 # OS Subject:
- there are four steps to learn an OS subject:
1. step1: "end user" -- linux commands - user commands

2. step2: "admin user" -- admin commands, shell script programming & installations.

+ Responsibilites of System Administrator:
- installing and configuring software, hardware and networks.
- monitoring system performance and troubleshooting issues.
- ensuring security and efficiency of IT infrastructure.
3. step3: "programmer user" -- system call programming
4. step4: "design/internals" -- to learn os internals/architecture

---

Q. What is Computer?
- Computer is a **machine/hardware/digital device** mainly contains: **Processor/CPU, Memory Devices, I/O Devices** etc....
- Basic Functions of Computer:
1. data storage
2. data processing
3. data movement
4. control
- by using computer machine various/different tasks can be performed efficiently and accurately.

Q. What is a Program?
- set of instructions given to the machine to do specific task.

- there are three types of programs:
1. "user programs": programmer user defined programs.
e.g. main.c, main.cpp, hello.java etc...

2. "application programs": e.g. notepad, google chrome, mozilla firefox, wordpad, eclipse etc...

3. "system programs": e.g. device driver, loader, scheduler, interrupt handler program etc...

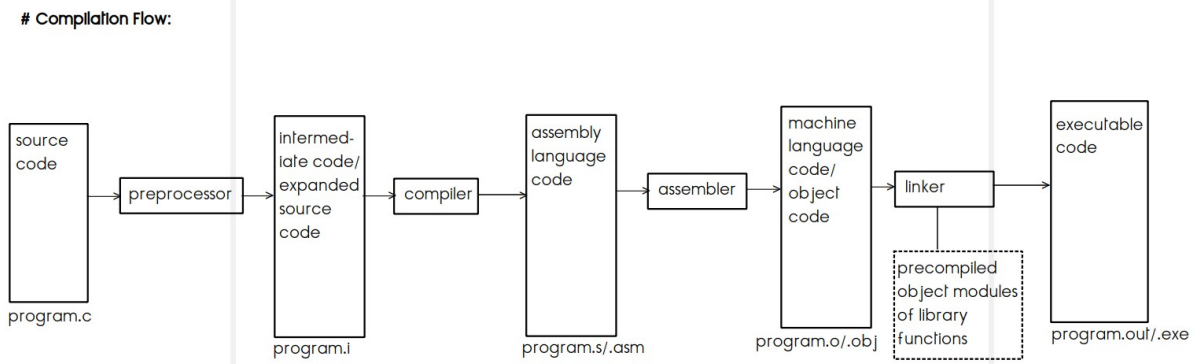
+ "source code": program written in any programming language is called as a source code.
e.g. c source code, c++ source code, java source code etc....
- to write a source code/program we required a tool/program reffered as an editor program also reffered as a **source code editor.**
e.g. notepad, gedit, vi editor, eclipse source code editor program, vs code editor etc...

- **Eclipse is an IDE: Integrated Developement Environment**, written mostly in Java and its primary use is for developing java applications, but it may be used to develope applications in other programming languages via plug-ins, including Ada, ABAP, C, C++, C#, COBOL, D, Fortran, JavaScript, Perl, PHP, Python, R, Ruby etc...

- IDE is an **"application software"** which is a collection of tools/programs like an **editor(source code editor), compiler, linker, assembler, debugger** etc... required for faster software developement.
- Any IDE normally consists of a **source code editor, build automation tools, and a debugger.**
- Most of the IDEs have intelligent code completion.
- Some IDEs such as **netbeans** and **eclipse**, contains a compiler, interpreter, or both.
- IDE is in constrast with vi editor, gcc, ld etc....

e.g. eclipse, netbeans, code blocks, visual studio, borland turbo c, turbo c++, android studio, anjuta, Dev C++, CodeLite, QT Creator etc...

Compilation Flow:



+ "preprocessor": it is an application program gets executes before compilation and performs two tasks:
1. removes all comments from the source code, and
2. executes all pre-processor directives like #include, #define, #ifndef, #ifdef, #elif, #endif, #pragma etc... conditional compilation preprocessor directive i.e. header guards.

e.g. **"cpp" – c preprocessor, M4 macro processor in a UNIX like systems.**

– the output of preprocessor is an **intermediate code,** as this file gets created with the combination of header file and source file, size of this file gets increases and hence it is also called as an **expanded source code.**

– command to create an intermediate code/file from source code/file:

$gcc –E –o program.i program.c ==> program.i

+ "compiler": compiler is an application program which converts high level programming language code (i.e. human understandable language code) into the low level programming language code (i.e. machine understandable language code).

– output of a compiler is an **"assembly language code".**

- e.g. **GCC: GNU Compiler's Collection**, originally named as **GNU C Compiler.**
- GNU: GNU's Not UNIX/GNU is Not UNIX which is a recursive acronym.
- GNU: it is an Open Source Project by OSF (Open Source Foundation).
- Linux is also a GNU Project.
- Now a days GCC can be used for compilation of C++, FORTRAN, Objective C, Objective C++, Ada etc... programming languages.
- we need to use front ends of "gcc".
- Borland Turbo C, Turbo C++, Microsoft Visual C, etc....
- Compiler does **tokanization, syntax checking, code analysis, code optimization, parsing** etc...

"Compiler's Construction" -- By Aho, Ullman -- from AT&T Bells Labs.
- command to create an assembly language code from the source code:
$gcc -S program.c ==> program.s/program.asm

+ assembler: it is an application program which converts assembly language code into the machine language code i.e. **object code.**
e.g. Linux: **"asm"**
Windows : Microsoft Assembler : **"masm"**
Turbo Assembler : **"tasm"**
etc...

- declarations of library functions exists in a **system header files** e.g. stdio.h, string.h, stdlib.h etc.. , whereas definitions of all library functions are exists in a "lib" folder in **"precompiled object module"** format.

+ linker: it is an application program which links object file(s) in a project with precompiled object modules of library functions and creates a "single" executable file.
e.g. **"ld"** -- link editor in Linux.

"build = compilation + linking"

compilation --> to create object code from source code.
linking --> linking of all object files with precompiled object modules by the linker and creates single executable file.

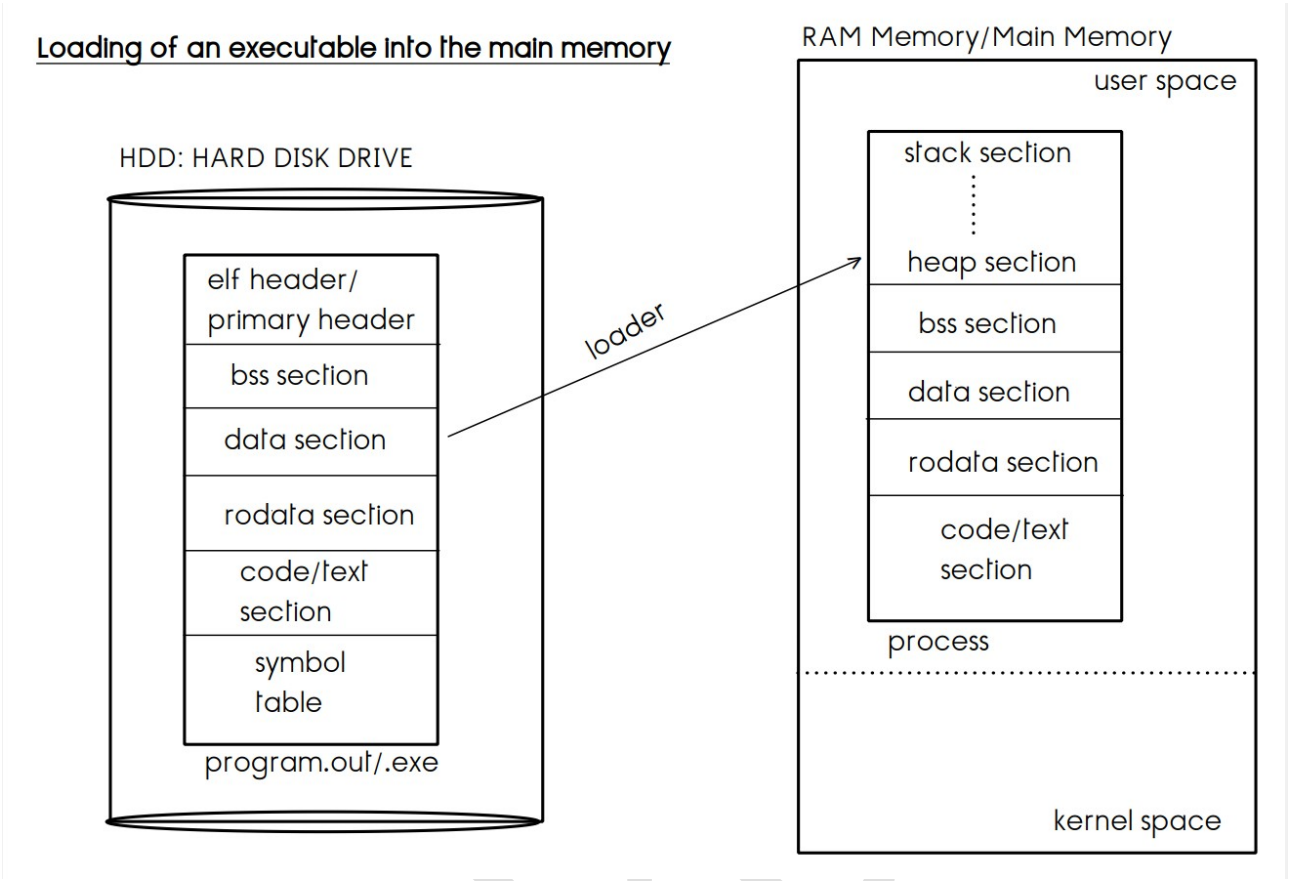
- command to create executable file from object file:
    $gcc -o program.out program.o ==> program.out

- command to execute a program:
    $./program.out

program.c ==> [preprocessor] ==> program.i ==> [compiler] ==> program.s/.asm ==> [assembler] ==> program.o/.obj ==> [linker] ==> program.out/.exe

- **"portability"**: c program written on one machine/ platform can be compiled and execute on any other machine/platform.

- In Linux file format of an executable file is **"ELF": Executable & Linkable Format (Formerly named as Extensible Linking Format)**, whereas in Windows file format of an executable file is **"PE": Portable Executable.**
- ELF is a common standard file format for **an executable files, object codes, shared libraries,** and **core dumps.**
- **core dump** is also called as **crash dump, memory dump, or system dump** consists of the recorded state of the working memory of a computer program at a specific time, generally when the program has crashed or otherwise terminated abnormally.
- executable file is a program only contains set of instructions with extra information.
- in Linux ELF is a **"sectioned binary"**, i.e. executable file in linux is divided into different sections, mainly it contans: **exe header/primary header, data section, bss i.e. block started by symbol section, rodata i.e. read only data section, code/text section, symbol table** etc....



1. "exe header/primary header": it contains info to starts an execution of a program(executable file), mainly it contains:
I. Magic Number: it is a constant number generated by the compiler which is file format specific.
- In ELF magic number starts with **"7f E L F"** -- ASCII values of letters E, L & F in its equivalent hexadecimal format.
- In PE -- magic number starts with ASCII values of letters M Z in its equivalent hexadecimal format, as **"Mark Zbikowski"** architect of windows operating system at Microsoft.
ii. addr of entry point function: it contains addr of _start routine in which addr of function can be kept from which execution to be started.
iii. info about remaining sections (metadata -- data about data)

2. "data section": it contains initialized global & static variables.

3. "bss section (bock started by symbol)": it contains uninitialized global & static variables.

4. "rodata section (read only data section)": it contains constants & string literals.
e.g.
const int num = 100; -> constants
char *str = "sunbeam infotech"; -> string literals

5. "code/text section": executable instructions

6. "symbol table": it contains info about symbols i.e. functions and their variables in tabular format.
etc...

+ Magic Number:
- Magic number are the first few bytes of a file which are unique to a particular file type. These unique bits are reffered as magic number, also sometimes reffered as **file signature.**
- These bits/bytes can be used by the system **to differentiate between and recognize different files without file extension.**

+ "loader": it is a system program i.e. part/module of an OS, which first verifies the file format of an executable file, if file format matches then only it checks the magic number, and if both the matches then only it loads program into the main memory, and we say program becomes a process/program has been executed.

- Command Name: readelf - is the command to display information about one or more ELF format object file/s an executable file in linux.
$readelf -h filename - display info about the elf header
$readelf -a fileame - display all info about an executable file.

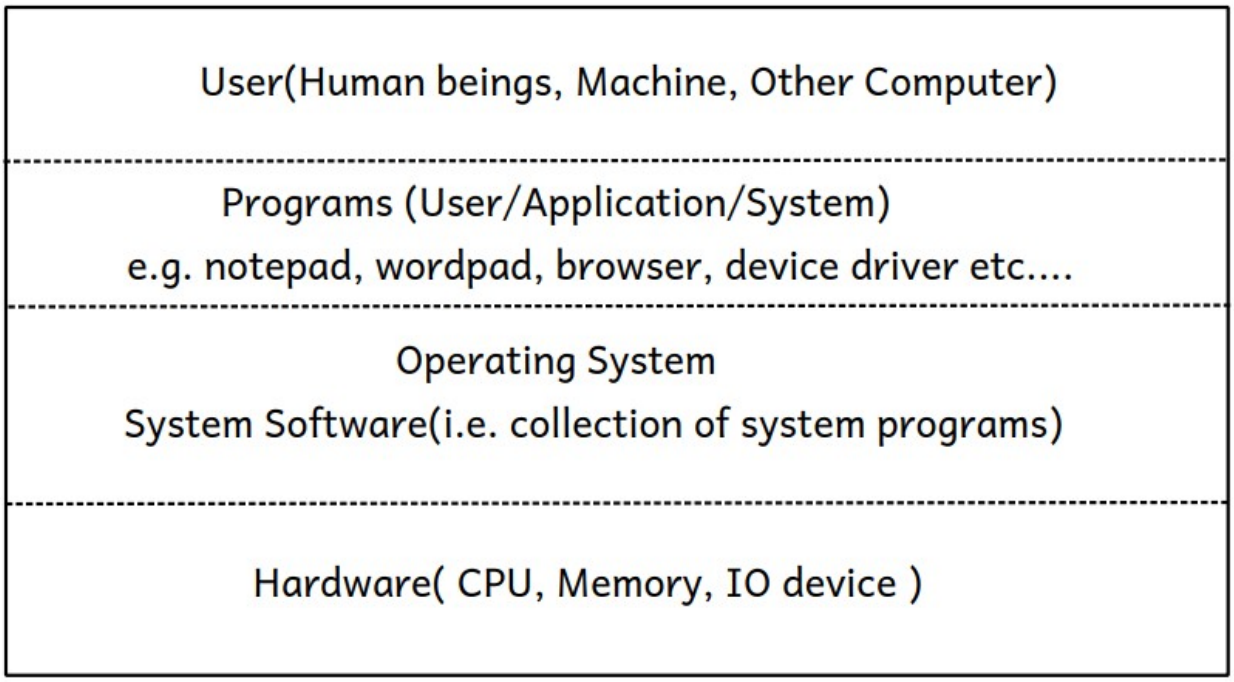
- Command Name: objdump - command to display information about object files, executable files as well core dump file.

- command to create all intermediate files while compilation:
$gcc -save-temps program.c
- command to link c program which do not having main()
$gcc -c program.c --> program.o
$gcc -o program.out -nostartfiles -esachin program.o --> program.out

- Command Name: size - The GNU size utility lists the section sizes and the total size for each of the object or archive files.

- Command Name: file - determine file type
- file command tests each argument in an attempt to classify it. There are three sets of tests, performed in this order: **filesystem tests**, **magic tests**, and **language tests**. The first test that succeeds causes the file type to be printed.

 # What is an OS?



- It is a **system software** (i.e. collection of system programs), which acts as an interface between user and computer hardware (i.e. processor, memory devices & io devices etc...).
- An OS also acts as an interface between programs and computer hardware.
- As an OS allocates required resources like CPU time, main memory, i/o device access to running programs, it is also called as **"resource allocator".**
- As an OS manages avaialable resources among all running programs, it is also called as a **"resource manager".**
- An OS provides environment/platform to all types of programs to complete their execution.
- An OS controls execution of all programs, as well as it controls all h/w devices connected to

the system, so it is also called as a **"control program"**.