Bash Scripting (Rough notes)  
  
- Plain text editor

* Terminal emulator
* Bash  
    
  Diff OS :-   
  - MacOS :- Bash is pre installed
* Linux distros already has bash
* **B**ourne **A**gain **S**hell

Extension : .sh  
  
Step 1 : create a new empty file.sh

Step 2 : #!/bin/bash 🡪 Shebang

Step 3 :- chmod +x

Step 4 :- ./ 🡪 Run

Step 5 :- # is used as comments

Try later on :-

#!/bin/sh

echo

echo "Current working directory: $(pwd)"

echo

echo "Today's date & time: $(date)"

echo

* Variables :- only letters (A to Z or a to z), numbers (0 to 9) or (\_)  
  - Defining Variables   
  variable\_xyz=variable\_value

VAR1=”Deadpool”  
Var2=111

Read-only Variables

* #!/bin/sh
* NAME="DEADPOOL"
* readonly NAME
* NAME="IRONMAN"

Something that you need to remember :-   
- Never put spaces next to the equal sign, why?

Because Bash will that you you want to run a process instead

#!/bin/sh

HELLO="hello variable world"

echo $HELLO #should print "hello variable world"

The syntax of if, elseif, else is:

if <test\_expression>; then

<command-to-execute>

elif <test\_expression>; then

<command-to-execute>

else

<command-to-execute>

fi

Integer comparision :-

* -eq 🡪 equal to -if [ “$a” –eq “$b”]
* -ne 🡪 is not equal to -if [ “$a” –ne “$b”]
* -gt 🡪 is greater than -if [ “$a” -gt “$b”]
* -ge 🡪 greater than or equal to –if [ “$a” -ge “$b”]
* -lt 🡪 is less than
* < is less than
* <= 🡪 is less than or equal to
* > 🡪 is greater than
* >= 🡪 is greater than or equal to

#!/bin/bash

if [ 8 -gt 6 ] && [ 10 -eq 10 ];

then

echo "this is true"

fi

if [ "my life" == "mylife" ] && [ 3 -gt 10 ];

then

echo "conditions are false"

fi

-------------------------------------------------------------------------------------------

#!/bin/bash

#nested if statement

if [ $1 -gt 50 ]

then

echo "Number is greater than 50"

if (( $1 % 2 == 0 ))

then

echo "and it is an even number."

fi

fi

Note :- Make sure you pass the argument when you run this code like   
./ filename 100

Loop Syntax   
  
  
for variable in list

Do

Commands

Done

for (( expression1; expression2; expressions 3 ))  
  
do

commands

done

#!/bin/bash

learn="Hey welcome to devops pro course"

for learn in $learn

do

echo $learn

done

echo "loops is completed"

While loop :-

Syntax of while loops :-

while [ expression ];

do

commands;

multiple commands;

done

while [ condition ]; do commands; done

while control-command; do commands; done

#!/bin/bash

read -p "Enter starting number: " snum

read -p "Enter ending number: " enum

while [[ $snum -le $enum ]];

do

echo $snum

((snum++))

done

echo "This is what we wanted to print again"

Until Loops

until [ expression ]’

do

command1

command2

..

..

..

command

done

#!/bin/bash

i=1

until [ $i -gt 10 ]

do

echo $i

((i++))

Done

#!/bin/bash

max=5

a=1

b=0

until [[ $a -gt $max || $b -gt $max ]];

do

echo "a = $a & b = $b"

((a++))

((b++))

Done

SIMPLE PASSWORD GENERATOR :-   
  
#!/bin/bash

echo "Welcome to simple password generator"

echo "Please enter the length of the password"

read PASS\_LENGTH

for p in $(seq 1 5);

do

openssl rand -base64 48 | cut -c1-$PASS\_LENGTH

done