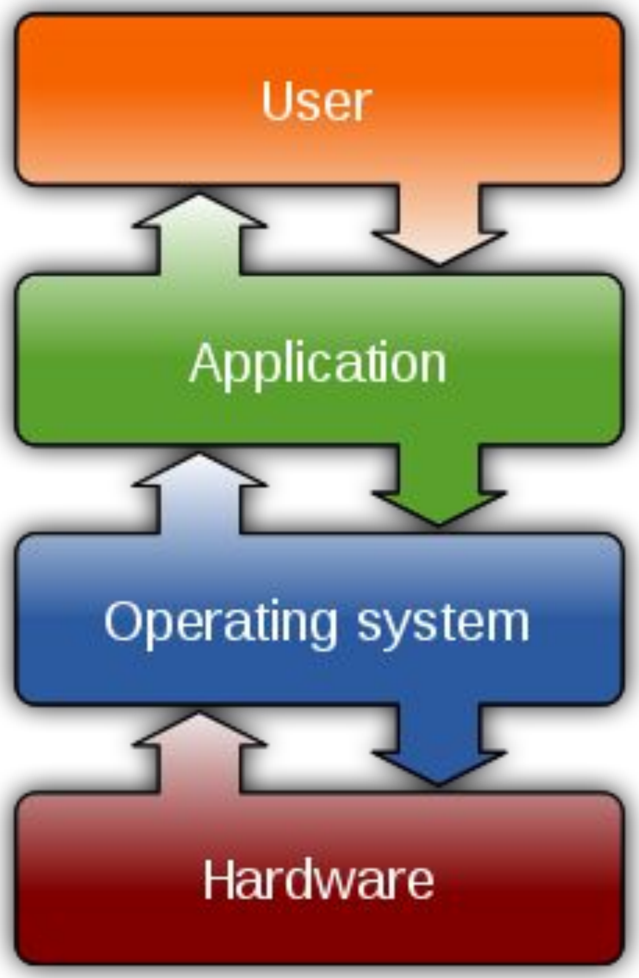
- Ransomware
- Adware
- Trojan
- Keyloggers

Operating System

Operating systems are found on many devices that contain a computer, embedded systems and super computers

Operating System

system software that manages computer hardware, software resources, and provides common services for computer programs. Time-sharing operating systems schedule tasks for efficient use of the system and may also include accounting software for cost allocation of processor time, mass storage, printing, and other resources.



User

Application

Operating system

Hardware

Operating System

For hardware functions such as input and output and memory allocation, the operating system acts as an intermediary between programs and the computer hardware, although the application code is usually executed directly by the hardware and frequently makes system calls to an OS function or is interrupted by it

Operating System

Types of OS : Single/Multi Tasking, Single/Multi User, Distributed, Templated, Embedded, Real Time and Library

Operating System

- Linux : Ubuntu, Red Hat, Fedora etc
- Windows
- Macintosh
- Raspbian

Writing Softwares

Software = Domain, Environment, Logic

Writing Softwares

Software can be built for variety of domains. For developing softwares we use programming languages. Softwares are built and executed in specific environments using dedicated runtimes.

Writing Softwares

Mostly softwares are Desktop Applications, Javascript, Server Softwares, Plugins, Embedded Softwares, Web Applications etc

Firmware

software that provides the low-level control for a hardware.

- Applications
- Examples

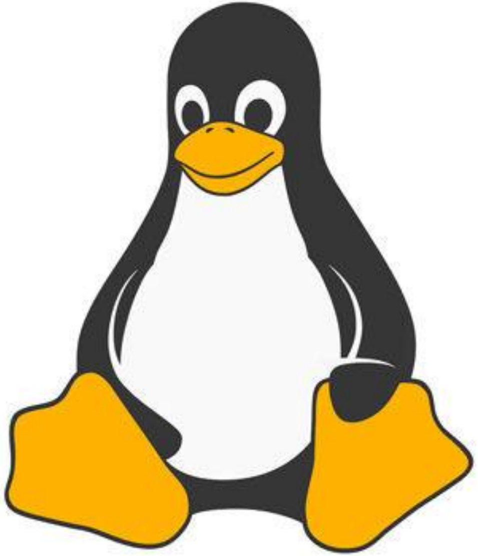
Applications

- Personal Computers, BIOS is as important as the operating system in a working computer
- Home/Consumer Products, firmware updates to update GUI, performance etc
- Automobiles, On-board computer and various sensors to detect mechanical problems

Examples

- Timing and control systems for washing machines
- Controlling sound and video attributes
- Hard disk drive, solid-state drive or optical disc drive firmware
- Video BIOS of a graphics card
- LibreCMC – a 100% free software router distribution based on the Linux-libre kernel

Linux Introduction



Linux™

Linux

most prominent examples of free and open-source software

- Basics Of Linux
- Linux Distributions
- Working with Ubuntu
- Linux Commands

Basics of Linux

Linux is one of the most prominent examples of free and open-source software collaboration.

Basics of Linux

Linux is a family of open-source Unix-like operating systems based on the Linux kernel, an operating system kernel first released on September 17, 1991, by Linus Torvalds. Linux is typically packaged in a Linux distribution.

Basics of Linux

Linux was originally developed for personal computers based on the Intel x86 architecture, but has since been ported to more platforms than any other operating system. Because of the dominance of the Linux-based Android on smartphones, Linux also has the largest installed base of all general-purpose operating systems.

Usages of Linux

Linux also runs on embedded systems, i.e. devices whose operating system is typically built into the firmware and is highly tailored to the system.

Cloud is Linux

90% of all cloud infrastructure is powered by Linux including supercomputers and cloud providers. 74% of smartphones in the world are Linux-based.

Linux Distributions

Linux distribution comprises a Linux kernel, GNU tools and libraries

Linux Distributions

A Linux distribution is an operating system made from a software collection that is based upon the Linux kernel and, often, a package management system.

Linux Distributions

Linux users usually obtain their operating system by downloading one of the Linux distributions, which are available for a wide variety of systems ranging from embedded devices (for example, OpenWrt) and personal computers (for example, Linux Mint) to powerful supercomputers (for example, Rocks Cluster Distribution).

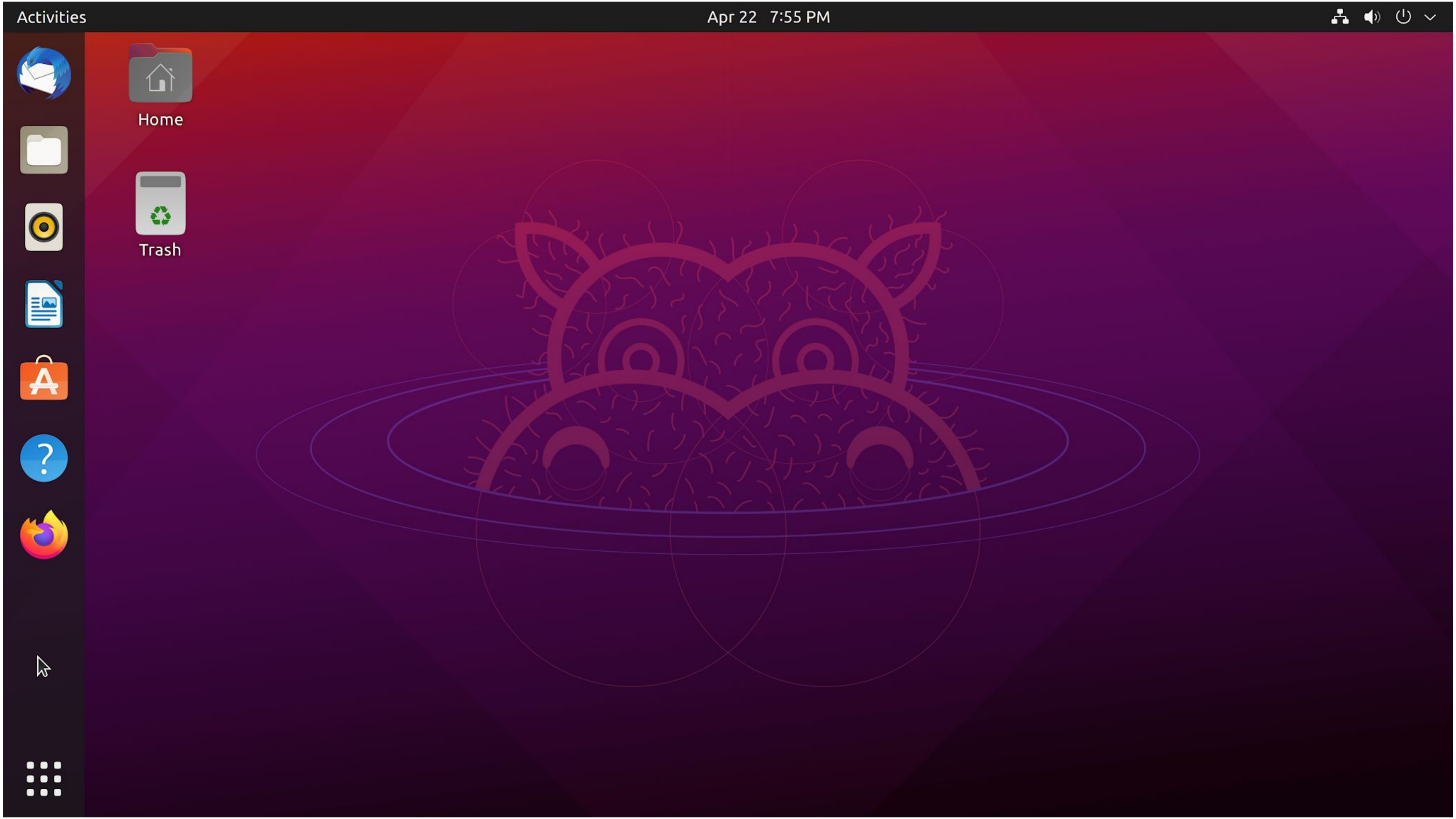
redhat
MEPIS
turbolinux
LUNAR
EvilEntity
debian
Vine Linux
cAos/CentOS
MiniKazit
UTUTO
archlinux
m0n0wall
jamd
Knoppix STD
gentoo linux
DeLi Linux
Hiweed
amlug
slackware linux
yellow dog linux
Fedora
LPG
PLD
SLAX
COREL LINUX
Progeny
GeeXBox
BIGLINUX
FREEDUC
Lycoris
EnGarde
Mandrakelinux
BeatrIX
Linspire
SuSE
YOPER
BearOps
ASPLINUX
kalango
Slackintosh
Frugalware
Foresight
Mint
PCLinuxOS
PCLinuxOS
Haydar Linux
sabayon
ubuntu
JULEX
blag

Working with Ubuntu

Ubuntu is named after the Nguni philosophy of ubuntu, "humanity to others"

Ubuntu OS

Ubuntu is a Linux distribution based on Debian and composed mostly of free and open-source software. Ubuntu is officially released in three editions: Desktop, Server, and Core for Internet of things devices and robots.

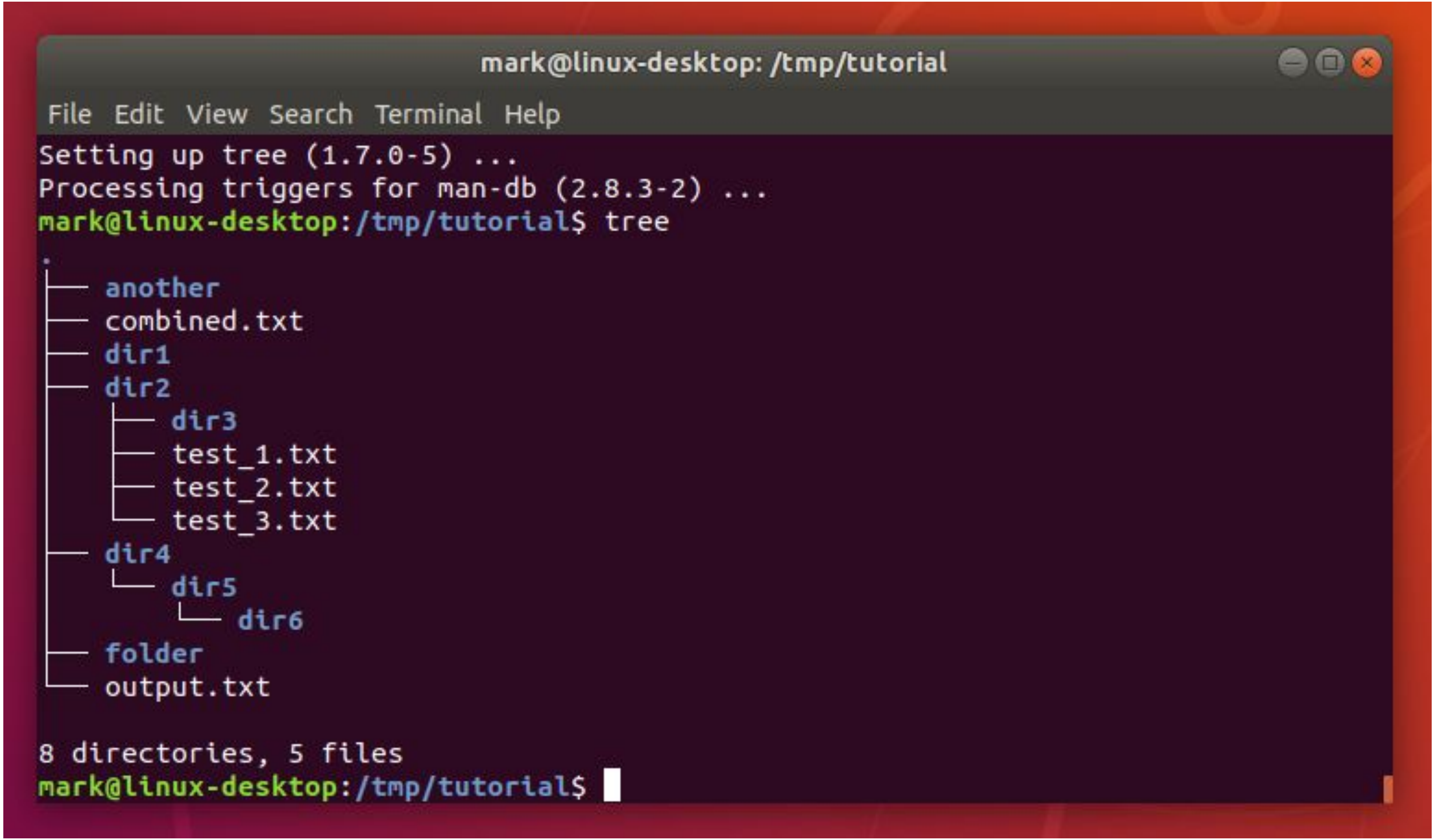

Activities
Apr 22 7:55 PM
Home
Trash

Ubuntu OS

All the editions can run on the computer alone, or in a virtual machine. Ubuntu is a popular operating system for cloud computing, with support for OpenStack. Ubuntu's default desktop has been GNOME, since version 17.10.

Terminal

As any Linux user knows, it’s the command line terminal where the magic happens. It’s perfect for file management, development, remote administration and a thousand other tasks.



mark@linux-desktop: /tmp/tutorial
File Edit View Search Terminal Help

```
Setting up tree (1.7.0-5) ...
Processing triggers for man-db (2.8.3-2) ...
mark@linux-desktop:/tmp/tutorial$ tree
.
├── another
├── combined.txt
├── dir1
├── dir2
│   ├── dir3
│   ├── test_1.txt
│   ├── test_2.txt
│   └── test_3.txt
├── dir4
│   └── dir5
│       └── dir6
├── folder
└── output.txt

8 directories, 5 files
mark@linux-desktop:/tmp/tutorial$ 
```

Linux Commands

The Linux command line is a text interface to your computer.

Command

The Linux command is a utility of the Linux operating system. All basic and advanced tasks can be done by executing commands. The commands are executed on the Linux terminal

Command

Lets practice few commands

Basics of Programming

Programming

Place where a thread can be a string

- What is programming
- Compiler and Interpreter
- Basics of Coding
- Choose Language

What is a Programming ?

activity of giving instructions to computer for performing task

Computer Programming

Computer programming is the process of designing and building an executable computer program to accomplish a specific computing result or to perform a specific task.

Coding

Programming involves tasks such as: analysis, generating algorithms, profiling algorithms' accuracy and resource consumption, and the implementation of algorithms in a chosen programming language (commonly referred to as coding).

Processing

The source code of a program is written in one or more languages that are intelligible to programmers, rather than machine code, which is directly executed by the central processing unit

Problem Solving

The purpose of programming is to find a sequence of instructions that will automate the performance of a task (which can be as complex as an operating system) on a computer, often for solving a given problem.

Problem Solving

Proficient programming thus often requires expertise in several different subjects, including knowledge of the application domain, specialized algorithms, and formal logic.

Tasks Involved

testing, debugging, source code maintenance, implementation of build systems, and management of derived artifacts, such as the machine code of computer programs.

Software Engineering

These might be considered part of the programming process, but often the term Software Development is used for this larger process. Software Engineering combines engineering techniques with Software Development practices.

Compiler and Interpreter

two main means by which programming languages are implemented

Compiler

a compiler is a computer program that translates computer code written in one programming language (the source language) into another language (the target language).

Compiler

primarily used for programs that translate source code from a high-level programming language to a lower level language (e.g. assembly language, object code, or machine code) to create an executable program

Cross Compiler

capable of creating executable code for a platform other than the one on which the compiler is running. For example, a compiler that runs on a PC but generates code that runs on Android/iOS smartphone is a cross compiler.

Bootstrap Compiler

is written in the language that it intends to compile. An initial core version of the compiler is generated in a different language; successive expanded versions of the compiler are developed using this minimal subset of the language.

Decompiler, Transpiler, Rewriter

A program that translates from a low-level language to a higher level one is a decompiler. A program that translates between high-level languages is usually called a source-to-source compiler or transpiler. A language rewriter is usually a program that translates the form of expressions without a change of language.

Compiler Operations

A compiler is likely to perform some or all of the following operations, often called phases: preprocessing, lexical analysis, parsing, semantic analysis (syntax-directed translation), conversion of input programs to an intermediate representation, code optimization and code generation.

Interpreter

is a computer program that directly executes instructions written in a programming or scripting language, without requiring them previously to have been compiled into a machine language program.

Strategies

- Parse the source code and perform its behavior directly;
- Translate source code into efficient intermediate representation or object code and immediately execute that;
- Explicitly execute stored precompiled code made by a compiler which is part of the interpreter system.

Examples

- Python uses second type
- Java/Swift/Kotlin uses mix of second and third

Basics of Coding

the process of creating and maintaining the source code

Algorithm

In mathematics and computer science, an algorithm is a finite sequence of well-defined, computer-implementable instructions, typically to solve a class of specific problems or to perform a computation.

Algorithm

Algorithms are always unambiguous and are used as specifications for performing calculations, data processing, automated reasoning, and other tasks.

Pseudocode

In computer science, pseudocode is a plain language description of the steps in an algorithm or another system. Pseudocode often uses structural conventions of a normal programming language, but is intended for human reading rather than machine reading.

Pseudocode

It typically omits details that are essential for machine understanding of the algorithm, such as variable declarations and language-specific code.

Variables

a variable or scalar is an abstract storage location paired with an associated symbolic name, which contains some known or unknown quantity of information referred to as a value; a variable is a container for a particular type of data (like integer, float, String and etc

Data Type

data type or simply type is an attribute of data which tells the compiler or interpreter how the programmer intends to use the data. Most programming languages support basic data types of integer numbers (of varying sizes), floating-point numbers (which approximate real numbers), characters and Booleans.

Operators

operators are constructs defined within programming languages which behave generally like functions, but which differ syntactically or semantically.

Operators

arithmetic (e.g. addition with +), comparison (e.g. "greater than" with >), and logical operations (e.g. AND, also written && in some languages). Assignment (usually = or :=), field access in a record or object (usually .), and the scope resolution operator (often :: or .)

Conditional Statements

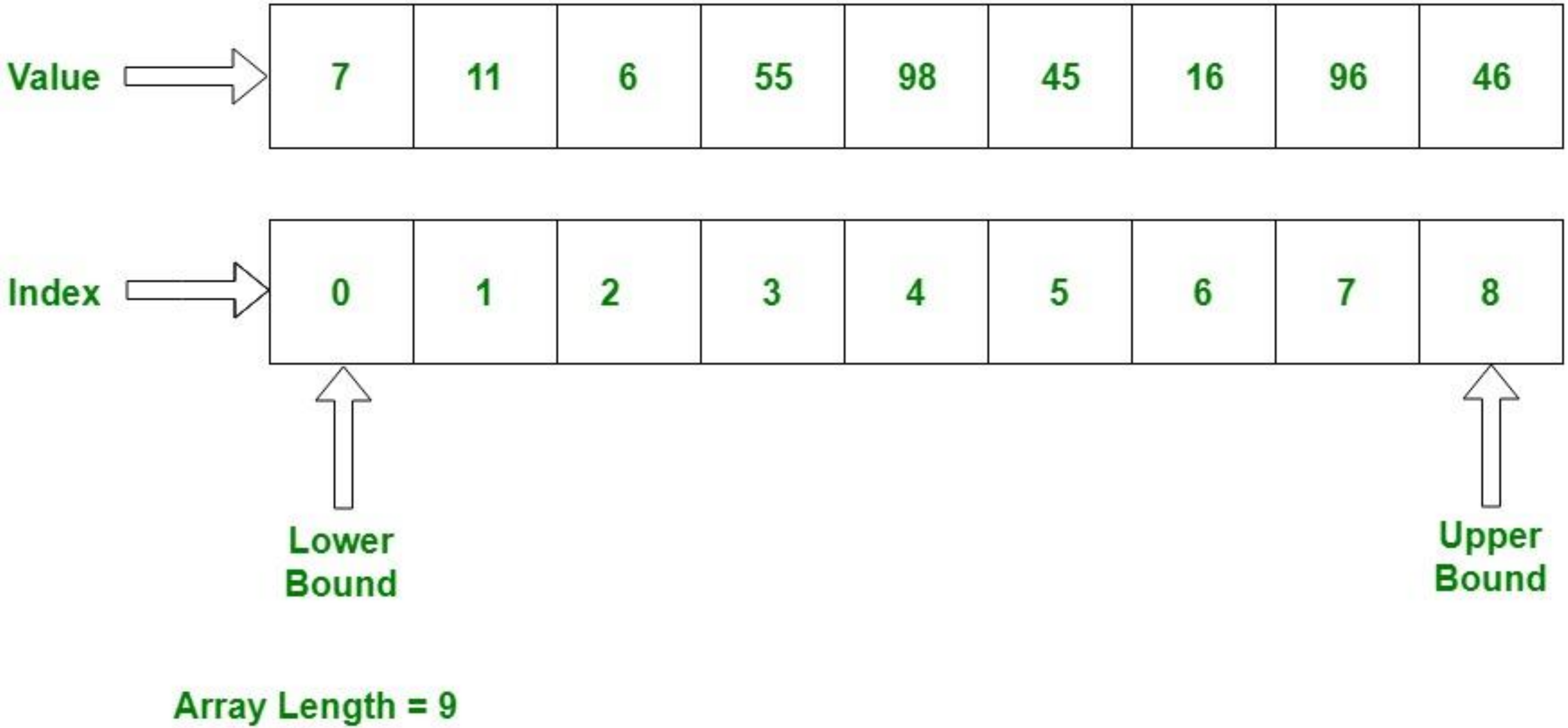
conditional statements are programming language commands for handling decisions. Specifically, it perform different computations or actions depending on whether a programmer-defined boolean condition evaluates to true or false.

Conditional Statements

```
If stock=0 Then
    message= order new stock
Else
    message= there is stock
End If
```

Arrays

is a data structure consisting of a collection of elements (values or variables), each identified by at least one array index or key. An array is stored such that the position of each element can be computed from its index tuple by a mathematical formula



Value	7	11	6	55	98	45	16	96	46
Index	0	1	2	3	4	5	6	7	8

Array Length = 9

Control Flow Statement

control flow is the order in which individual statements, instructions or function calls of an imperative program are executed or evaluated. The emphasis on explicit control flow distinguishes an imperative programming language from a declarative programming language.

Control Flow Statement

a control flow statement results in a choice being made as to which of two or more paths to follow. Interrupts and signals are low-level mechanisms that can alter the flow of control in a way similar to a subroutine, but usually occur as a response to some external event

Loops

A loop is a sequence of statements which is specified once but which may be carried out several times in succession. The code "inside" the loop is obeyed a specified number of times, or once for each of a collection of items, or until some condition is met, or indefinitely.

Loops

Count-controlled loops

```
for ( I=1; I<=N; ++I ) {
}
```

Loops

Condition-controlled loops

```
Do {


  } while (test);
```

Loops

Collection-controlled loops

```
 foreach (string s in collection) {
}
```

Function

is a sequence of program instructions that performs a specific task, packaged as a unit. This unit can then be used in programs wherever that particular task should be performed.

Function

```
def simple_function():
    print('Hello world!')
    print('Wikipedia')
simple_function()
```

Recursion

is a method of solving a problem where the solution depends on solutions to smaller instances of the same problem. Such problems can generally be solved by iteration, but this needs to identify and index the smaller instances at programming time.

Recursion

```
int fac1(int n) {
  if (n <= 0)  return 1;
  else  return fac1(n-1)*n;
}
```

Logic Error

is a bug in a program that causes it to operate incorrectly, but not to terminate abnormally (or crash). A logic error produces unintended or undesired output or other behaviour, although it may not immediately be recognized as such

Syntax Error

Error in the syntax of a sequence of characters or tokens that is intended to be written in compile-time. A program will not compile until all syntax errors are corrected. For interpreted languages, however, a syntax error may be detected during program execution,

Exception Handling

process of responding to the occurrence of exceptions – anomalous or exceptional conditions requiring special processing – during the execution of a program. an exception breaks the normal flow of execution and executes a pre-registered exception handler;

Classes

a class is an extensible program-code-template for creating objects, providing initial values for state (member variables) and implementations of behavior (member functions or methods).

Objects

In the object-oriented programming, object can be a combination of variables, functions, and data structures; In Non OOP, an object can be a variable, a data structure, a function, or a method, and as such, is a value in memory referenced by an identifier.

Data Structures

Is a data organization, management, and storage format that enables efficient access and modification. data structure is a collection of data values, the relationships among them, and the functions or operations that can be applied to the data

Data Structures

Data structures provide a means to manage large amounts of data efficiently for uses such as large databases and internet indexing services.

Data Structures

Data structures can be used to organize the storage and retrieval of information stored in both main memory and secondary memory.

Memory Management

Memory management is a form of resource management applied to computer memory. The essential requirement of memory management is to provide ways to dynamically allocate portions of memory to programs at their request, and free it for reuse when no longer needed.

Garbage Collection

is a strategy for automatically detecting memory allocated to objects that are no longer usable in a program, and returning that allocated memory to a pool of free memory locations.

Object Oriented Programming

is a programming paradigm based on the concept of "objects", which can contain data and code: data in the form of fields (often known as attributes or properties), and code, in the form of procedures (often known as methods).

Object Oriented Programming

OOP languages are diverse, but the most popular ones are class-based, meaning that objects are instances of classes,

Functional Programming

is a programming paradigm where programs are constructed by applying and composing functions.

Functional Programming

functions are treated as first-class citizens, meaning that they can be bound to names (including local identifiers), passed as arguments, and returned from other functions, just as any other data type can.

Async Programing

refers to the occurrence of events independent of the main program flow and ways to deal with such events. These may be "outside" events such as the arrival of signals, or actions instigated by a program that take place concurrently with program execution, without the program blocking to wait for results.

Libraries

a library is a collection of non-volatile resources used by computer programs, often for software development. These may include configuration data, documentation, help data, message templates, pre-written code and subroutines, classes, values or type specifications.

Framework

Is an abstraction in which software, providing generic functionality, can be selectively changed by additional user-written code, thus providing application-specific software.

API

An application programming interface (API) is a connection between computers or between computer programs. It is a type of software interface, offering a service to other pieces of software

Thanks,

Let's Code Now

Credits

- https://en.wikipedia.org/