بسم الله الرحمن الرحیم

# مقدمات درس روشهای شبیه سازی در فیزیک (نظریه و محاسبات)

# Preliminaries for Advanced topics in computational Physics and Optimization

## Part 2

سیدمحمدصادق موحد

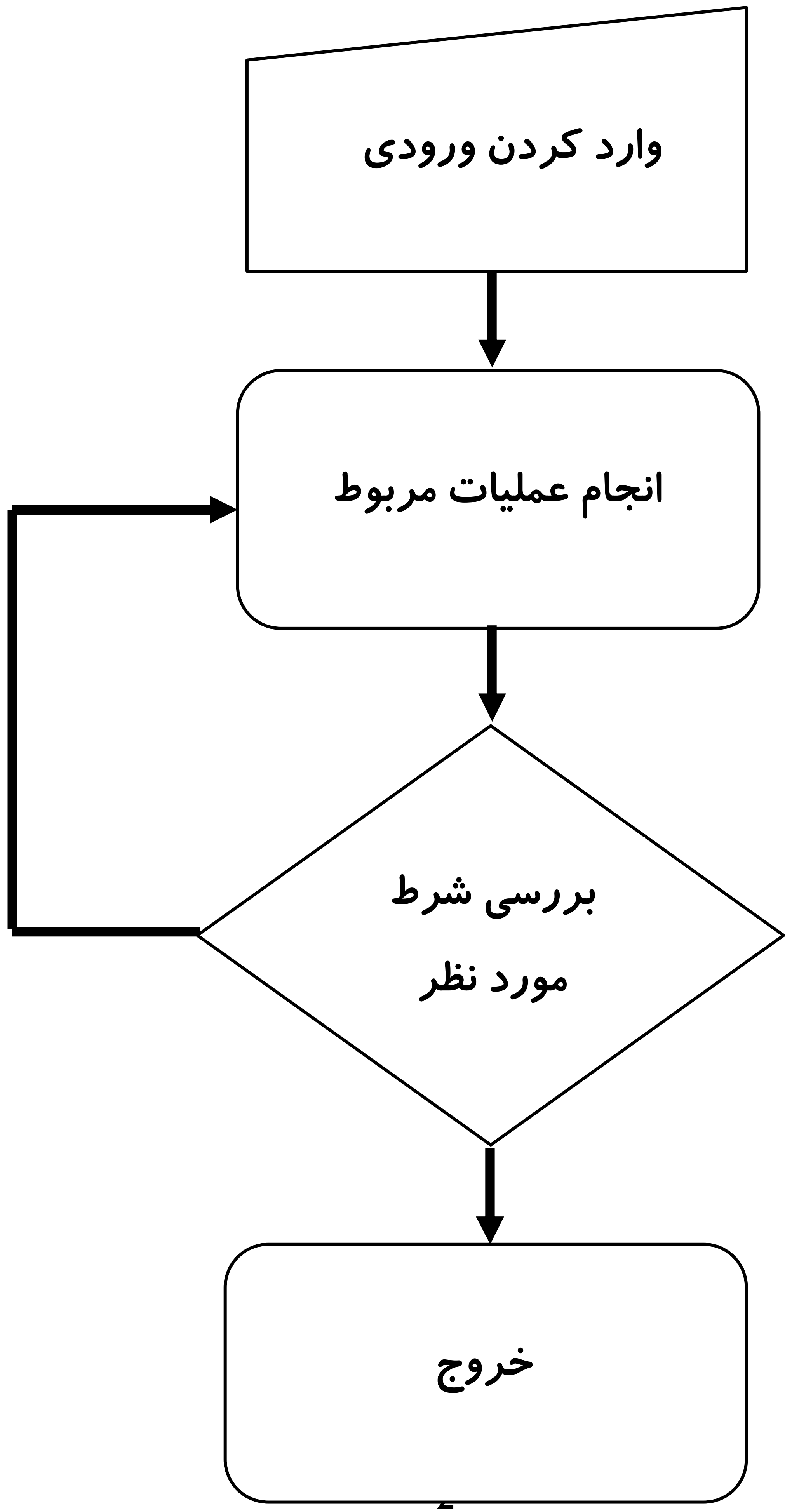
دانشکده فیزیک دانشگاه شهید بهشتی

گروه کیهانشناسی محاسباتی و آزمایشگاه ابن سینا

نیم سال اول، سال تحصیلی ۱۴۰۴-۱۴۰۳

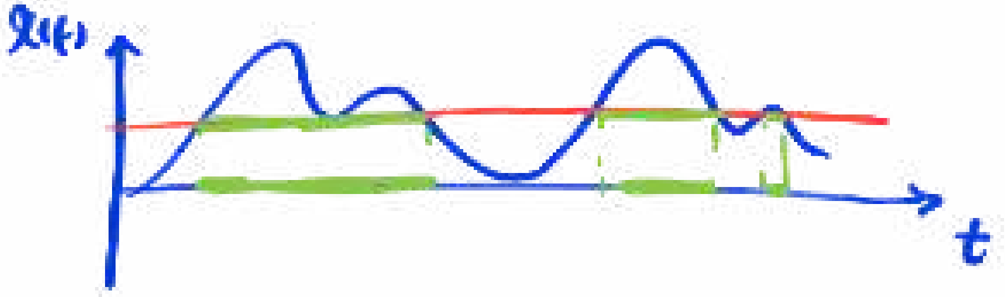
ccg.sbu.ac.ir

smovahed.ir


وارد کردن ورودی
انجام عملیات مربوط
بررسی شرط
مورد نظر
خروج

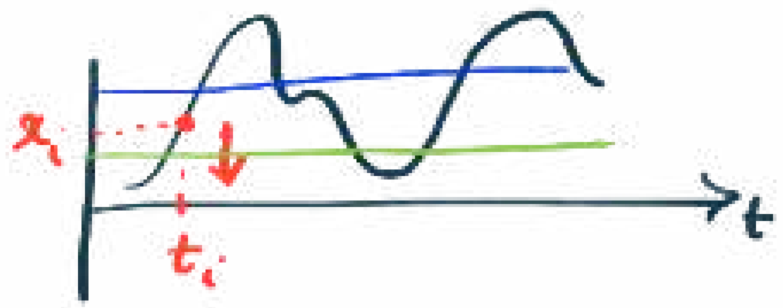
Import Data $\{x_i\}$ . $i=1, \dots N$

$$\Delta x = \frac{\mathrm{Max}(x)-\mathrm{Min}(x)}{M}$$



Minkowski $\theta$

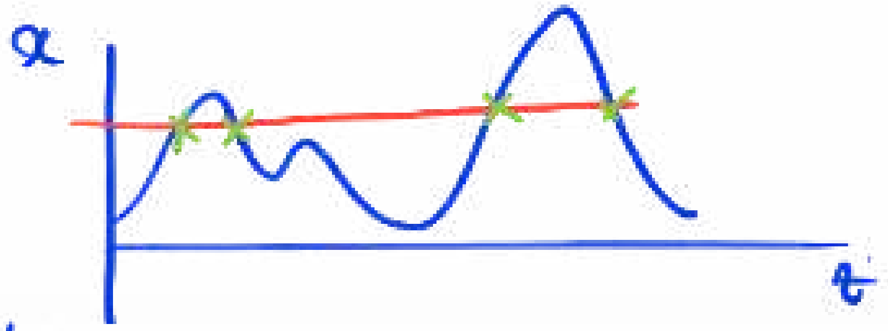
```
loop on Data  $i=1, N$
    $K = \frac{x(i)}{\Delta x}$
    loop  $l = K_{min}, K, \Delta K$
        $N\theta(l) = N\theta(l) + 1$
    End loop
End loop
$N\theta = \frac{N\theta}{N}$
```



تمام ترازهای پس‌تر را تحت تأثیر قرار می‌دهد.

در تراز سبز سهم دارد ولی در تراز آبی را اجرا [illegible]

نرمال می‌شود

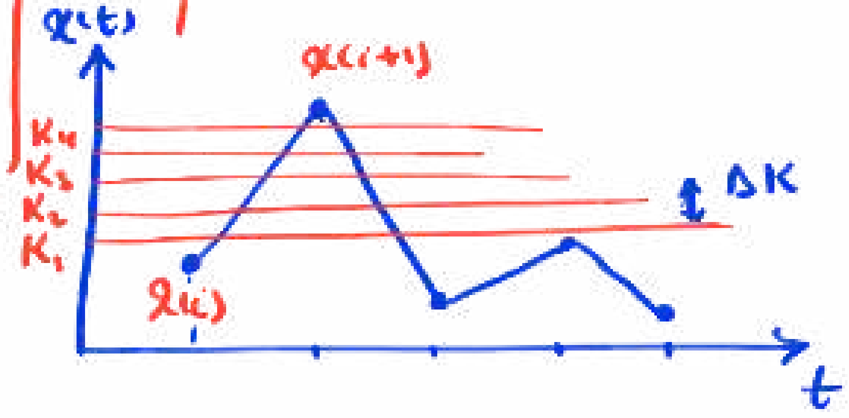


# MinKowski1

```
loop on Data  $i=1, N-1$
    $K_1 = \frac{x(i)}{\Delta x}$
    $K_2 = \frac{x(i+1)}{\Delta x}$
    If  $x(i+1) \geq x(i)$
        loop  $l = K_1, K_2, \Delta K$
            $N1(l) = N1(l) + 1$
        End loop
    End if
End loop
$N1 = N1/N$
$N_1 = 2N_1$  ←  only for Stationary and Regular Data
End Program
```

قطع برخوردها با شیب مثبت را شمرده ایم

ترازهای مختلف



# The style of a typical Procedure oriented programming (POP) (1)

متغیرها معرفی می گردند (global)

```
module parameters
use numerical_libraries
implicit none
real(8),parameter::num=150
character(256)::numstring
real(8) x,yy,y,q1(-100:100),q2(0:100),q3(-100:100,0:10)
Real*8, ALLOCATABLE :: z1(:),P12(:,:),z2(:)
INTEGER    IR, IS, J
COMPLEX(8)    C, CEXP, CMPLX, COEF(num,num),coef_cmb(num,num)
end module parameters
!***********************Main program **************
main_program
use parameters
implicit none
INTEGER, PARAMETER:: double=SELECTED_REAL_KIND(15,307)
real(double) CHSQ,DF1,t2,chi_sq_GSN_GN, &

&     P_value_t_GSN_GN(0:1000),t_GS_G(0:10000)
integer LDB,num11
open(10,file='input_file.txt')    !*********** file explanation
pi=4*atan(1.0)
!*********** read data from input file*************
do i =1,10000000
      read(10,*,end=9)dummy
      counter=counter+1
enddo
9     rewind(10)
allocate (z1(counter), z2(counter), P12(counter,conter))
do i=1,counter
      read(10,*) z1(i), z2(i)
enddo
call initial_condition    !****** call a typical subroutine
call random_seed
call noise
call gaussian_map
call print_result
```

## The style of a typical Procedure oriented programming (POP) (2)

```
      deallocate (z1,z2,P12)
      end maine_program
!************ subroutines
subroutine initial_condition
use parameters
implicit none
im=(0,1)
lda=num
nca=num
nra=num
end subroutine initial_condition
!********************************
subroutine noise
use parameters
implicit none
call random_seed
do i=1,num**2
call random_number(x)
call random_number (y)
z1(i)=sqrt(-2*log(x))*cos(2*pi*y)
z2(i)=sqrt(-2*log(x))*sin(2*pi*y)
enddo
End subroutine noise
!*****************
```

http://facultymembers.sbu.ac.ir/movahed/attachments/computational_all.pdf

# The style of a typical Object Oriented Programming (OOP)

مثالی در مورد یک برنامه که به صورت شئ‌گرا OOP نوشته شده‌است (http://fortranwiki.org):

```
module class_Circle
      implicit none
      private
      public :: Circle, circle_area, circle_print
      real :: pi = 3.1415926535897931d0 ! Class-wide private constant
      type Circle
       real :: radius
       end type Circle
      contains
      function circle_area(this) result(area)
      type(Circle), intent(in) :: this
      real :: area
      area = pi * this%radius**2
      end function circle_area
      subroutine circle_print(this)
      type(Circle), intent(in) :: this
      real :: area
      area = circle_area(this)  ! Call the circle_area function
      print *, 'Circle: r = ', this%radius, ' area = ', area
      end subroutine circle_print
end module class_Circle
program circle_test
 use class_Circle
 implicit none
 type(Circle) :: c     ! Declare a variable of type Circle(برای شئ یک حالت نسبت داده‌است)
 c = Circle(1.5)        ! Use the implicit constructor, radius = 1.5.
 call circle_print(c)   ! Call a class subroutine
end program circle_test
```

**Introduction to command's line in Linux by Seyed Danial Movahed**

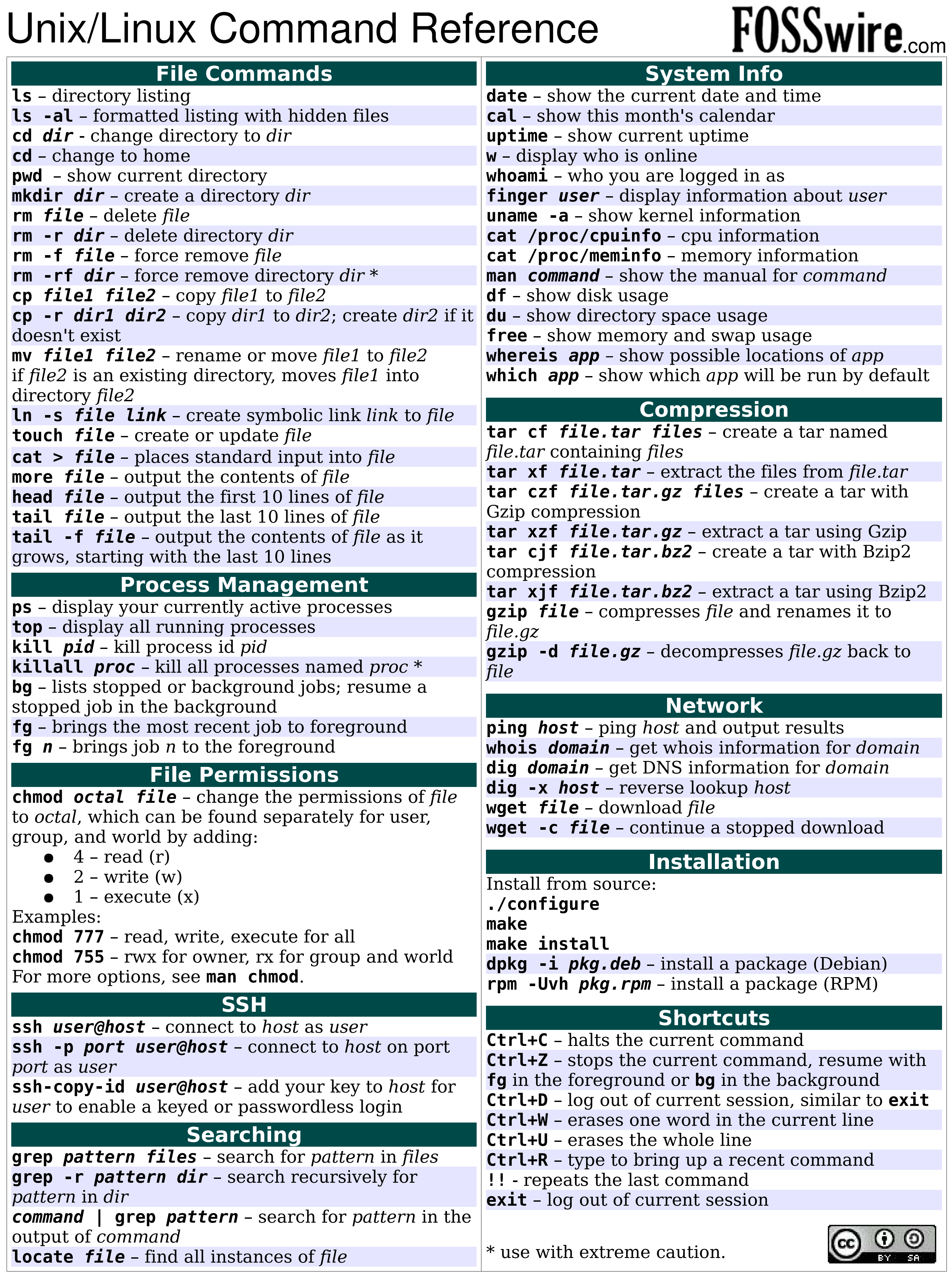
**Some references**

[1] Tansley, David SW, and David V. Tansley. *Linux and UNIX shell programming*. Addison-Wesley Professional, 2000.
[2] Greenberg, Michael, Konstantinos Kallas, and Nikos Vasilakis. "Unix shell programming: the next 50 years." *Proceedings of the Workshop on Hot Topics in Operating Systems*. 2021.
[3] Kidwai, Abdullah, et al. "A comparative study on shells in Linux: A review." *Materials Today: Proceedings* 37 (2021): 2612-2616.
[4] Kappelmann-Fenzl, Melanie. "Introduction to Command Line (Linux/Unix)." *Next Generation Sequencing and Data Analysis*. Cham: Springer International Publishing, 2021. 71-78.

| Lecture no. | Topics |
|---|---|
| Lecture 1 | 1) Linux and Unix<br>2) Why Linux?<br>3) What is Shell?<br>4) Installation of packages |
| Lecture 2 | 5-1) Common Commands (sudo, ls, file, df, du, whoami, awk, sort, hed, more, less, man, curl, wget,...) |
| Lecture 3 | 5-2) Common Commands (sudo, ls, file, df, du, whoami, awk, sort, hed, more, less, man, curl, wget ...) |
| Lecture 4 | 6) File permission and Ownership<br>7) Find and locate |
| Lecture 5 | 8) Running commands (top, nohup, kill, kill signal, killall, …)<br>9) Variables (environmental variables, path, ...)<br>10) Command execution order |
| Lecture 6 | 11) cat and Editors<br>12) Grep and Regular expression |
| Lecture 7 | 13-1) Shell script (read, if, for, echo, …) |
| Lecture 8 | 13-2) Shell script (read, if, for, echo, …) |
| Lecture 9 | 14) alias and login environment, shell functions<br>15) Connection to clusters (ssh, scp, …) and multiplexers (tmux, screen,..) |
| Lecture 10 | 16) init systems<br>17) Version controls |

Course 20 some preliminaries on GNU/Linux By Seyed Danial Movahed Video (14030630)

https://drive.google.com/file/d/1XIAFFyDoWLbRSixFalh_K9r5cQijb6sv/view?usp=drive_link



# Unix/Linux Command Reference

FOSSwire.com

## File Commands

**ls** – directory listing
**ls -al** – formatted listing with hidden files
**cd *dir*** - change directory to *dir*
**cd** – change to home
**pwd** – show current directory
**mkdir *dir*** – create a directory *dir*
**rm *file*** – delete *file*
**rm -r *dir*** – delete directory *dir*
**rm -f *file*** – force remove *file*
**rm -rf *dir*** – force remove directory *dir* *
**cp *file1 file2*** – copy *file1* to *file2*
**cp -r *dir1 dir2*** – copy *dir1* to *dir2*; create *dir2* if it doesn't exist
**mv *file1 file2*** – rename or move *file1* to *file2* if *file2* is an existing directory, moves *file1* into directory *file2*
**ln -s *file link*** – create symbolic link *link* to *file*
**touch *file*** – create or update *file*
**cat > *file*** – places standard input into *file*
**more *file*** – output the contents of *file*
**head *file*** – output the first 10 lines of *file*
**tail *file*** – output the last 10 lines of *file*
**tail -f *file*** – output the contents of *file* as it grows, starting with the last 10 lines

## Process Management

**ps** – display your currently active processes
**top** – display all running processes
**kill *pid*** – kill process id *pid*
**killall *proc*** – kill all processes named *proc* *
**bg** – lists stopped or background jobs; resume a stopped job in the background
**fg** – brings the most recent job to foreground
**fg *n*** – brings job *n* to the foreground

## File Permissions

**chmod *octal file*** – change the permissions of *file* to *octal*, which can be found separately for user, group, and world by adding:

- 4 – read (r)
- 2 – write (w)
- 1 – execute (x)

Examples:
**chmod 777** – read, write, execute for all
**chmod 755** – rwx for owner, rx for group and world
For more options, see **man chmod**.

## SSH

**ssh *user@host*** – connect to *host* as *user*
**ssh -p *port user@host*** – connect to *host* on port *port* as *user*
**ssh-copy-id *user@host*** – add your key to *host* for *user* to enable a keyed or passwordless login

## Searching

**grep *pattern files*** – search for *pattern* in *files*
**grep -r *pattern dir*** – search recursively for *pattern* in *dir*
***command* | grep *pattern*** – search for *pattern* in the output of *command*
**locate *file*** – find all instances of *file*

## System Info

**date** – show the current date and time
**cal** – show this month's calendar
**uptime** – show current uptime
**w** – display who is online
**whoami** – who you are logged in as
**finger *user*** – display information about *user*
**uname -a** – show kernel information
**cat /proc/cpuinfo** – cpu information
**cat /proc/meminfo** – memory information
**man *command*** – show the manual for *command*
**df** – show disk usage
**du** – show directory space usage
**free** – show memory and swap usage
**whereis *app*** – show possible locations of *app*
**which *app*** – show which *app* will be run by default

## Compression

**tar cf *file.tar files*** – create a tar named *file.tar* containing *files*
**tar xf *file.tar*** – extract the files from *file.tar*
**tar czf *file.tar.gz files*** – create a tar with Gzip compression
**tar xzf *file.tar.gz*** – extract a tar using Gzip
**tar cjf *file.tar.bz2*** – create a tar with Bzip2 compression
**tar xjf *file.tar.bz2*** – extract a tar using Bzip2
**gzip *file*** – compresses *file* and renames it to *file.gz*
**gzip -d *file.gz*** – decompresses *file.gz* back to *file*

## Network

**ping *host*** – ping *host* and output results
**whois *domain*** – get whois information for *domain*
**dig *domain*** – get DNS information for *domain*
**dig -x *host*** – reverse lookup *host*
**wget *file*** – download *file*
**wget -c *file*** – continue a stopped download

## Installation

Install from source:

```
./configure
make
make install
```

**dpkg -i *pkg.deb*** – install a package (Debian)
**rpm -Uvh *pkg.rpm*** – install a package (RPM)

## Shortcuts

**Ctrl+C** – halts the current command
**Ctrl+Z** – stops the current command, resume with **fg** in the foreground or **bg** in the background
**Ctrl+D** – log out of current session, similar to **exit**
**Ctrl+W** – erases one word in the current line
**Ctrl+U** – erases the whole line
**Ctrl+R** – type to bring up a recent command
**!!** - repeats the last command
**exit** – log out of current session

* use with extreme caution.

http://facultymembers.sbu.ac.ir/movahed/attachments/computational_all.pdf

## Terminal: General properties

Take a look at the

http://facultymembers.sbu.ac.ir/movahed/attachments/Introduction%20to%20command%20Linux.pdf
http://facultymembers.sbu.ac.ir/movahed/attachments/computational_all.pdf

### Some essential commands (1)

1) `ls` : list directory contents
   `$` (run as normal user) `ls (ls -l; ls -a; ls -t; ls -r; ls -h; ls -lh; ls -latrh)`
2) `cd` : Change working directory
   `$ cd dirname`
   `$ cd ..` (move up one directory)
   `$ cd address` (move to address )
3) `pwd` : Print working directory
   `$ pwd`

## Terminal: General properties

Some essential commands (2)

4) `rm` : Remove a file (rm -r; rm -f : forced remove)

```
$ rm filename
```

`$ rm -r` : Remove recursively a folder

5) `cp` : Copy a file (cp -r)

```
$ cp sourcefile destinationfile
$ cp file1 file2 … destinationdirector
$ cp -r sourcedirectory
destinationdirectory
```

6) `mkdir` : Make an empty directory (mkdir -p)

```
$ mkdir dirname
```

`$ mkdir -p dirname` (making a directory without error)

# Terminal: General properties

## Some essential commands (3)

7) `touch` : Make an empty file

```
$ touch filename
```

8) `mv` : Move (transform) a file/folder to another address/ name

```
$ mv file1\folder1 file2\folder2 … destinationdirectory
$ mv initialfilename\initialfoldername finalfilename\finalfoldername
```

9) `cat` : Print file contents

```
$ cat filename
```

10) `top` : Display running processes

## Terminal: General properties

### Some essential commands (4)

11) chmod : Change the permissions of file\directory (view permissions with ls -l)

```
-rw-r-----  1 sadegh  sadegh  0 Sep 27 17:40 testmode
```

Permission for the User owning the file: Read, Write and no eXecute permission

Permission for the users in a Group owning the file: Read, no Write and no eXecute permission

Permission for Others owning the file: no Read, no Write and no eXecute permission

## Terminal: General properties

### Some essential commands (5)

11-continue)

**`chmod`** : Change the permissions of file\directory

(view permissions with `ls -l`)

```
$ ls -l testmode
-rw-r-----   1 sadegh  sadegh  0 Sep 27 17:40 testmode

$ chmod g+w testmode
-rw-rw----   1 sadegh  sadegh  0 Sep 27 17:40 testmode
 110 -> 1x2^2+1x2^1+0x2^0=6
    110 -> 1x2^2+1x2^1+0x2^0=6
       000 -> 0x2^2+0x2^1+0x2^0=0
  $ chmod g+w testmode  is equivalent to $ chmod 660 testmode
                        in this example
$ chmod g-w testmode
-rw-r-----   1 sadegh  sadegh  0 Sep 27 17:40 testmode
```

## Terminal: General properties

- Making alias and unalias (Local capability):
  Example 1: ls -> show the list of content in the current location;

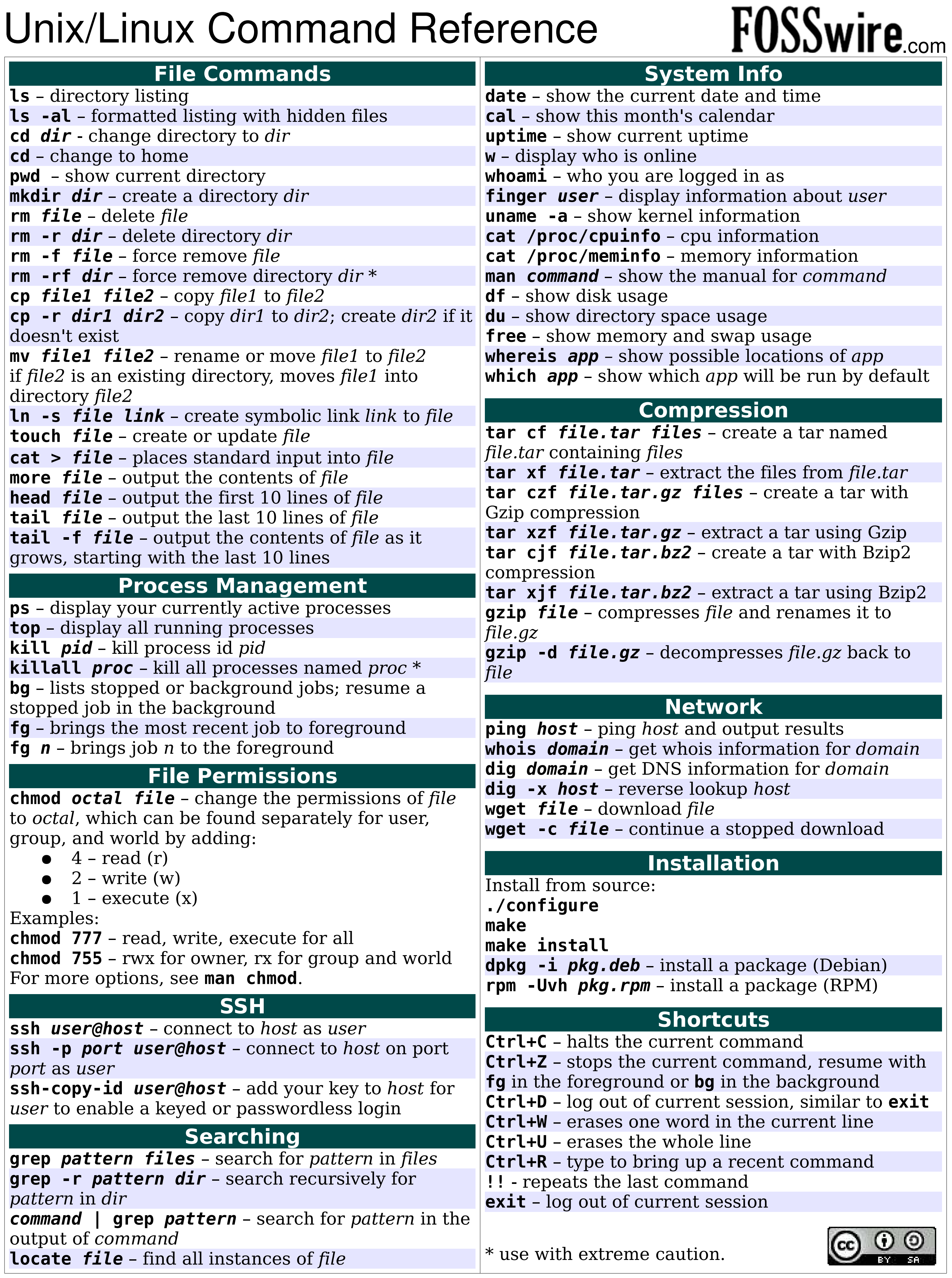
  alias “list” instead of “ls”

  ```
  $ alias list=“ls"
  
  $ unalias list
  ```

  Example 2: making an alias to open a typical program

  ```
  $ alias math="open -a Mathematica”  only for Mac OS
  ```
- To make a permanent alias add it to your **shell’s rc** file
  Example: `nano ~/.zshrc alias add list=‘ls’`

  ```
  $ which $SHELL
  $ source ~/.zshrc
  ```



# Unix/Linux Command Reference

FOSSwire.com

## File Commands

**ls** – directory listing
**ls -al** – formatted listing with hidden files
**cd *dir*** - change directory to *dir*
**cd** – change to home
**pwd** – show current directory
**mkdir *dir*** – create a directory *dir*
**rm *file*** – delete *file*
**rm -r *dir*** – delete directory *dir*
**rm -f *file*** – force remove *file*
**rm -rf *dir*** – force remove directory *dir* *
**cp *file1 file2*** – copy *file1* to *file2*
**cp -r *dir1 dir2*** – copy *dir1* to *dir2*; create *dir2* if it doesn't exist
**mv *file1 file2*** – rename or move *file1* to *file2* if *file2* is an existing directory, moves *file1* into directory *file2*
**ln -s *file link*** – create symbolic link *link* to *file*
**touch *file*** – create or update *file*
**cat > *file*** – places standard input into *file*
**more *file*** – output the contents of *file*
**head *file*** – output the first 10 lines of *file*
**tail *file*** – output the last 10 lines of *file*
**tail -f *file*** – output the contents of *file* as it grows, starting with the last 10 lines

## Process Management

**ps** – display your currently active processes
**top** – display all running processes
**kill *pid*** – kill process id *pid*
**killall *proc*** – kill all processes named *proc* *
**bg** – lists stopped or background jobs; resume a stopped job in the background
**fg** – brings the most recent job to foreground
**fg *n*** – brings job *n* to the foreground

## File Permissions

**chmod *octal file*** – change the permissions of *file* to *octal*, which can be found separately for user, group, and world by adding:

- 4 – read (r)
- 2 – write (w)
- 1 – execute (x)

Examples:
**chmod 777** – read, write, execute for all
**chmod 755** – rwx for owner, rx for group and world
For more options, see **man chmod**.

## SSH

**ssh *user@host*** – connect to *host* as *user*
**ssh -p *port user@host*** – connect to *host* on port *port* as *user*
**ssh-copy-id *user@host*** – add your key to *host* for *user* to enable a keyed or passwordless login

## Searching

**grep *pattern files*** – search for *pattern* in *files*
**grep -r *pattern dir*** – search recursively for *pattern* in *dir*
***command* | grep *pattern*** – search for *pattern* in the output of *command*
**locate *file*** – find all instances of *file*

## System Info

**date** – show the current date and time
**cal** – show this month's calendar
**uptime** – show current uptime
**w** – display who is online
**whoami** – who you are logged in as
**finger *user*** – display information about *user*
**uname -a** – show kernel information
**cat /proc/cpuinfo** – cpu information
**cat /proc/meminfo** – memory information
**man *command*** – show the manual for *command*
**df** – show disk usage
**du** – show directory space usage
**free** – show memory and swap usage
**whereis *app*** – show possible locations of *app*
**which *app*** – show which *app* will be run by default

## Compression

**tar cf *file.tar files*** – create a tar named *file.tar* containing *files*
**tar xf *file.tar*** – extract the files from *file.tar*
**tar czf *file.tar.gz files*** – create a tar with Gzip compression
**tar xzf *file.tar.gz*** – extract a tar using Gzip
**tar cjf *file.tar.bz2*** – create a tar with Bzip2 compression
**tar xjf *file.tar.bz2*** – extract a tar using Bzip2
**gzip *file*** – compresses *file* and renames it to *file.gz*
**gzip -d *file.gz*** – decompresses *file.gz* back to *file*

## Network

**ping *host*** – ping *host* and output results
**whois *domain*** – get whois information for *domain*
**dig *domain*** – get DNS information for *domain*
**dig -x *host*** – reverse lookup *host*
**wget *file*** – download *file*
**wget -c *file*** – continue a stopped download

## Installation

Install from source:

```
./configure
make
make install
```

**dpkg -i *pkg.deb*** – install a package (Debian)
**rpm -Uvh *pkg.rpm*** – install a package (RPM)

## Shortcuts

**Ctrl+C** – halts the current command
**Ctrl+Z** – stops the current command, resume with **fg** in the foreground or **bg** in the background
**Ctrl+D** – log out of current session, similar to **exit**
**Ctrl+W** – erases one word in the current line
**Ctrl+U** – erases the whole line
**Ctrl+R** – type to bring up a recent command
**!!** - repeats the last command
**exit** – log out of current session

* use with extreme caution.

http://facultymembers.sbu.ac.ir/movahed/attachments/computational_all.pdf

## Shell script

Some main questions:

1) What is the shell script good for?
2) What is the shell script itself?
3) How can make a shell script?

## What is a shell script good for?

1) Making a recipe;
2) Including different commands ranging from making a folder to call a compiler to compile and then run an executive program and so on;

## Shell script: Structure

#! (shebang (hashbang) character):

$ `which $SHELL` show you where your current shell and add it

/bin/zsh

```
#!/bin/bash
or
#!/bin/zsh
i=0
num=100
for((i=1; i<=num; i++)); do
mkdir -p sadegh.${i}
name=sadegh.${i}
cp danial_story.jpg ${name}
echo ${name}
done
```

To make an executive file: change the mode via

chmod u+x file.sh

## Shell script: Example 2

Example 2: Make a shell script to do following tasks:
reading from a file and make associated folders and plot input data

## Shell script: Example 2

Example 2: Make a shell script to do following tasks:
reading from a file and make associated folders and move a typical file to each created folder

```
#!/bin/bash
i=0
for name in $(cat input) ; do
let "i=i+1"
C[i]=$name
echo $name
mkdir -p $name
cp danial_story.jpg ${name}
done
```

## Shell script: Example 3

Example 3: Make a shell script to do following tasks:
1) We have 48 text file entitled 1.txt to 48.txt;
2) We have a file including the name of countries and we would like to assign each text file to the corresponding country's name in separated folders. Also we are going to select all available pairs (all combinations) $\frac{48!}{(2)!(48-2)!} = 1128$
3) Move each two corresponding data to associated folder and plot the data in that folder

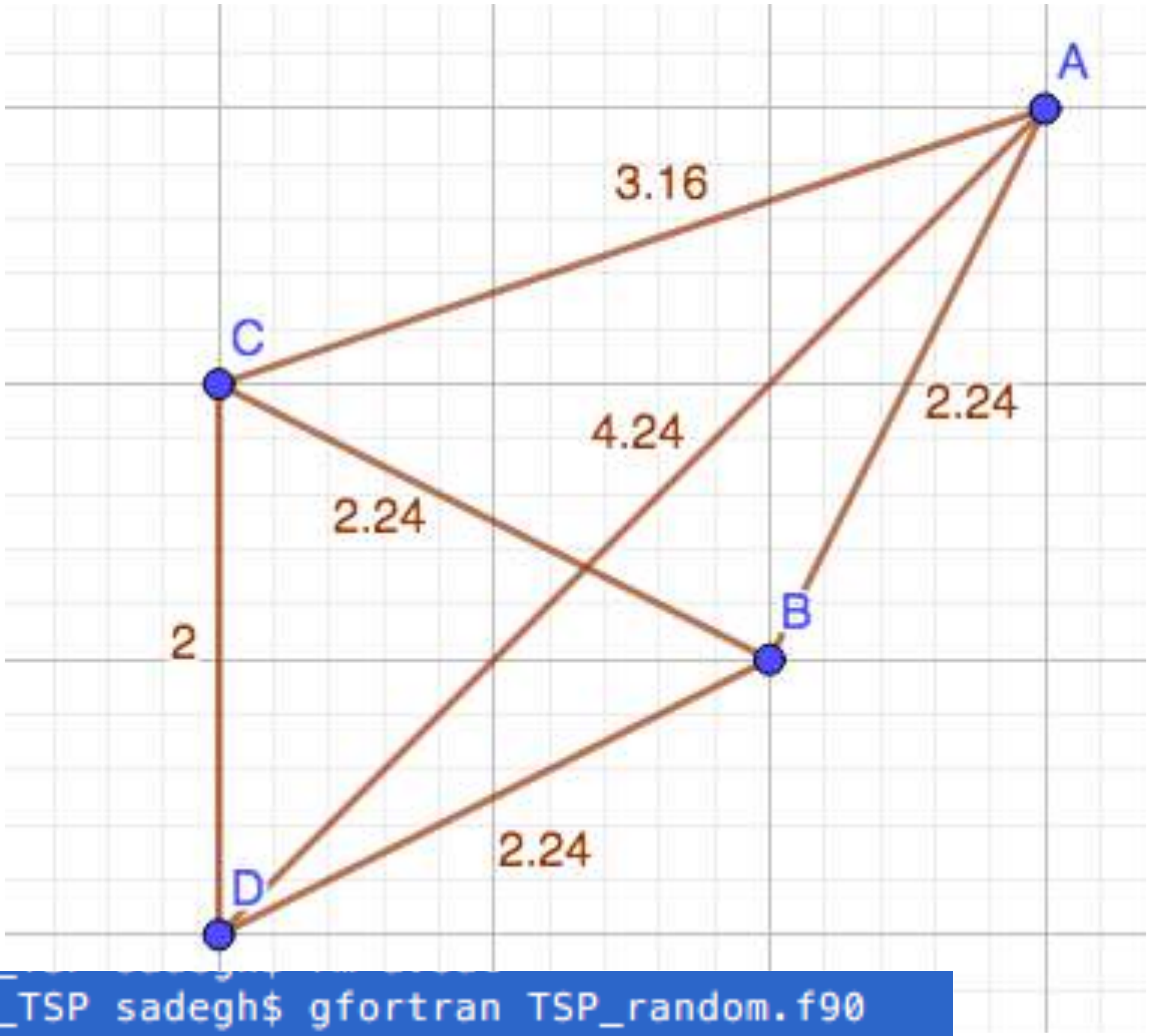
## Bash script: Example 3

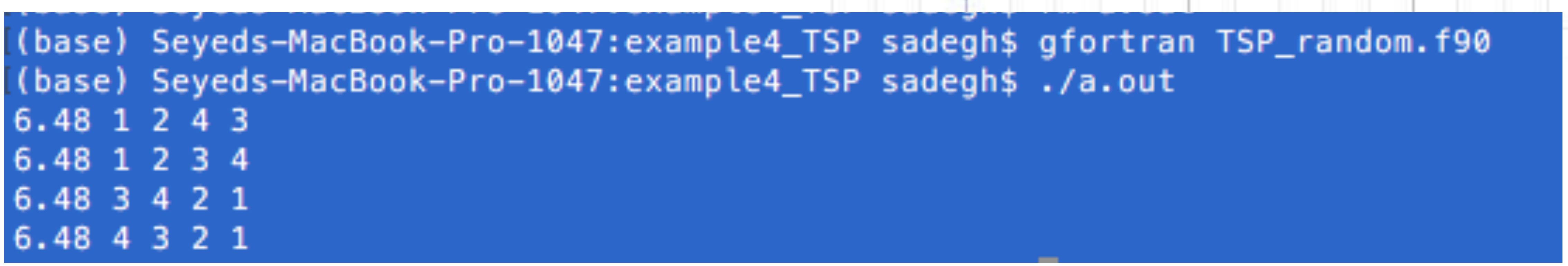
```
#!/usr/bin/env bash

i=0
for name in $(cat list_arrange); do
    let i=$i+1
    c[i]=$name
    #echo $name
done
let num=$i
for ((i=1; i<=$num; i++)); do
    let k=$i+1
    for ((j=$k; j<=$num; j++)); do
        mkdir -p ${c[i]}_${c[j]}
        cp $i.txt ${c[i]}_${c[j]}/${c[i]}.txt
        cp $j.txt ${c[i]}_${c[j]}/${c[j]}.txt
        echo ${c[i]}
        echo ${c[j]}
        cd ${c[i]}_${c[j]}
        python3.6 ../plot.py ${c[i]} ${c[j]}
        cd ..
    done
done
```

## Shell script: Example 4

Example 4: Traveling Salesman Problem (TSP)

A,B,C,D
A,B,D,C
C,D,B,A
D,C,B,A





```
(base) Seyeds-MacBook-Pro-1047:example4_TSP sadegh$ gfortran TSP_random.f90
(base) Seyeds-MacBook-Pro-1047:example4_TSP sadegh$ ./a.out
6.48 1 2 4 3
6.48 1 2 3 4
6.48 3 4 2 1
6.48 4 3 2 1
```

## Bash script: Example 5

## Bash script: Example 6

Others left as exercises for you

## Some useful commands

Example 1: we are interested in copying a file from our machine to cluster

```
scp ./plot.py m_movahed@192.168.220.100:/share/
users/m_movahed/TDA
```

Example 2: after finishing our program in the cluster, we want to move the results from cluster to our local machine

```
scp m_movahed@192.168.220.100:/share/users/
m_movahed/TDA/plot1.py .
```

Notice: Use "tmux" when you are connected to cluster

# Some useful commands: tmux command

1) Connecting to the cluster
2) In corresponding terminal type: `tmux` (pre-installed) to create a session
3) `tmux ls` (shows a list of sessions)
4) `Ctrl+b  %` (splitting vertically the terminal)
5) `Ctrl+b  "` (splitting horizontally the terminal)
6) moving between different sessions
   `Ctrl+b  arrows (top, down, left, right)`
7) Submitting a job and running a program
8) `Ctrl+b d` ——> to Detach from session
9) `tmux attach -t <session-ID>`
10) Exit (disconnecting from cluster)
11) To check our job connect to cluster, `tmux ls, tmux attach -t <session-ID>`
12) To kill the session, `tmux kill-session -t <session-ID>`

see the Pooyan's lectures for more details via
http://ccg.sbu.ac.ir/resources/computers/

## Some useful commands

Notice: Use “tmux” when you are connected to cluster after reconnecting use “tmux attach”

Example:

```
while true
do
sleep 1
echo “Hello Dear”
done
```

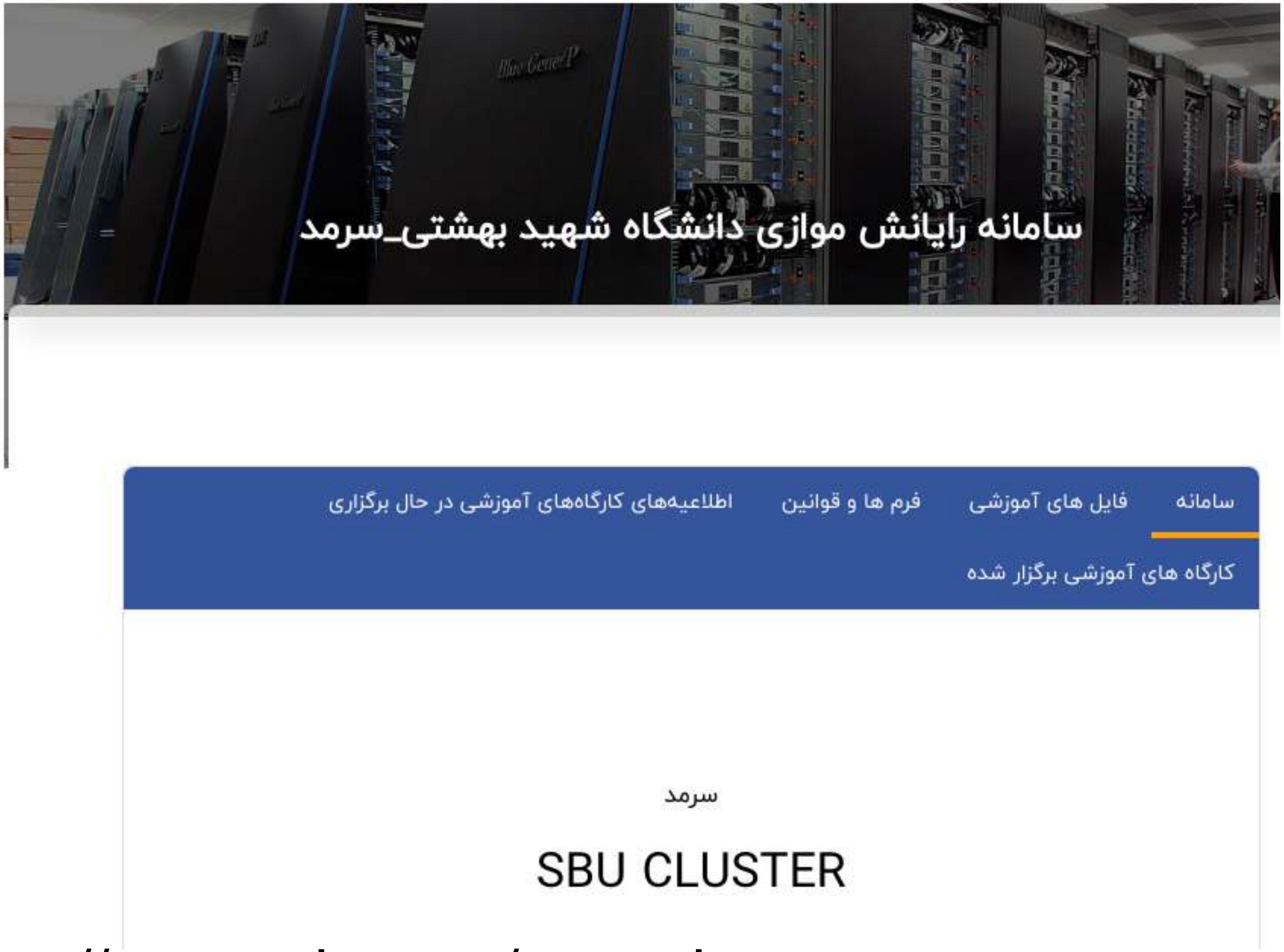
see the Pooyan’s lectures for more details via http://ccg.sbu.ac.ir/resources/computers/

## Running a job on a cluster

1) You need to make a shell script; (see the example)
2) You need to know terminal multiplexing

see the end of this file:
http://facultymembers.sbu.ac.ir/movahed/attachments/computational_all.pdf

## Sarmad


سامانه رایانش موازی دانشگاه شهید بهشتی_سرمد

سامانه
فایل های آموزشی
فرم ها و قوانین
اطلاعیه‌های کارگاه‌های آموزشی در حال برگزاری
کارگاه های آموزشی برگزار شده

سرمد

SBU CLUSTER

https://resevp.sbu.ac.ir/sarmad

از توجه شما سپاسگزارم