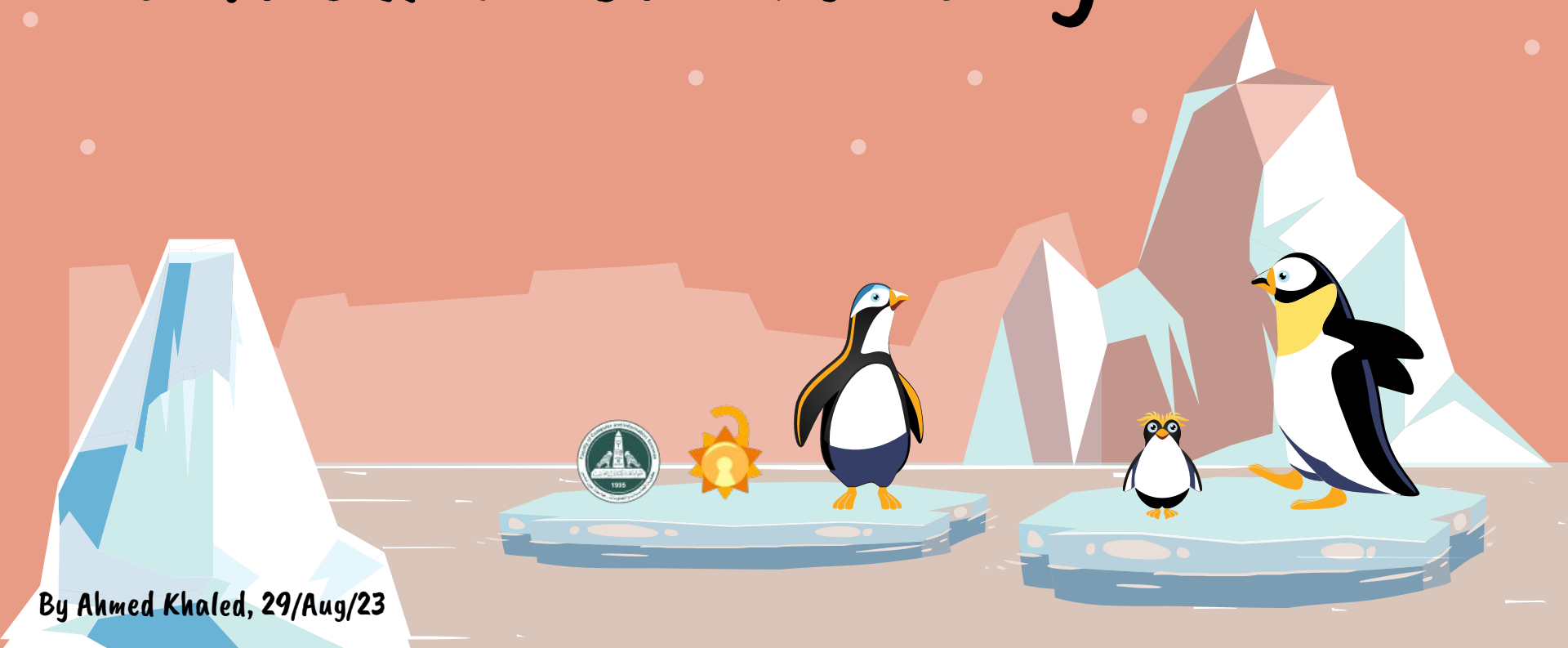
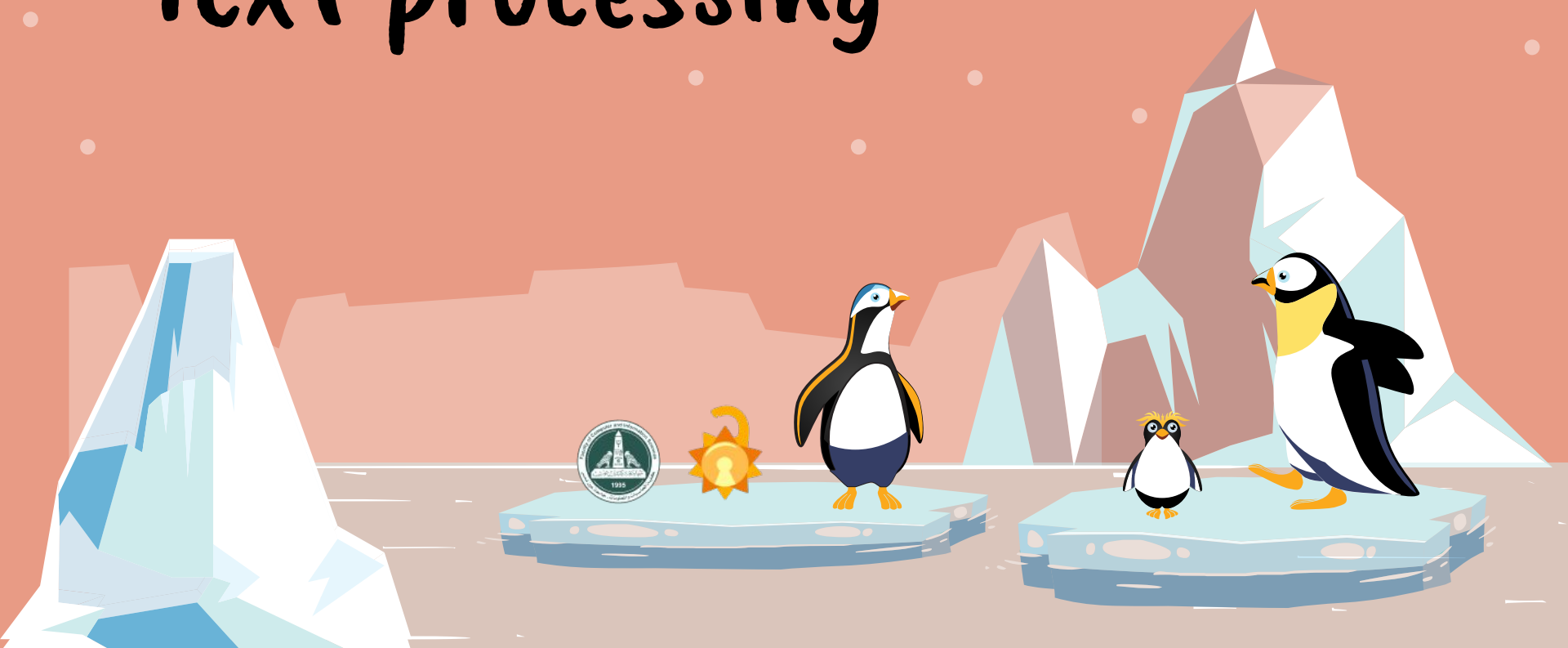
Linux Summer Training 23



By Ahmed Khaled, 29/Aug/23

Text processing



```
string user_answer;
cin >> user_answer;

if(user_answer == "yes" || user_answer == "YES" || user_answer == "Yes" || user_answer == "YeS")
    cout << "Great!" << endl;
else
    cout << "Please enter a valid answer" << endl;
```

yees?

yes please?

Outcomes

- Increase productivity (By automating tasks)
- Manipulate text (Search, Filter, Replace, Transform etc.)
- Write advanced bash scripts
- Validate user input in a smarter way
- Understand new concepts

Agenda

01 Simple commands

02 Regex

03 Grep

04 Sed

05 Awk

06 Summary

Simple commands

Sort

Options

- -n -> numerical
- -r -> reverse

Rev

Uniq

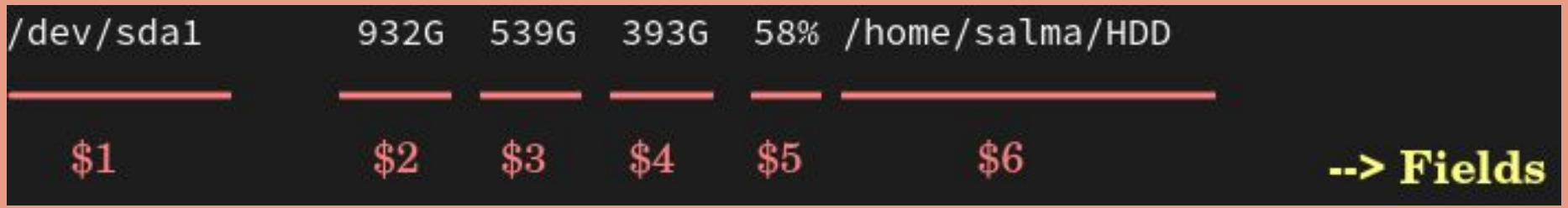
Options

- -c -> count
- -d -> duplicates
- -u -> unique

Cut

Options

- -c -> characters
- -f -> fields
- -d -> delimiter



Example

Tr

Options

- -d -> delete
- -s -> replace sequence with char

Hands on

```
mint@mint:~/osc$ cat /etc/shells
# /etc/shells: valid login shells
/bin/sh
/bin/bash
/usr/bin/bash
/bin/rbash
/usr/bin/rbash
/usr/bin/sh
/bin/dash
/usr/bin/dash
```

```
bash
dash
rbash
sh
shells= valid login shells
```

Sort

Rev

Uniq

Cut (-c, -f, -d)

Tr

Hands on Solution!

Paste

Options

- -s -> single line
- -d -> delimiter

Recap

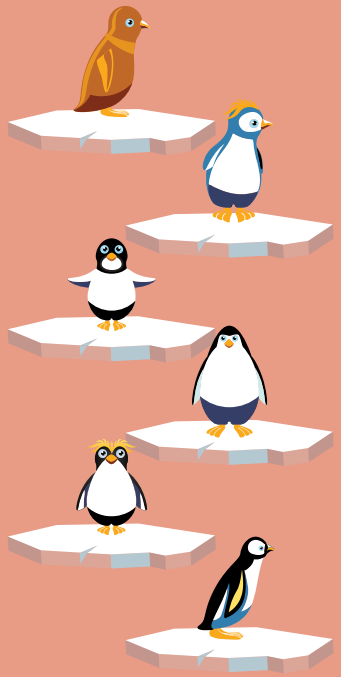
Regex

Example

```
string user_answer;
cin >> user_answer;

if(user_answer == "yes" || user_answer == "YES" || user_answer == "Yes" || user_answer == "YeS")
    cout << "Great!" << endl;
else
    cout << "Please enter a valid answer" << endl;
```

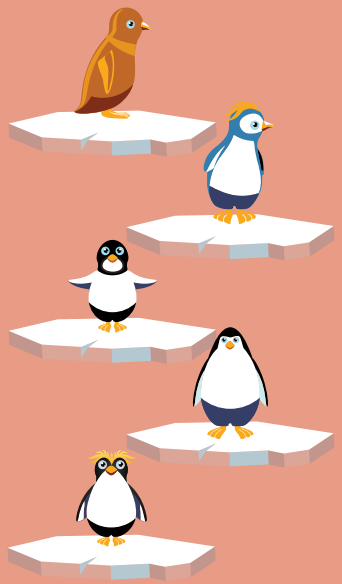
```
regex pattern("ye+s");

string user_answer;
cin >> user_answer;

if(regex_match(user_answer, pattern))
    cout << "Great!" << endl;
else
    cout << "Please enter a valid answer" << endl;
```

Regex symbols



?	0 or 1 of the previous (optional)
+	1 or more of the previous char
*	0 or more of the previous (‘+’ and ‘?’)
.	Anything
\w, \d, \s, \b	Characters, digits, spaces, word breaks
^, $	Beginning , end of line

Regex symbols



{ }	Ranges {4} , {4, } , {4,5}
[ ]	Character sets (^ -> except)
( )	Grouping
\1, \2	Reference
|	OR operator

Regex applications

- Programming languages, IDEs
- Finding patterns
- User input validation
- Cleaner code
- Smart search (Solution)

BREAK

Grep

Options

- -B, -E ({ }, (), ?, |, +), -P (\d) -> (Regex types)
- -i -> Case insensitive
- -n -> Line numbers
- -c (Count)
- -A, -B, -C -> (After, Before, Combination)
- -r -> Recursive

Hands on

- Grep email addresses

Ex: ahmed@gmail.com

- Grep old discord usernames

Ex: Ahmed#1010

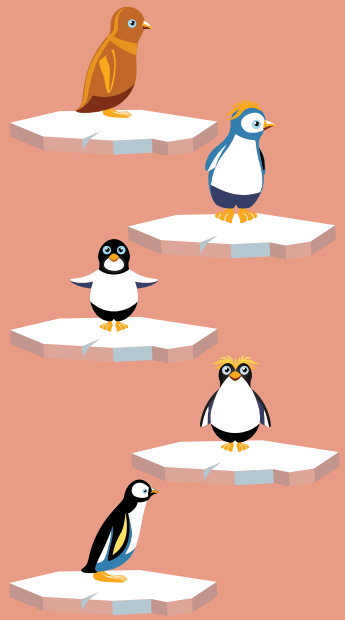
- Grep credit cards that contain 2 similar consecutive digits

Sed

Syntax

sed “s/pattern/replace/flags” file.txt

Sed options



Detect ‘/’s

Detect pattern in a line first (/ /)

Delete (d)

Multiple sed commands

Insert (&)

Hands on

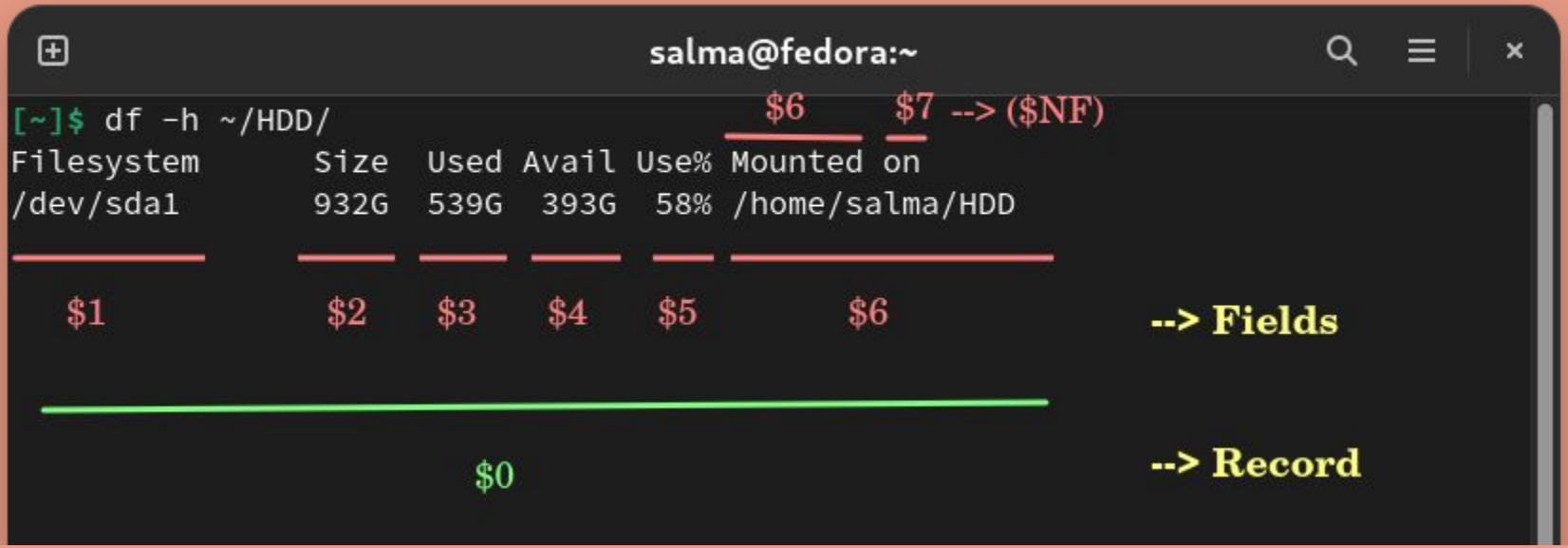
```
1234 5678 9101 1234
2999 5178 9101 2234
8482 3678 9102 1232
```

```
**** **** **** 1234
**** **** **** 2234
**** **** **** 1232
```

Hands on Solution!

Awk

How it works

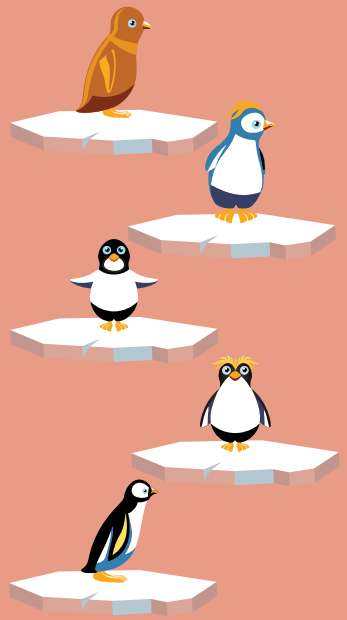

salma@fedora:~
[~]$ df -h ~/HDD/
Filesystem Size Used Avail Use% Mounted on
/dev/sda1 932G 539G 393G 58% /home/salma/HDD
$6
$7 --> ($NF)
$1
$2
$3
$4
$5
$6
--> Fields
$0
--> Record

Syntax

```
awk '/pattern/ {action}' file.txt
```

Example

Awk features



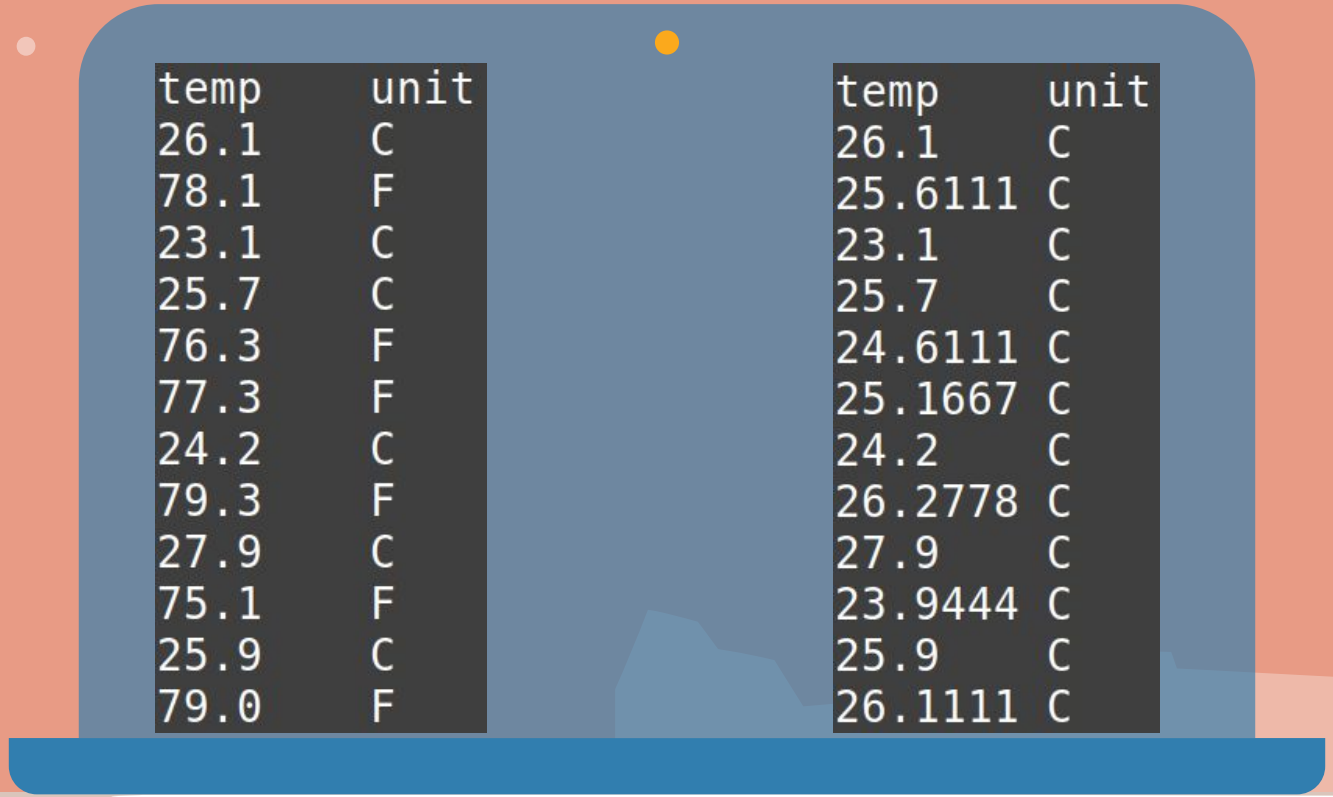
Arithmetic

If conditions + &&, ||

Relational expressions (~, !~, >)

Range patterns (,)

Begin, end


temp unit
26.1 C
78.1 F
23.1 C
25.7 C
76.3 F
77.3 F
24.2 C
79.3 F
27.9 C
75.1 F
25.9 C
79.0 F
temp unit
26.1 C
25.6111 C
23.1 C
25.7 C
24.6111 C
25.1667 C
24.2 C
26.2778 C
27.9 C
23.9444 C
25.9 C
26.1111 C

$$C = (F - 32) * 5/9$$

Hands on Solution!

Summary

Outcomes

- Increase productivity (By automating tasks)
- Manipulate text (Search, Filter, Replace, Transform etc.)
- Write advanced bash scripts
- Validate user input in a smarter way
- Understand new concepts

Thanks!