IDC 101 Introduction to Computers

Instructor: Dr. Shashi B Pandit
(Department of Biological Sciences)

IDC 101: Introduction to computers

- Overview of scientific computing and the role of computers in solving scientific problems.
- Introduction to operating systems. Number representation in computers and roundoff error. Implications for numerical computing.
- Python programming. Basics and flowcharts. Data types and building blocks. Control statement. Functions. Arrays. Input/Output.
- Pseudo random numbers, applications of random sequences in scientific computing, simulating data and experiments, estimating errors in experiments using simulations.

- Data visualisation and analysis, statistical analysis, curve fitting using the least square fit approach. Series summation, numerical integration.
- Solutions of algebraic equations, iterative solutions. Recursion relations, logistics map. Brief overview of fractals resulting from simple maps. Bisection method. Newton-Raphson method.
- Ordinary differential equations, coupled equations, second order equations. Applications in evolution of population, reaction rates, mechanics.
- Systems of linear equations, matrices, row reduction, diagonalisation. Two dimensional arrays. Cellular automata.

IDC 101: Introduction to computers

- **Credit**: 2 (Lecture : 1 hour; Lab: 3 hours)

Lecture : 11 classes
Lab : 12 classes

- **Lab sessions**
  - Class will be split into groups each group is assigned a day for lab in a week.
  - Coding will be conducted on "**Google Colabaratory" or 'google colab**'
    (Google colab must be used using your Institute provided email id)

- **Evaluations**
  - Periodic evaluation will be conducted through Moodle or Google colab (Mid term, Quiz and End semester exam).
  - Lab sessions will be evaluated by co-Instructor managing a group.

IDC 101: Introduction to computers

- **Grading scheme (Any changes will be communicated in advance)**

  Quiz - 10%
  Mid term - 20%
  Continuous evaluation
  (including attendance) - 20%
  End Semester (Theory) - 25%
  Lab viva/tasks (Practical) - 25%

- **Instructors**
  - **Dr. Prasenjit Das (DPS) - Tuesday (9 am – 12)**
  - **Dr. Santhosh Kumar Pamula (DMS) – Monday (2 – 5 pm)**
  - **Dr. Yunus Ali Pulpadan (EES) - Tuesday (2 – 5 pm)**

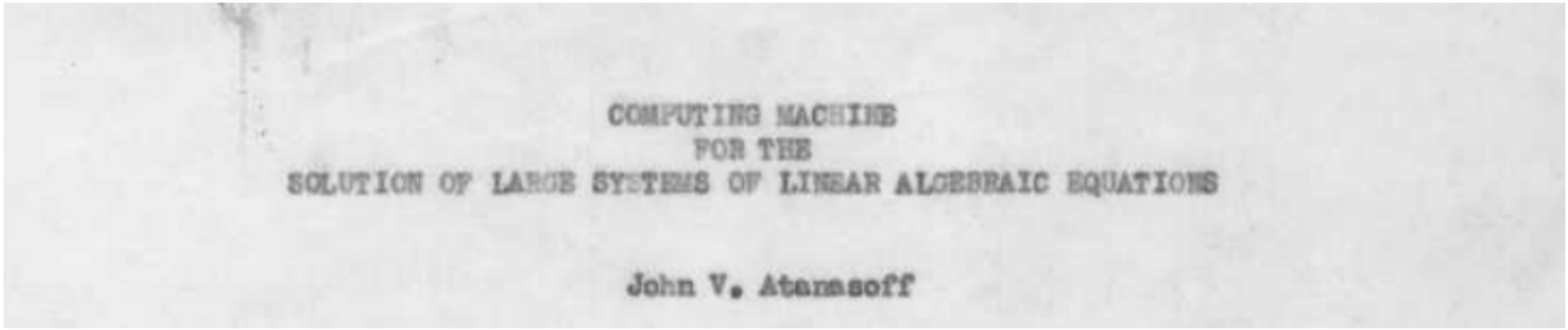
  - **Student/Postdoc tutors (7)**

IDC 101: Introduction to computers

Join *whatsapp* group

Overview of Scientific computing

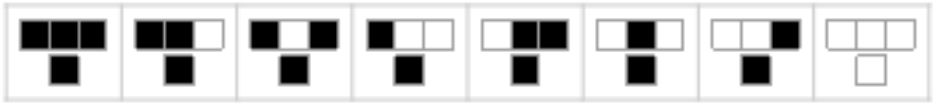
Scientific computing can broadly be defined as applications of computers to solve problems in domains of science (Physics, Mathematics, Biology, Chemistry, Geology, etc.).

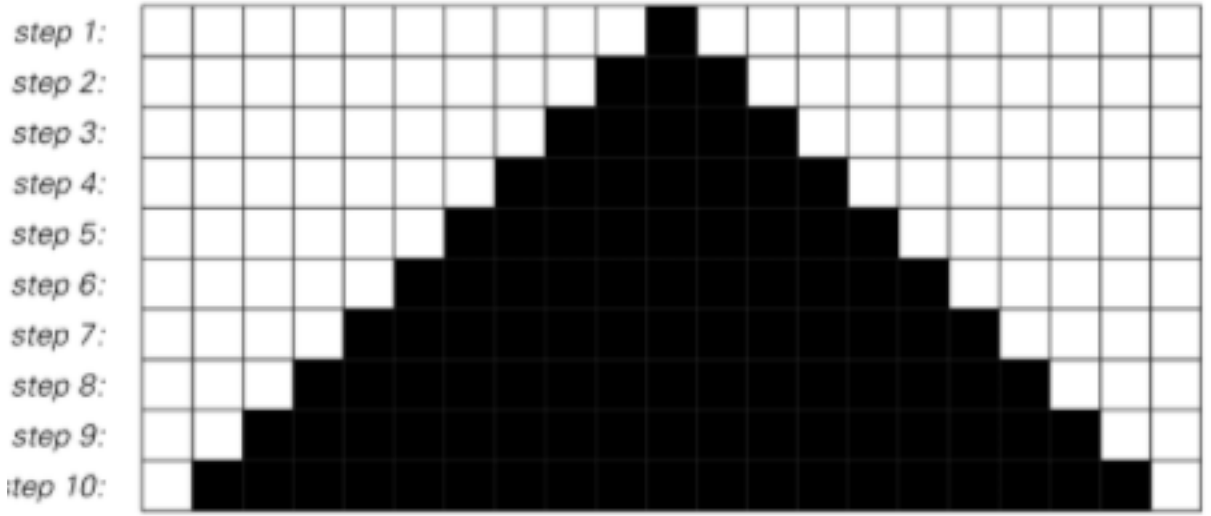


COMPUTING MACHINE
FOR THE
SOLUTION OF LARGE SYSTEMS OF LINEAR ALGEBRAIC EQUATIONS

John V. Atanasoff

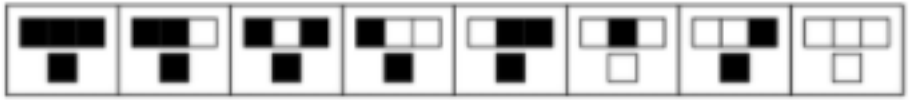
Overview of Scientific computing





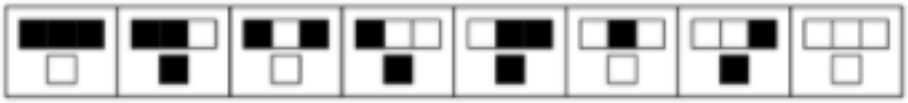
A new kind of science *Wolfram (2002)*

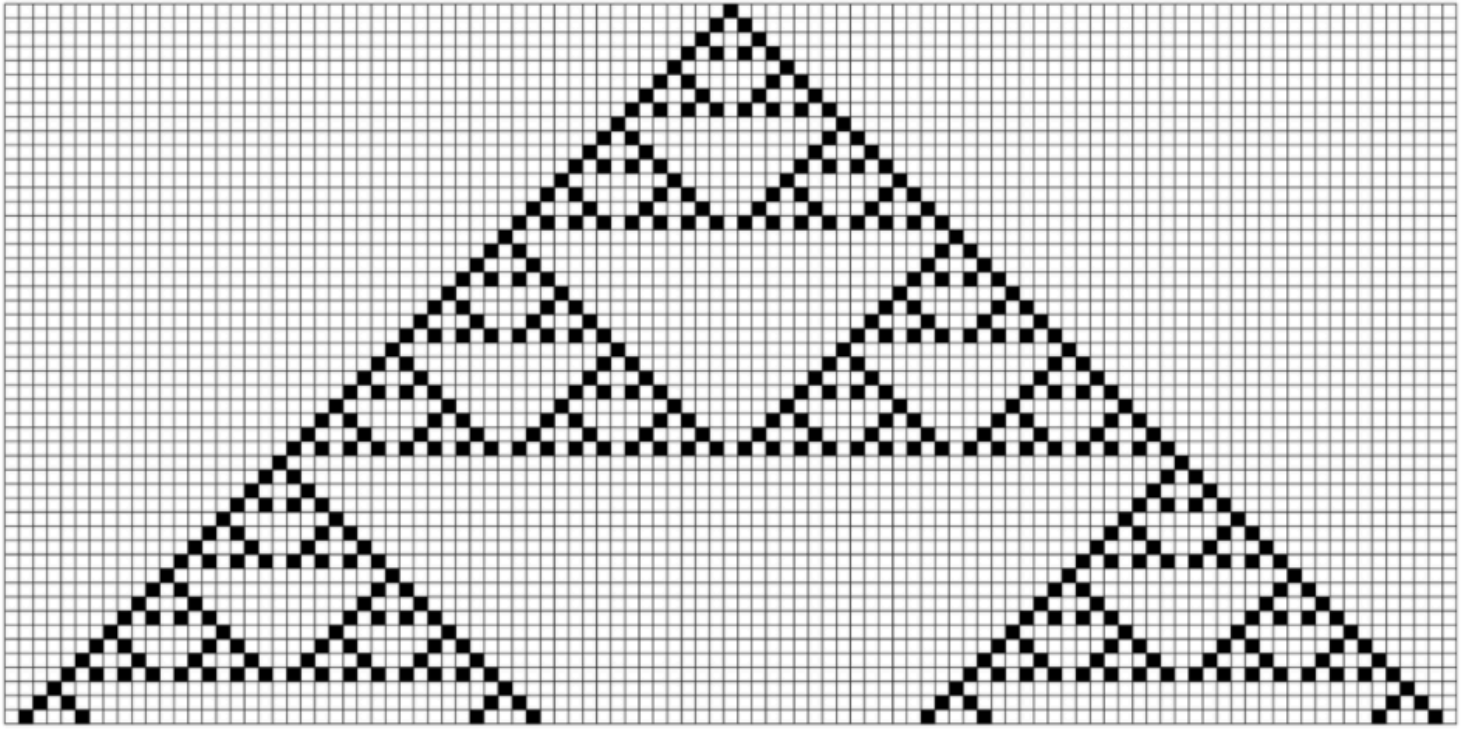
Overview of Scientific computing



A new kind of science *Wolfram (2002)*

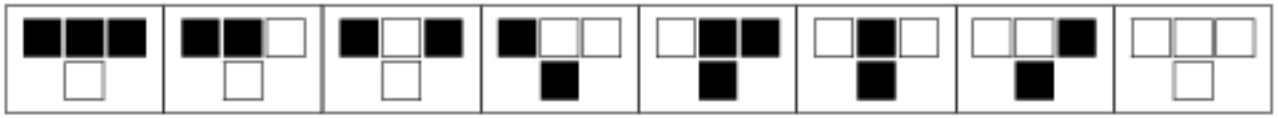
Overview of Scientific computing

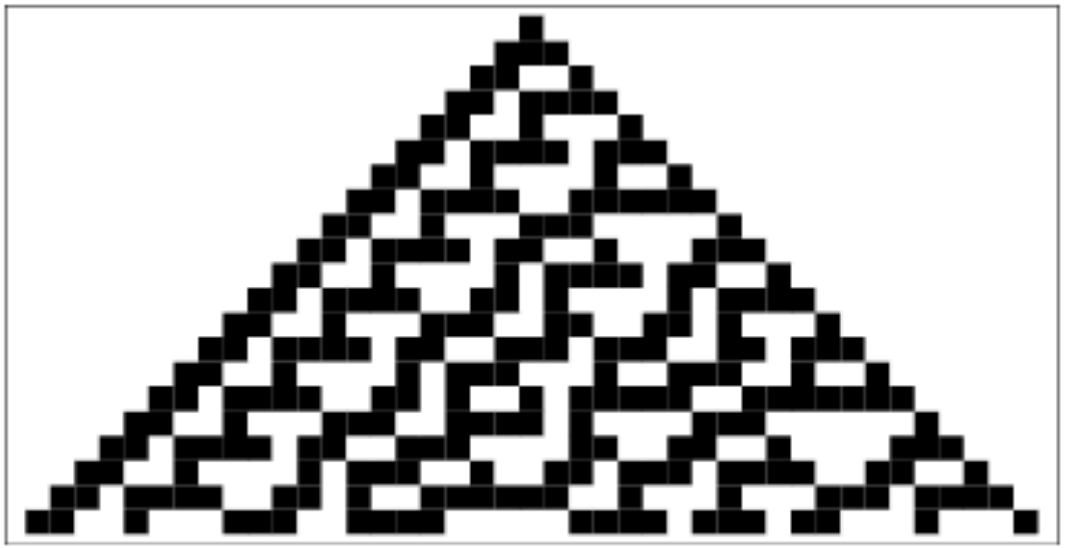




A new kind of science *Wolfram (2002)*

Overview of Scientific computing





Cellular automaton: It is a collection of colored cells on a grid of specified shape that evolves through a discrete number of steps following a rules based on neigbouring cell states.

A new kind of science *Wolfram (2002)*

UNIX operating system (OS)

Unix OS is a powerful multi-user and multi-tasking system developed in 1970's for mainframe, servers and workstations. It was developed at AT&T Bell labs by a team led by Ken Thompson and Dennis Ritchie. The system development has remained open-source (freely available to all) and it is community-based development (maintained by a group of developers).

Features: Portability, File security, communication (connection to a remote machine), command line. GUI etc.

There are multiple varieties of UNIX available such as Sun Solaris, Linux (many different types), MacOX etc.

The UNIX is primarily made of three layers/parts: kernel, Shell and programs

UNIX operating system (OS)

There are multiple varieties of UNIX available such as Sun Solaris, Linux (many different types), MacOX etc.

The UNIX is primarily made of three layers/parts: kernel, Shell and programs (Application program layer).

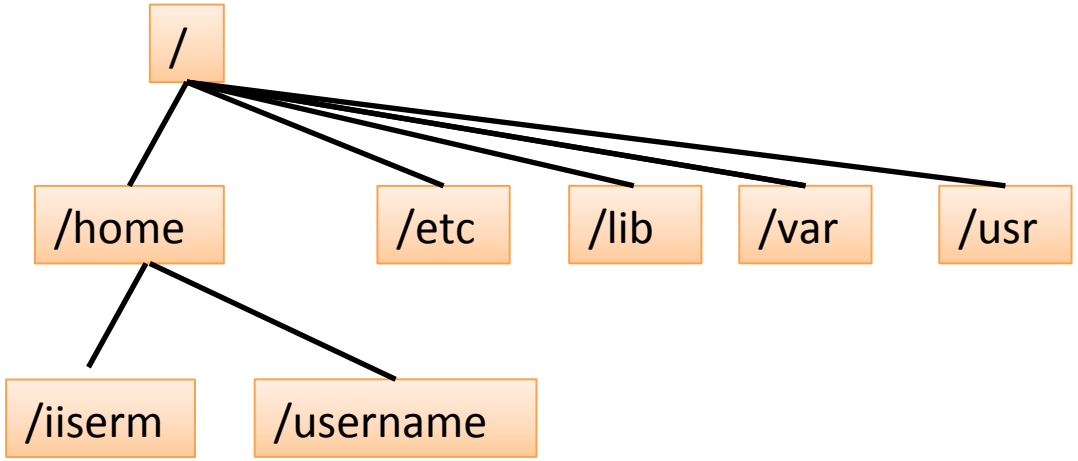
The **kernel** is the hub or core of UNIX operating system that maintains OS full-functionality through effectively communicating with hardware for various things such as to time/memory allocation to programs; device management (device drivers), file handling, communication to systems calls etc.

The **shell** is an interface between kernel and user. Essentially, it interprets the command submitted by user at the terminal and execute the process. It is Command Line Interpreter (CLI).

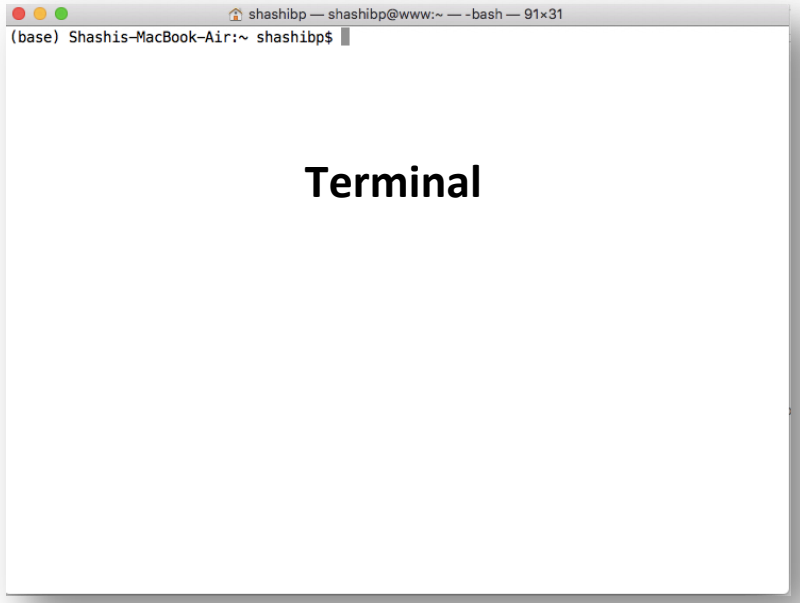
Bourne shell (sh)
C shell (csh or tcsh)
Korn shell (ksh)

UNIX OS data structure organization

In UNIX everything is a file or process. A file is commonly a collection of data/ command etc. Below is directory structure in UNIX file system…

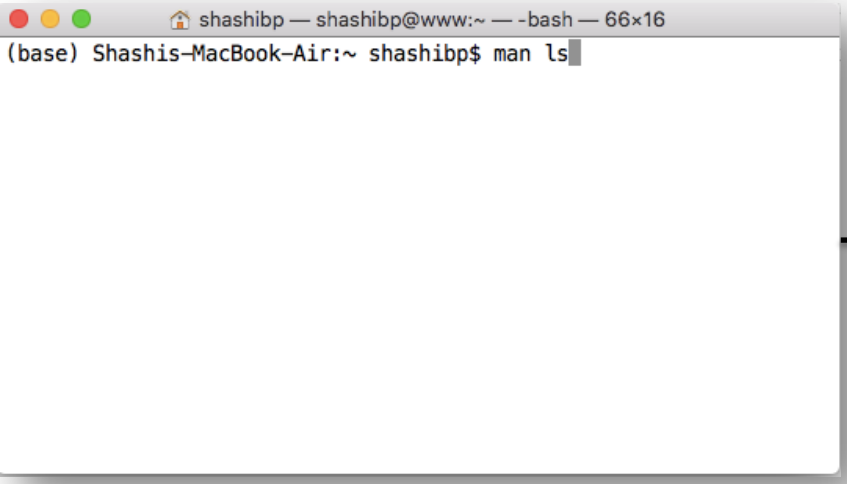


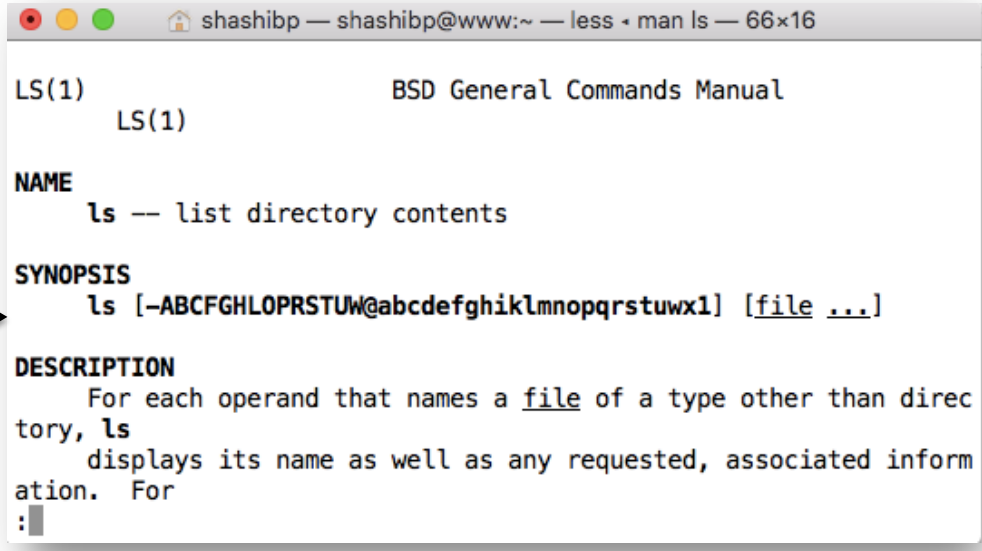
UNIX OS



There are many UNIX commands. To know more about any UNIX command, one can use manual by invoking '`man command name`'

To exit *man* page, one needs to press 'q/Q'





BASH shell

List of UNIX commands

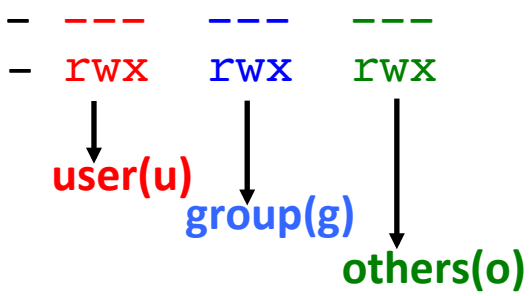
Command	Executes/ (meaning)
`pwd`	display the present working directory
`ls`	List files and directories
`ls -a`	List files and directories
`ls -1`	List files and directories with details
`mkdir "idc101"`	Make a directory named as idc101
`cd`	Change to home directory
`cd ..`	Change to one directory before the present directory
`cd ~/`	Change to home directory
`cp filea fileb`	Make a copy of filea named as fileb
`mv filea fileb`	Move filea into fileb (it is like renaming filea as fileb)
`rm filea`	Delete a filea
`wc filea`	Count number of lines/words/character in a filea

BASH shell

Command	Executes/ (meaning)
`head` *filea*	display first 10 lines of a file (filea)
`head -n12` *filea*	Display first 12 lines of a file
`tail` *filea*	Display last 10 lines of a file
`more` *filea*	Display file one page at a time (press enter to move to next page)
`less` *filea*	Display file one page at a time (press enter to move to next page) from last
`who`	Display the login details of a user
`touch` *filea*	Make an empty file filea
`echo "any text"`	Display the text within inverted commas on the screen
`echo "any text" >` *filea*	Write the text within inverted commas to a file named as *filea*

File permissions

```
[shashibp@www idc101]$ ls -al
total 16
drwxrwxr-x   2 shashibp shashibp   29 Nov  3 08:47 .
drwx------. 38 shashibp shashibp 8192 Nov  3 08:47 ..
-rw-rw-r--   1 shashibp shashibp   32 Nov  3 08:47 test.sh
```



r **R**ead permission (**4**)
w **W**rite permission (**2**)
x E**x**ecute permission (executable) (**1**)

Changing permission is by command **chmod**

chmod (u/g/o) +(adding permission) r/w/x (grant permissions for)
-(removing permission)
=(assigning permission)

chmod u+r FILENAME -> giving permission to user for reading a file
chmod u-r FILENAME -> giving permission to user for reading a file
chmod ugo+r FILENAME -> giving READ permission to all user,group,others

File permissions

r **R**ead permission (**4**)
w **W**rite permission (**2**)
x E**x**ecute permission (executable) (**1**)

Changing permission is by command **chmod using NUMBER system**

Add numbers to give a permission to user/group/others

For instance only **Read** permission **r--** **is 4**
For instance only **Read and write** permission **rw-** **is 6**
For instance only **Read, write and execute** permission **rwx** **is 7**
For instance only **Read and execute** permission **r-x** **is 5**
For instance only **NO** permission **---** **is 0**

Then,

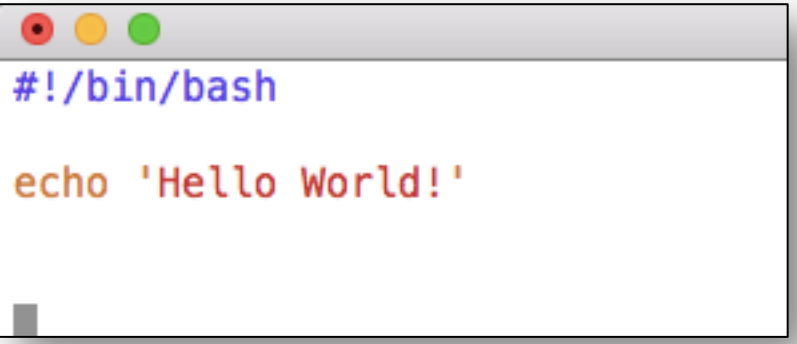
	ugo
chmod	**400**

Is giving only read permission to user ALONE and no permissions is for others

Shell scripting

Shell scripting

A shell script is simply a series of shell (bash) commands stitched (joined) together to achieve a task. Anything that will run on CLI, can be put on shell script.



```
#!/bin/bash

echo 'Hello World!'
```

#! It is referred to as *shebang,* which directs the shell to use program/interpreter to run the script below it.

Variable are defined starting with **$**
Special variable:
$0 - Program name
$1 - $9 – Variables input to the program