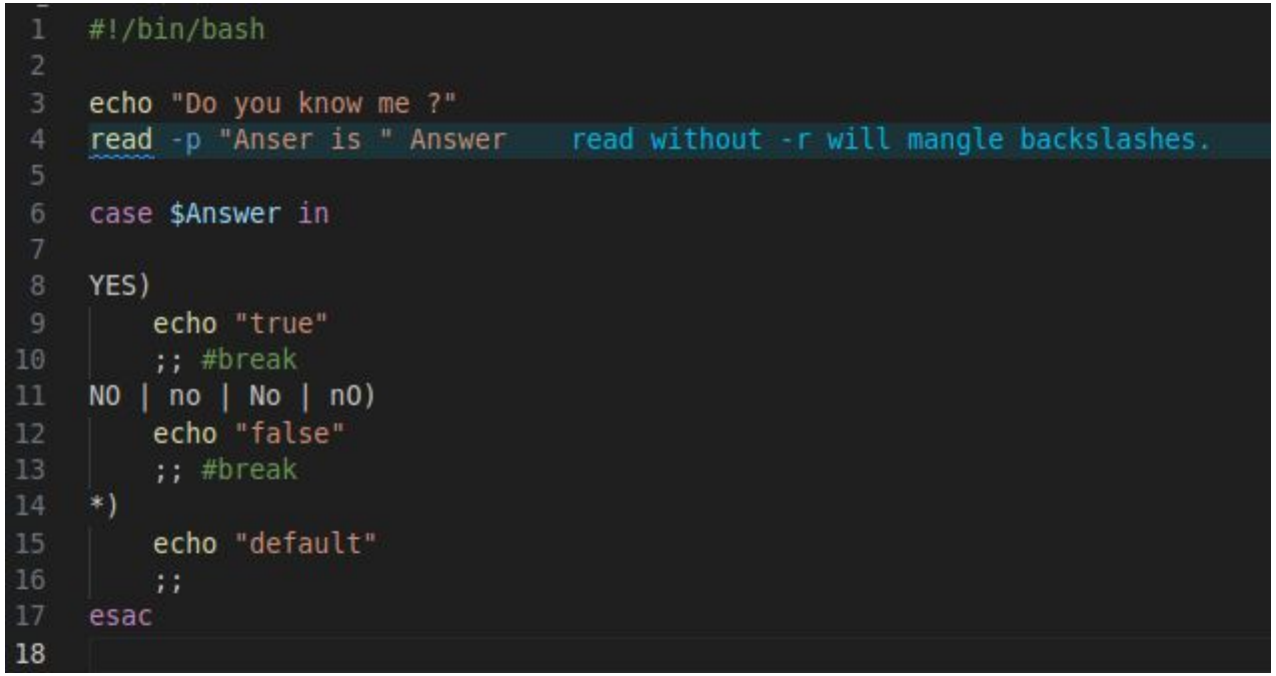
02 Function & String Bash

Moatasem Elsayed

Content

Switch

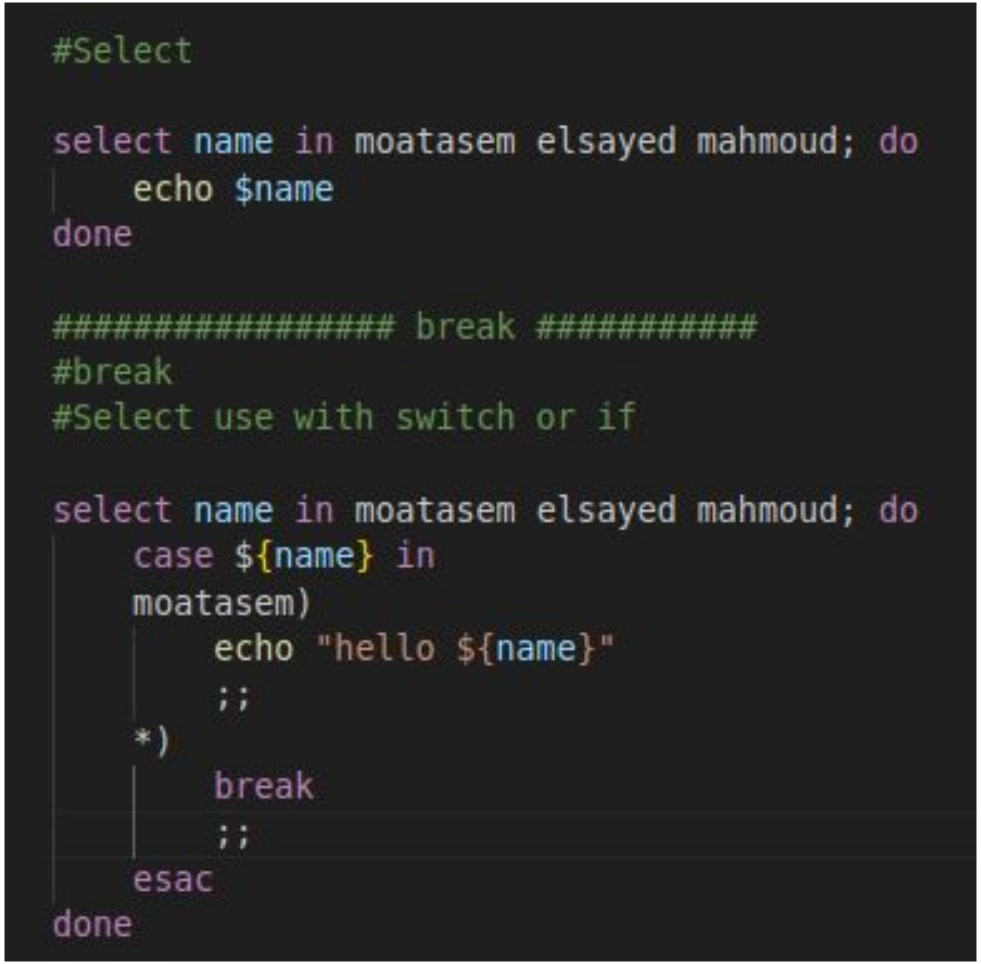
Switch Cash



```
#!/bin/bash

echo "Do you know me ?"
read -p "Anser is " Answer    read without -r will mangle backslashes.

case $Answer in

YES)
    echo "true"
    ;; #break
NO | no | No | nO)
    echo "false"
    ;; #break
*)
    echo "default"
    ;;
esac

```

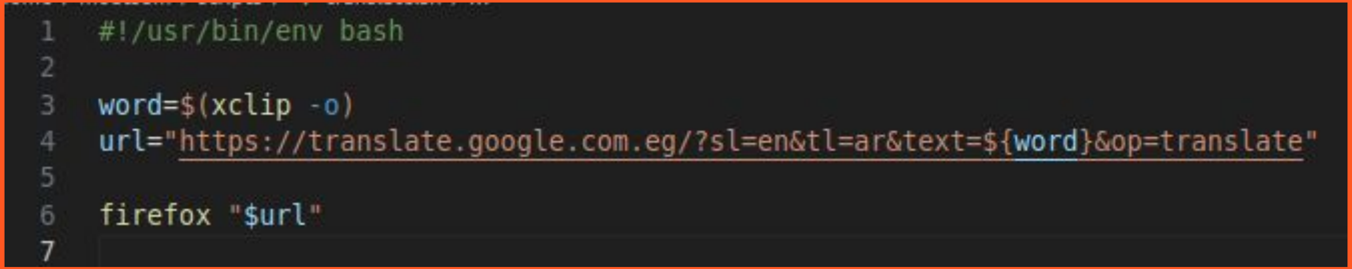
Select

select



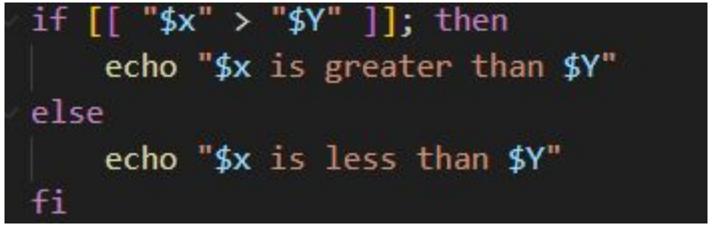
```
#Select

select name in moatasem elsayed mahmoud; do
    echo $name
done

################## break ###########
#break
#Select use with switch or if

select name in moatasem elsayed mahmoud; do
    case ${name} in
    moatasem)
        echo "hello ${name}"
        ;;
    *)
        break
        ;;
    esac
done
```

Lab:translate

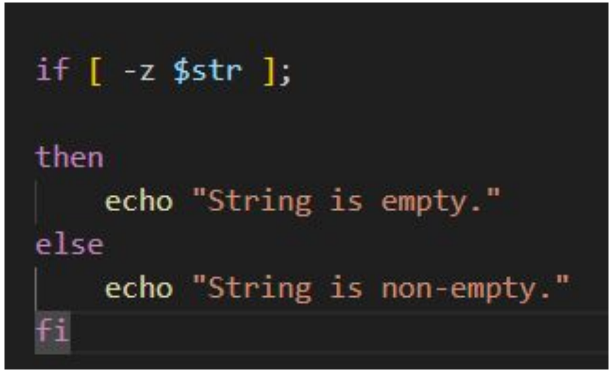

```bash
#!/usr/bin/env bash

word=$(xclip -o)
url="https://translate.google.com.eg/?sl=en&tl=ar&text=${word}&op=translate"

firefox "$url"
```

String

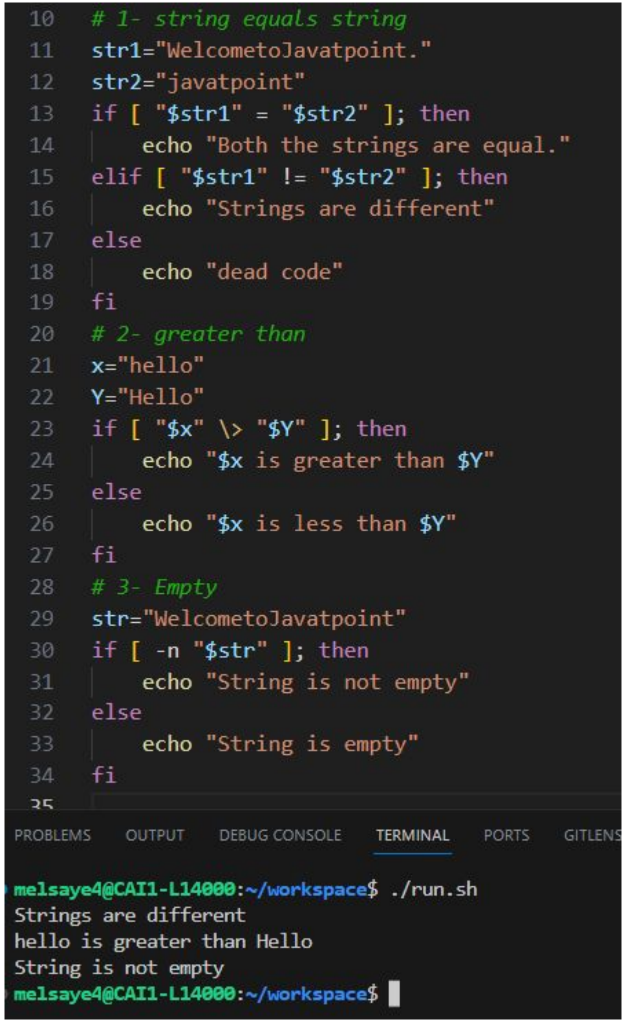
Checking on string



```
if [[ "$x" > "$Y" ]]; then
    echo "$x is greater than $Y"
else
    echo "$x is less than $Y"
fi
```



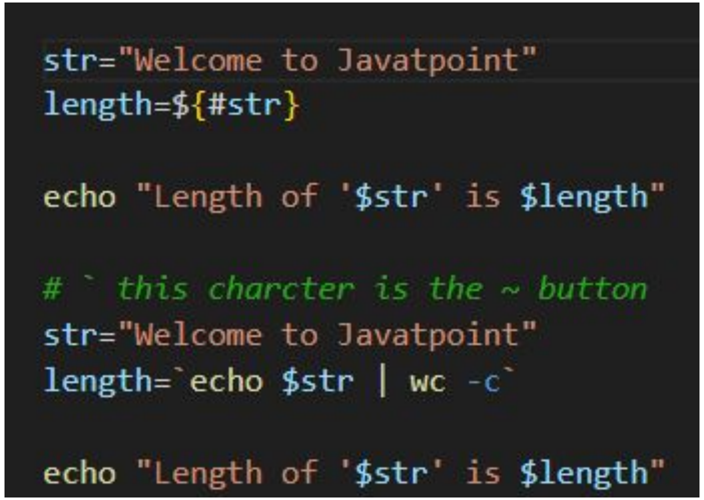
```
if [ -z $str ];

then
    echo "String is empty."
else
    echo "String is non-empty."
fi
```



```
# 1- string equals string
str1="WelcometoJavatpoint."
str2="javatpoint"
if [ "$str1" = "$str2" ]; then
    echo "Both the strings are equal."
elif [ "$str1" != "$str2" ]; then
    echo "Strings are different"
else
    echo "dead code"
fi
# 2- greater than
x="hello"
Y="Hello"
if [ "$x" \> "$Y" ]; then
    echo "$x is greater than $Y"
else
    echo "$x is less than $Y"
fi
# 3- Empty
str="WelcometoJavatpoint"
if [ -n "$str" ]; then
    echo "String is not empty"
else
    echo "String is empty"
fi
```

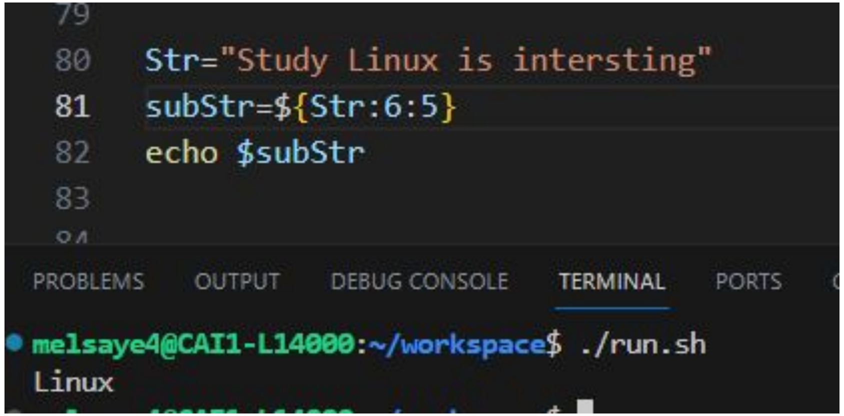
```
melsaye4@CAI1-L14000:~/workspace$ ./run.sh
Strings are different
hello is greater than Hello
String is not empty
melsaye4@CAI1-L14000:~/workspace$
```

Length



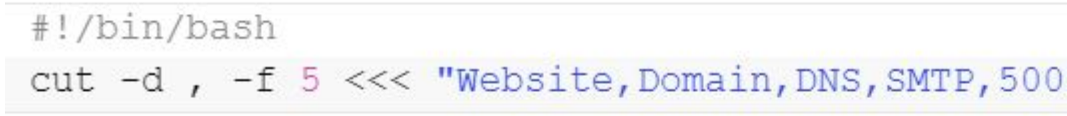
```
str="Welcome to Javatpoint"
length=${#str}

echo "Length of '$str' is $length"

# ` this charcter is the ~ button
str="Welcome to Javatpoint"
length=`echo $str | wc -c`

echo "Length of '$str' is $length"
```

split



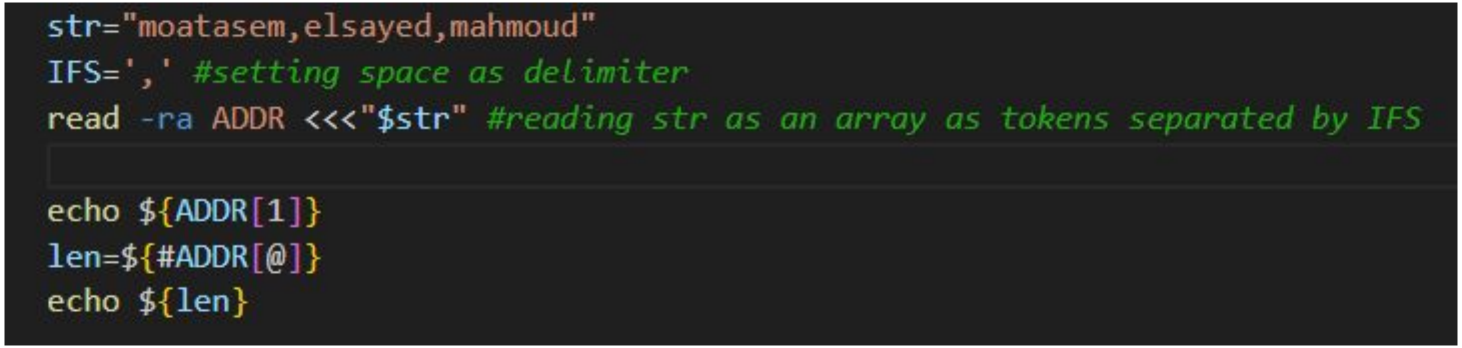
```
Str="Study Linux is intersting"
subStr=${Str:6:5}
echo $subStr
```

```
melsaye4@CAI1-L14000:~/workspace$ ./run.sh
Linux
```



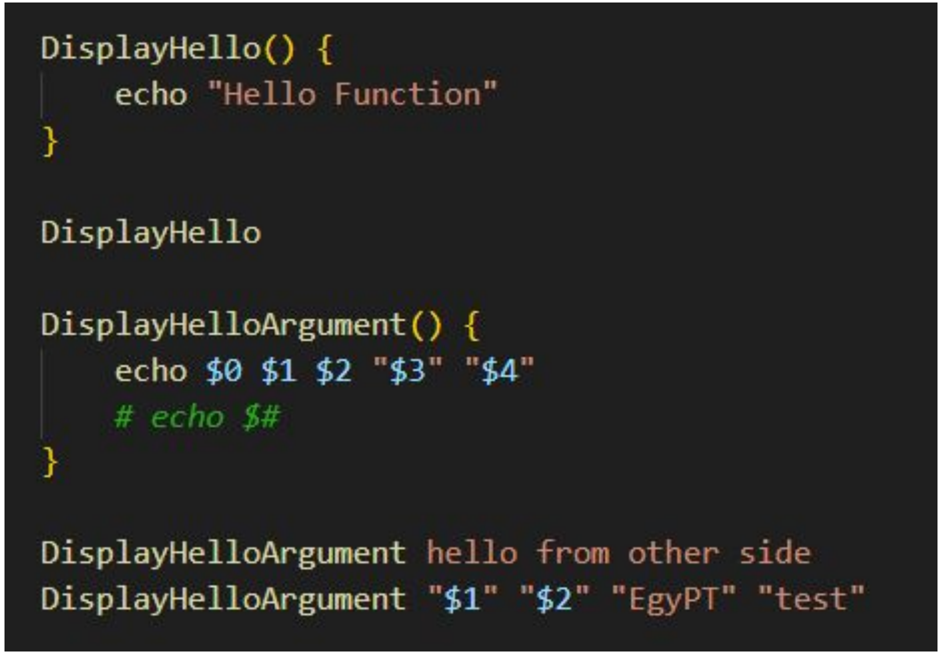
```
#!/bin/bash
cut -d , -f 5 <<< "Website,Domain,DNS,SMTP,500
```

split

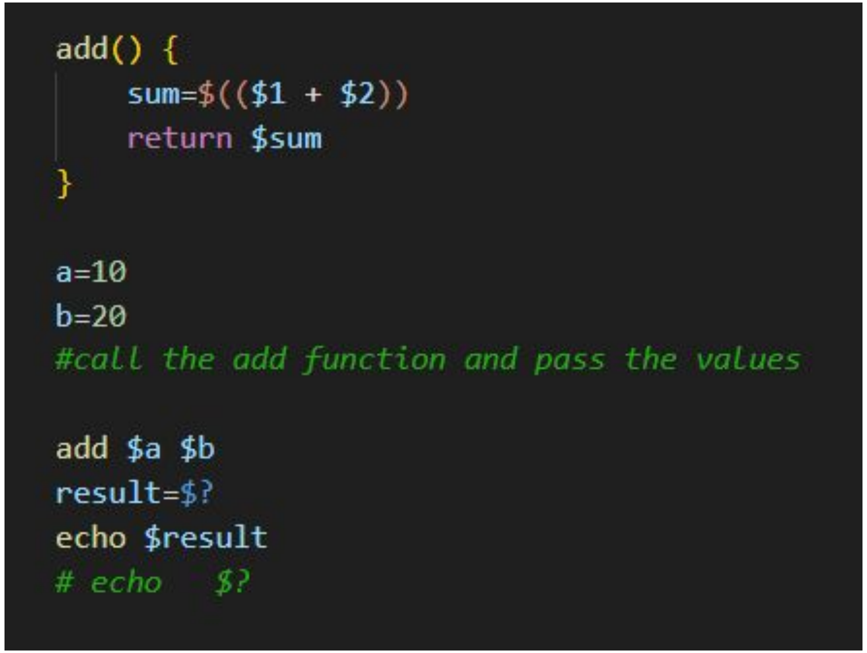


```
str="moatasem,elsayed,mahmoud"
IFS=',' #setting space as delimiter
read -ra ADDR <<<"$str" #reading str as an array as tokens separated by IFS

echo ${ADDR[1]}
len=${#ADDR[@]}
echo ${len}
```

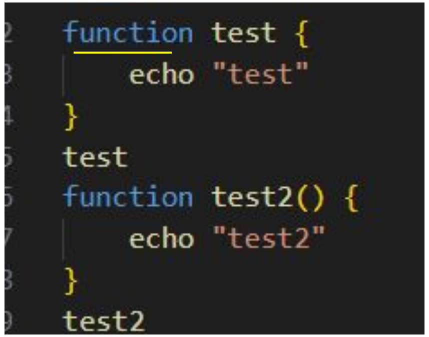
Functions



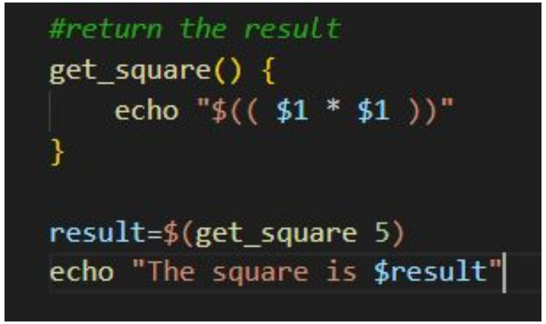
```bash
DisplayHello() {
    echo "Hello Function"
}

DisplayHello

DisplayHelloArgument() {
    echo $0 $1 $2 "$3" "$4"
    # echo $#
}

DisplayHelloArgument hello from other side
DisplayHelloArgument "$1" "$2" "EgyPT" "test"
```



```bash
add() {
    sum=$(($1 + $2))
    return $sum
}

a=10
b=20
#call the add function and pass the values

add $a $b
result=$?
echo $result
# echo   $?
```



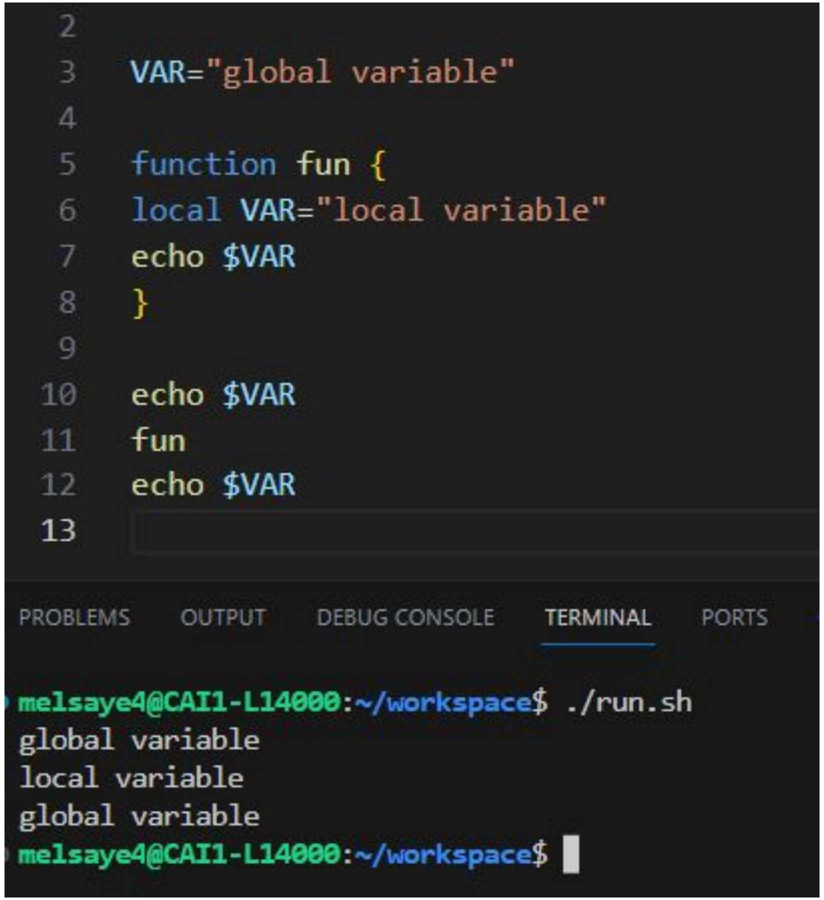
```bash
function test {
    echo "test"
}
test
function test2() {
    echo "test2"
}
test2
```



```bash
#return the result
get_square() {
    echo "$(( $1 * $1 ))"
}

result=$(get_square 5)
echo "The square is $result"
```

local, readonly,export ,unset

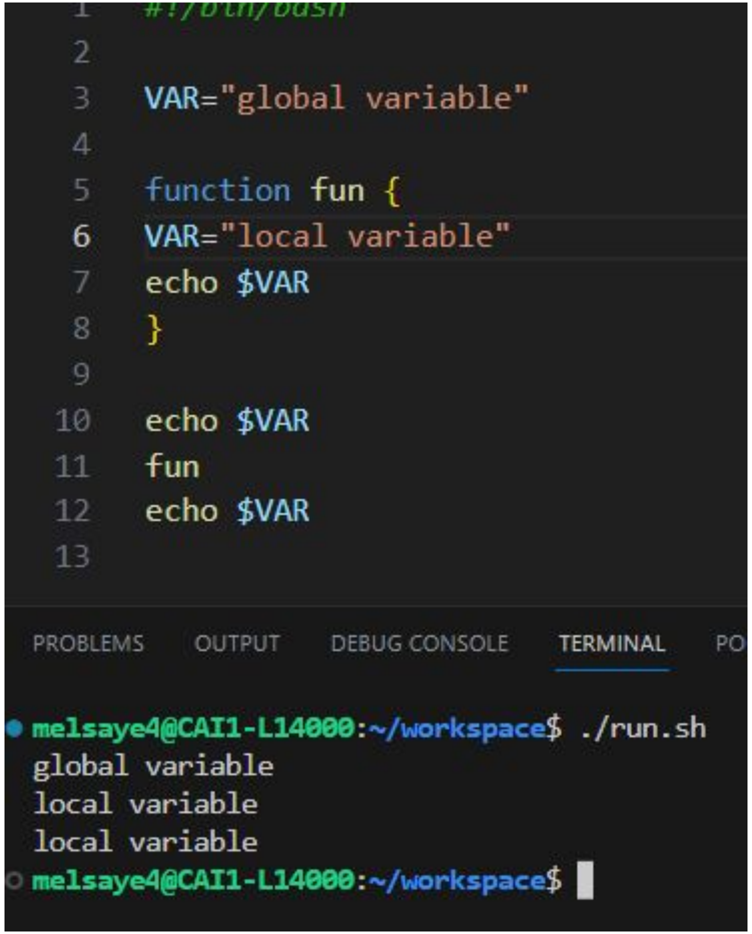
local



```
VAR="global variable"

function fun {
local VAR="local variable"
echo $VAR
}

echo $VAR
fun
echo $VAR

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
melsaye4@CAI1-L14000:~/workspace$ ./run.sh
global variable
local variable
global variable
melsaye4@CAI1-L14000:~/workspace$
```

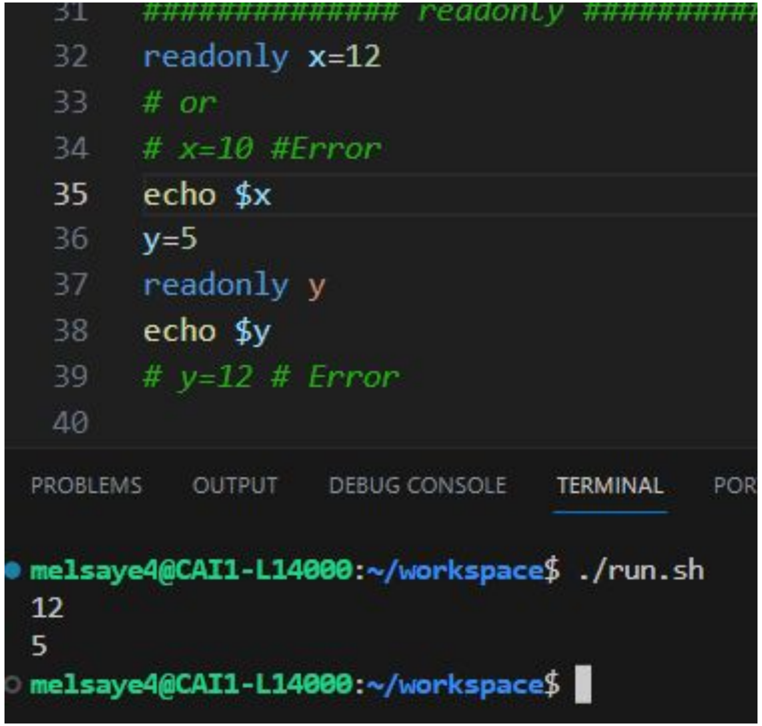


```
#!/bin/bash

VAR="global variable"

function fun {
VAR="local variable"
echo $VAR
}

echo $VAR
fun
echo $VAR

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PO

```
melsaye4@CAI1-L14000:~/workspace$ ./run.sh
global variable
local variable
local variable
melsaye4@CAI1-L14000:~/workspace$
```

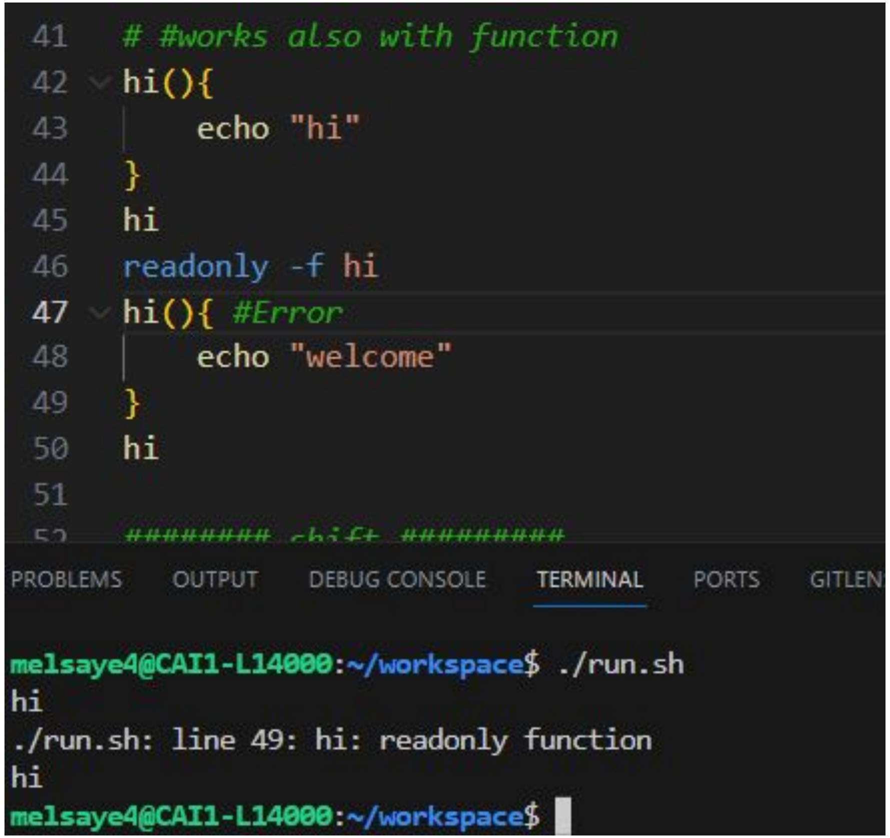
readonly



```
################ readonly ##########
readonly x=12
# or
# x=10 #Error
echo $x
y=5
readonly y
echo $y
# y=12 # Error

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL POR

```
melsaye4@CAI1-L14000:~/workspace$ ./run.sh
12
5
melsaye4@CAI1-L14000:~/workspace$
```

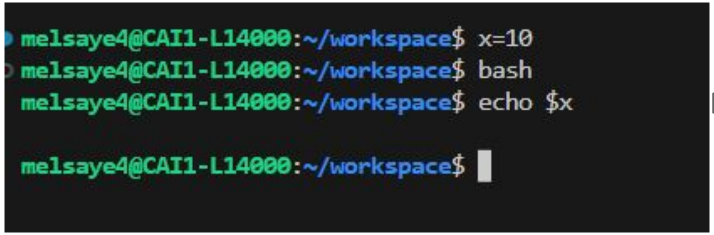


```
# #works also with function
hi(){
    echo "hi"
}
hi
readonly -f hi
hi(){ #Error
    echo "welcome"
}
hi

######### shift ##########
```

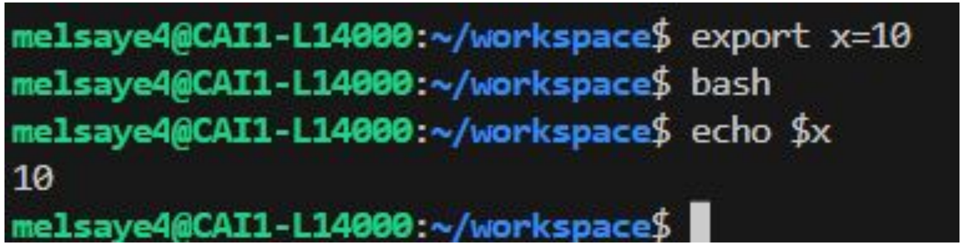
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLEN

```
melsaye4@CAI1-L14000:~/workspace$ ./run.sh
hi
./run.sh: line 49: hi: readonly function
hi
melsaye4@CAI1-L14000:~/workspace$
```

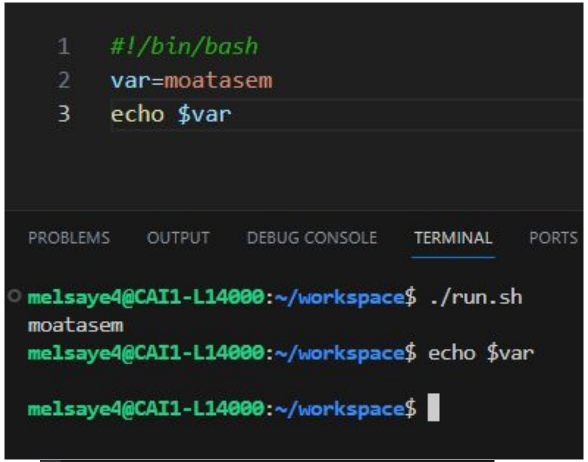
export



```
melsaye4@CAI1-L14000:~/workspace$ x=10
melsaye4@CAI1-L14000:~/workspace$ bash
melsaye4@CAI1-L14000:~/workspace$ echo $x

melsaye4@CAI1-L14000:~/workspace$ 
```

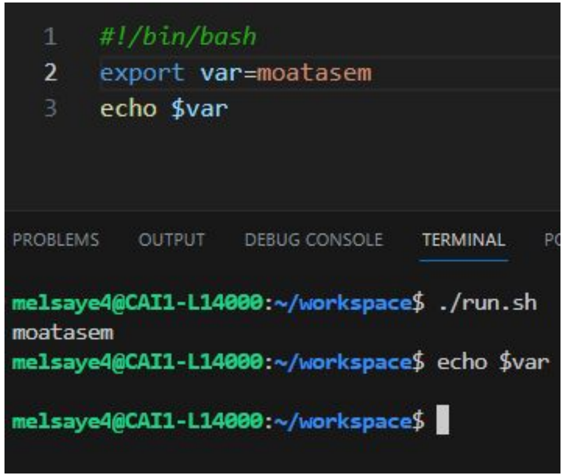


```
melsaye4@CAI1-L14000:~/workspace$ export x=10
melsaye4@CAI1-L14000:~/workspace$ bash
melsaye4@CAI1-L14000:~/workspace$ echo $x
10
melsaye4@CAI1-L14000:~/workspace$ 
```



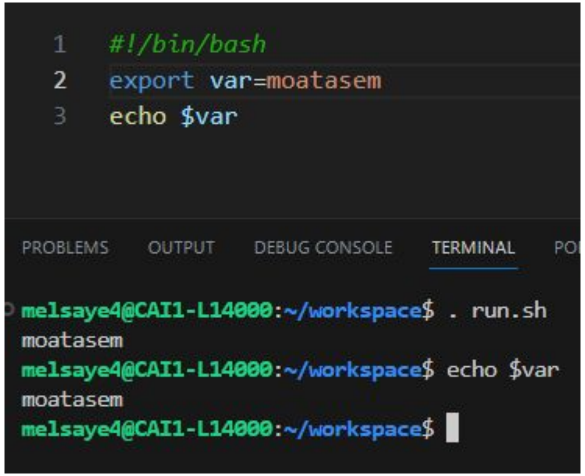
```
#!/bin/bash
var=moatasem
echo $var
```

```
melsaye4@CAI1-L14000:~/workspace$ ./run.sh
moatasem
melsaye4@CAI1-L14000:~/workspace$ echo $var

melsaye4@CAI1-L14000:~/workspace$ 
```



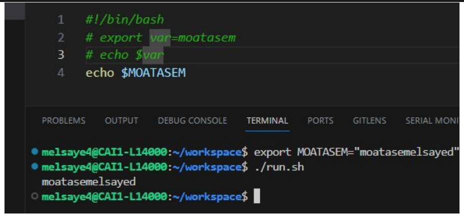
```
#!/bin/bash
export var=moatasem
echo $var
```

```
melsaye4@CAI1-L14000:~/workspace$ ./run.sh
moatasem
melsaye4@CAI1-L14000:~/workspace$ echo $var

melsaye4@CAI1-L14000:~/workspace$ 
```



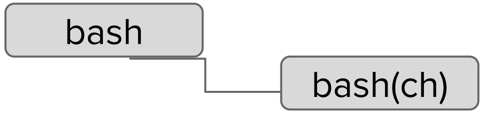
```
#!/bin/bash
export var=moatasem
echo $var
```

```
melsaye4@CAI1-L14000:~/workspace$ . run.sh
moatasem
melsaye4@CAI1-L14000:~/workspace$ echo $var
moatasem
melsaye4@CAI1-L14000:~/workspace$ 
```

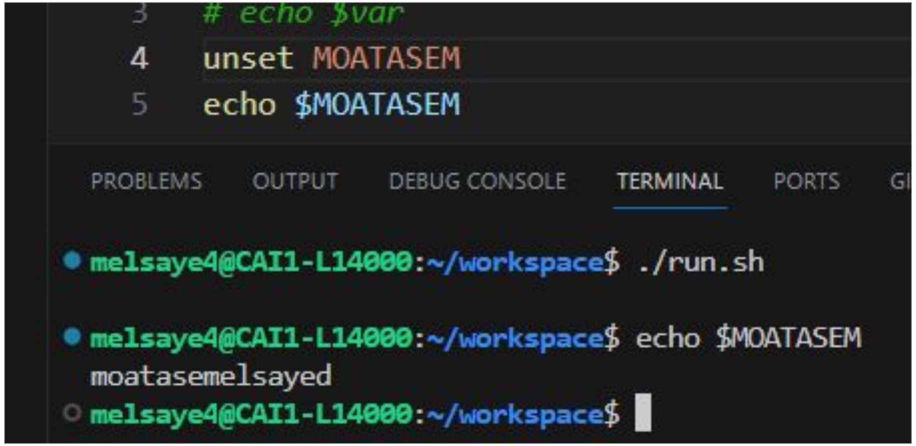


```
#!/bin/bash
# export var=moatasem
# echo $var
echo $MOATASEM
```

```
melsaye4@CAI1-L14000:~/workspace$ export MOATASEM="moatasemelsayed"
melsaye4@CAI1-L14000:~/workspace$ ./run.sh
moatasemelsayed
melsaye4@CAI1-L14000:~/workspace$ 
```



unset

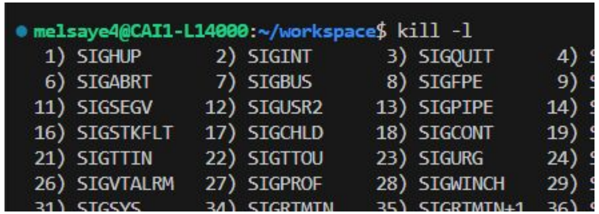


```
# echo $var
unset MOATASEM
echo $MOATASEM
```

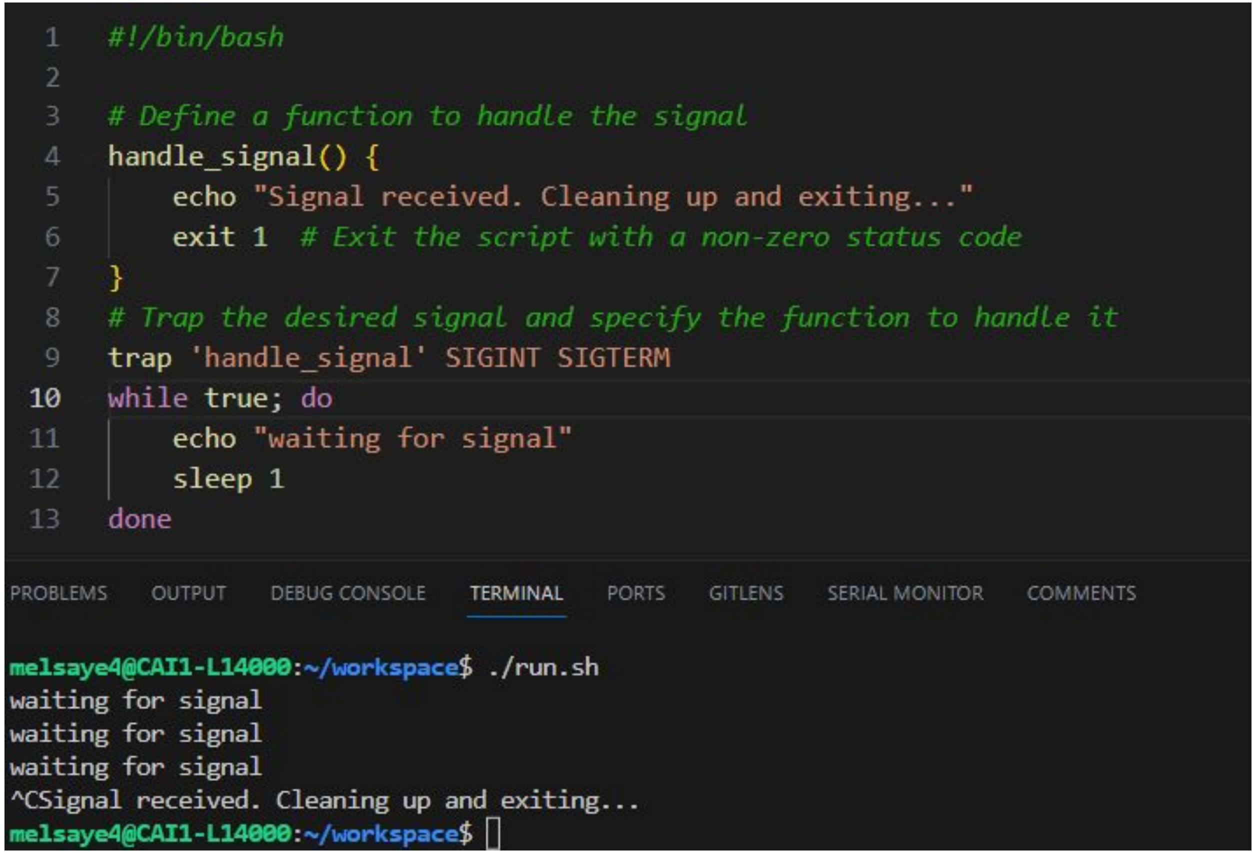
```
melsaye4@CAI1-L14000:~/workspace$ ./run.sh

melsaye4@CAI1-L14000:~/workspace$ echo $MOATASEM
moatasemelsayed
melsaye4@CAI1-L14000:~/workspace$
```

Trap

Catch signal



```
melsaye4@CAI1-L14000:~/workspace$ kill -l
 1) SIGHUP       2) SIGINT       3) SIGQUIT      4) S
 6) SIGABRT      7) SIGBUS       8) SIGFPE       9) S
11) SIGSEGV     12) SIGUSR2     13) SIGPIPE     14) S
16) SIGSTKFLT   17) SIGCHLD     18) SIGCONT     19) S
21) SIGTTIN     22) SIGTTOU     23) SIGURG      24) S
26) SIGVTALRM   27) SIGPROF     28) SIGWINCH    29) S
31) SIGSYS      34) SIGRTMIN    35) SIGRTMIN+1  36)
```

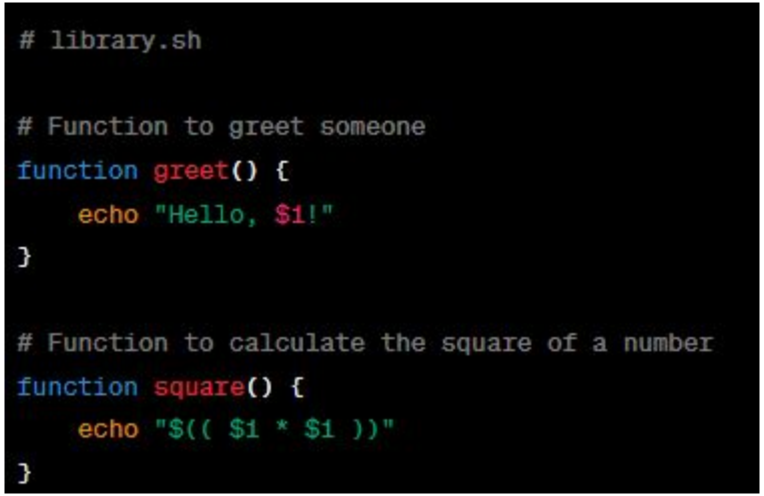


```bash
#!/bin/bash

# Define a function to handle the signal
handle_signal() {
    echo "Signal received. Cleaning up and exiting..."
    exit 1  # Exit the script with a non-zero status code
}
# Trap the desired signal and specify the function to handle it
trap 'handle_signal' SIGINT SIGTERM
while true; do
    echo "waiting for signal"
    sleep 1
done
```

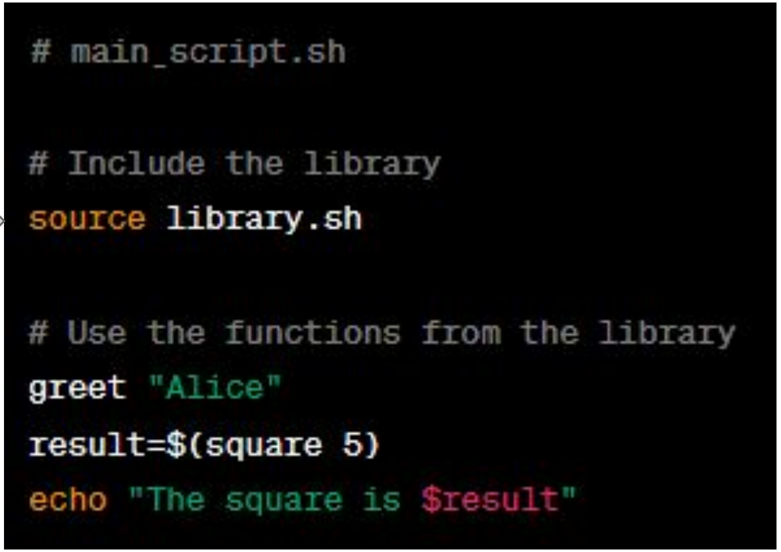
```
melsaye4@CAI1-L14000:~/workspace$ ./run.sh
waiting for signal
waiting for signal
waiting for signal
^CSignal received. Cleaning up and exiting...
melsaye4@CAI1-L14000:~/workspace$ 
```

Modularity

Modularity

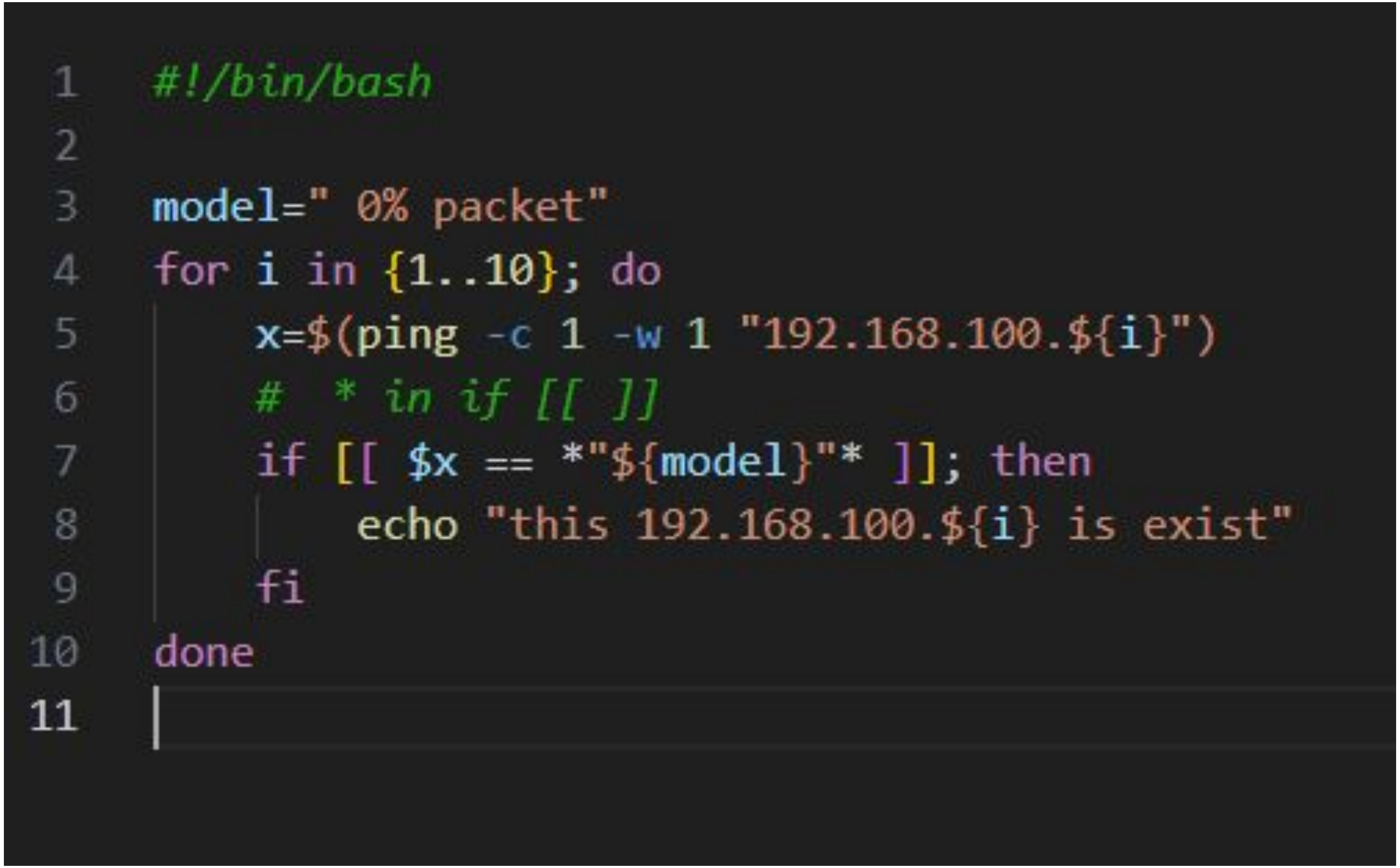


```
# library.sh

# Function to greet someone
function greet() {
    echo "Hello, $1!"
}

# Function to calculate the square of a number
function square() {
    echo "$(( $1 * $1 ))"
}
```



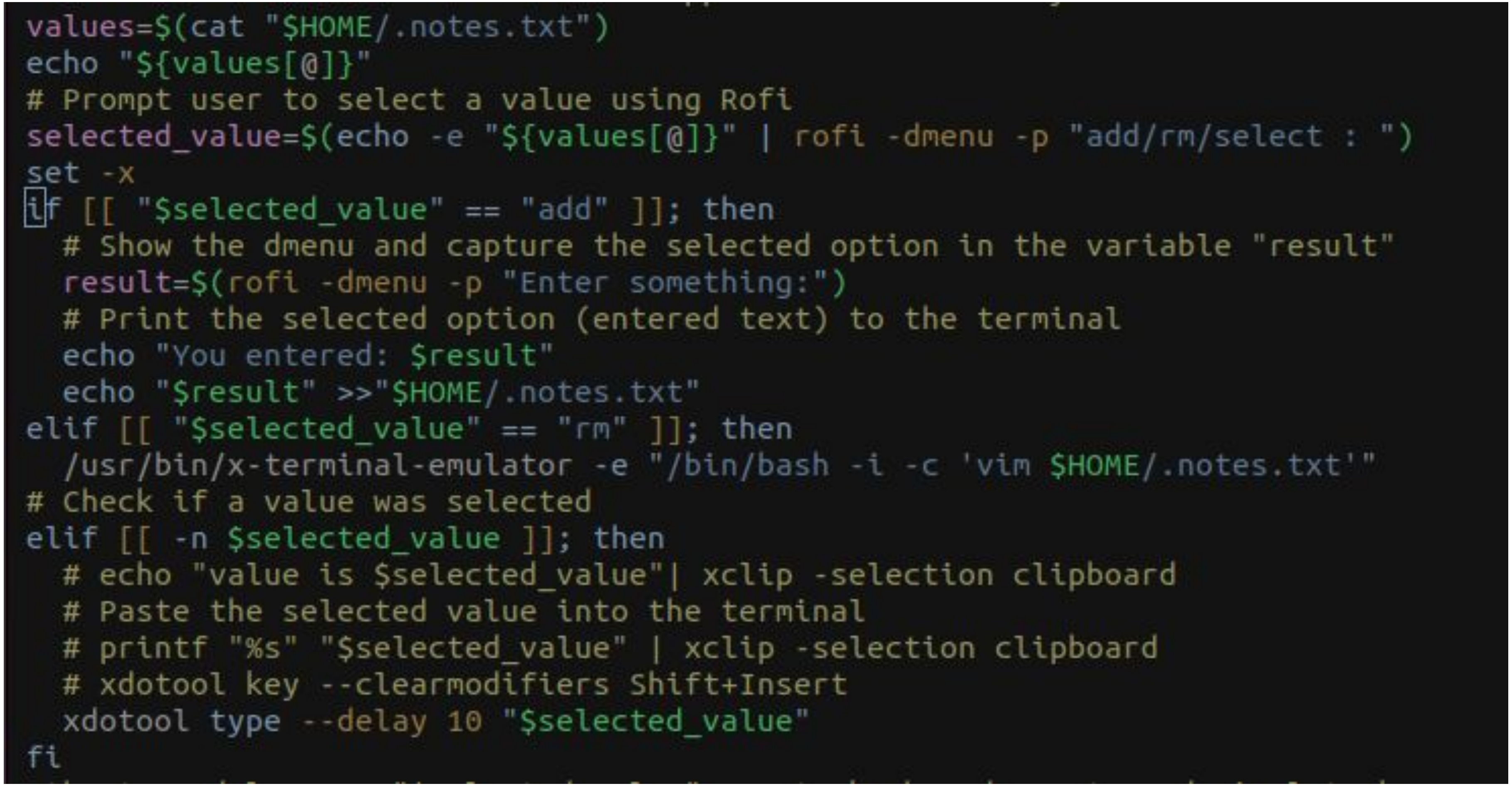
```
# main_script.sh

# Include the library
source library.sh

# Use the functions from the library
greet "Alice"
result=$(square 5)
echo "The square is $result"
```

Lab:ping Network



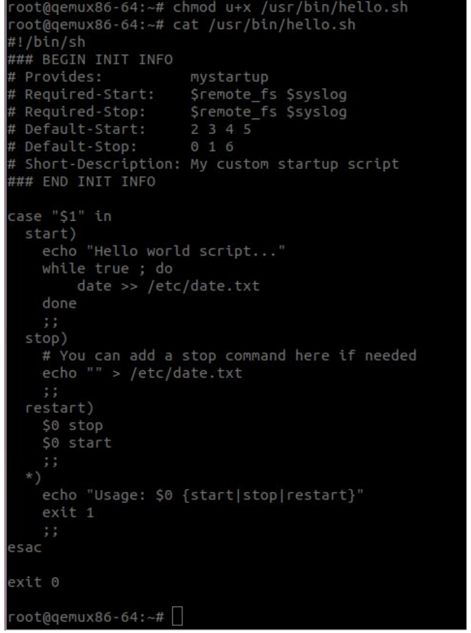
```bash
#!/bin/bash

model=" 0% packet"
for i in {1..10}; do
    x=$(ping -c 1 -w 1 "192.168.100.${i}")
    #  * in if [[ ]]
    if [[ $x == *"${model}"* ]]; then
        echo "this 192.168.100.${i} is exist"
    fi
done

```

Lab:xdotool



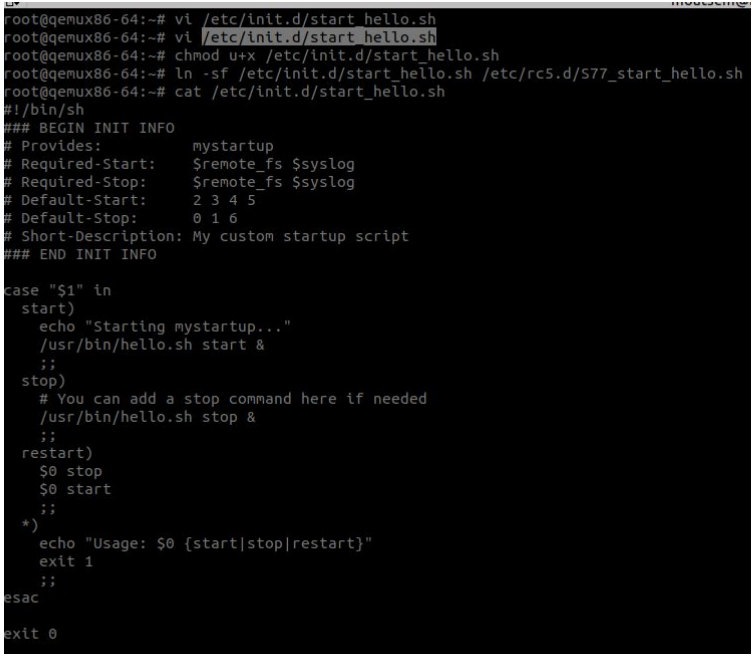
```bash
values=$(cat "$HOME/.notes.txt")
echo "${values[@]}"
# Prompt user to select a value using Rofi
selected_value=$(echo -e "${values[@]}" | rofi -dmenu -p "add/rm/select : ")
set -x
if [[ "$selected_value" == "add" ]]; then
  # Show the dmenu and capture the selected option in the variable "result"
  result=$(rofi -dmenu -p "Enter something:")
  # Print the selected option (entered text) to the terminal
  echo "You entered: $result"
  echo "$result" >>"$HOME/.notes.txt"
elif [[ "$selected_value" == "rm" ]]; then
  /usr/bin/x-terminal-emulator -e "/bin/bash -i -c 'vim $HOME/.notes.txt'"
# Check if a value was selected
elif [[ -n $selected_value ]]; then
  # echo "value is $selected_value"| xclip -selection clipboard
  # Paste the selected value into the terminal
  # printf "%s" "$selected_value" | xclip -selection clipboard
  # xdotool key --clearmodifiers Shift+Insert
  xdotool type --delay 10 "$selected_value"
fi
```

Lab:startup SV



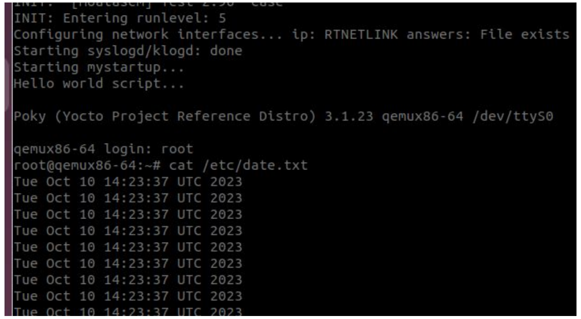
```
root@qemux86-64:~# chmod u+x /usr/bin/hello.sh
root@qemux86-64:~# cat /usr/bin/hello.sh
#!/bin/sh
### BEGIN INIT INFO
# Provides:          mystartup
# Required-Start:    $remote_fs $syslog
# Required-Stop:     $remote_fs $syslog
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: My custom startup script
### END INIT INFO

case "$1" in
  start)
    echo "Hello world script..."
    while true ; do
        date >> /etc/date.txt
    done
    ;;
  stop)
    # You can add a stop command here if needed
    echo "" > /etc/date.txt
    ;;
  restart)
    $0 stop
    $0 start
    ;;
  *)
    echo "Usage: $0 {start|stop|restart}"
    exit 1
    ;;
esac

exit 0

root@qemux86-64:~# 
```

Executable

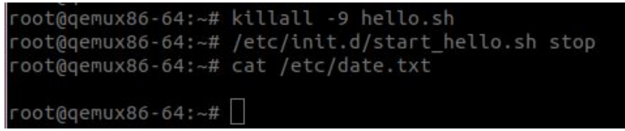


```
root@qemux86-64:~# vi /etc/init.d/start_hello.sh
root@qemux86-64:~# vi /etc/init.d/start_hello.sh
root@qemux86-64:~# chmod u+x /etc/init.d/start_hello.sh
root@qemux86-64:~# ln -sf /etc/init.d/start_hello.sh /etc/rc5.d/S77_start_hello.sh
root@qemux86-64:~# cat /etc/init.d/start_hello.sh
#!/bin/sh
### BEGIN INIT INFO
# Provides:          mystartup
# Required-Start:    $remote_fs $syslog
# Required-Stop:     $remote_fs $syslog
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: My custom startup script
### END INIT INFO

case "$1" in
  start)
    echo "Starting mystartup..."
    /usr/bin/hello.sh start &
    ;;
  stop)
    # You can add a stop command here if needed
    /usr/bin/hello.sh stop &
    ;;
  restart)
    $0 stop
    $0 start
    ;;
  *)
    echo "Usage: $0 {start|stop|restart}"
    exit 1
    ;;
esac

exit 0
```

startup



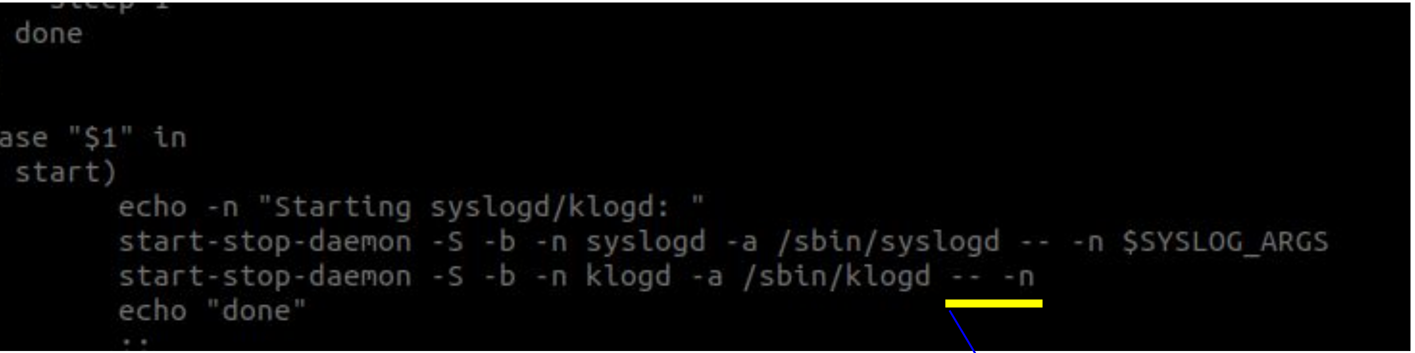
```
INIT: Entering runlevel: 5
Configuring network interfaces... ip: RTNETLINK answers: File exists
Starting syslogd/klogd: done
Starting mystartup...
Hello world script...

Poky (Yocto Project Reference Distro) 3.1.23 qemux86-64 /dev/ttyS0

qemux86-64 login: root
root@qemux86-64:~# cat /etc/date.txt
Tue Oct 10 14:23:37 UTC 2023
Tue Oct 10 14:23:37 UTC 2023
Tue Oct 10 14:23:37 UTC 2023
Tue Oct 10 14:23:37 UTC 2023
Tue Oct 10 14:23:37 UTC 2023
Tue Oct 10 14:23:37 UTC 2023
Tue Oct 10 14:23:37 UTC 2023
Tue Oct 10 14:23:37 UTC 2023
Tue Oct 10 14:23:37 UTC 2023
```



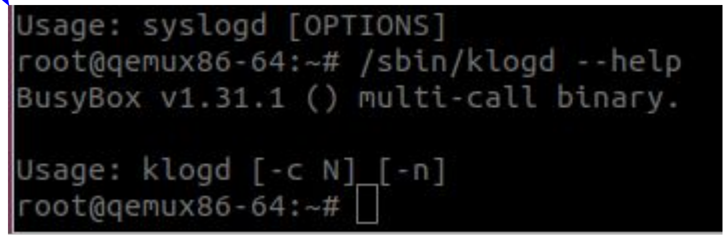
```
root@qemux86-64:~# killall -9 hello.sh
root@qemux86-64:~# /etc/init.d/start_hello.sh stop
root@qemux86-64:~# cat /etc/date.txt

root@qemux86-64:~# 
```

```
cp myscript.sh /etc/init.d/
chmod +x /etc/init.d/myscript.sh
sudo update-rc.d myscript.sh defaults
-> it will create symlink to rc3.d/
```

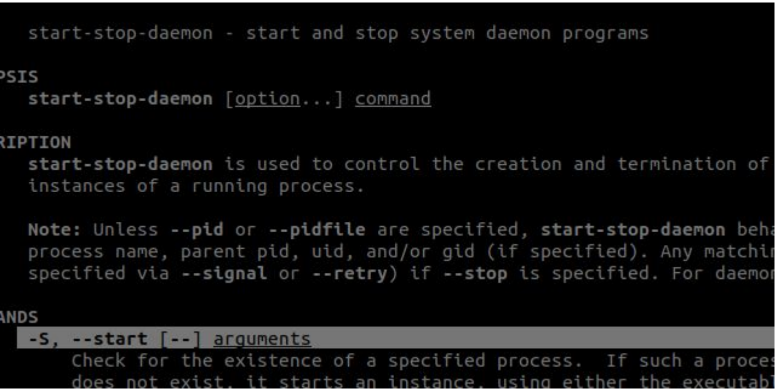
start-stop-daemon



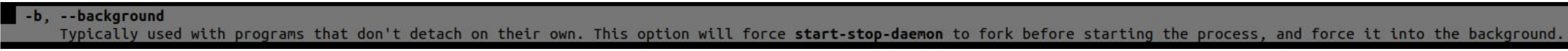
```
    sleep 1
 done

ase "$1" in
 start)
       echo -n "Starting syslogd/klogd: "
       start-stop-daemon -S -b -n syslogd -a /sbin/syslogd -- -n $SYSLOG_ARGS
       start-stop-daemon -S -b -n klogd -a /sbin/klogd -- -n
       echo "done"
       ;;
```



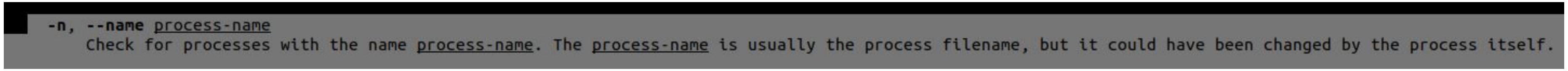
```
Usage: syslogd [OPTIONS]
root@qemux86-64:~# /sbin/klogd --help
BusyBox v1.31.1 () multi-call binary.

Usage: klogd [-c N] [-n]
root@qemux86-64:~# 
```



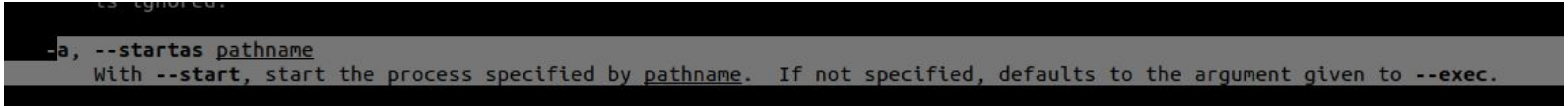
```
  start-stop-daemon - start and stop system daemon programs

PSIS
  start-stop-daemon [option...] command

RIPTION
  start-stop-daemon is used to control the creation and termination of
  instances of a running process.

  Note: Unless --pid or --pidfile are specified, start-stop-daemon beha
  process name, parent pid, uid, and/or gid (if specified). Any matchin
  specified via --signal or --retry) if --stop is specified. For daemon

ANDS
  -S, --start [--] arguments
      Check for the existence of a specified process.  If such a proces
      does not exist, it starts an instance, using either the executabl
```



```
  -b, --background
     Typically used with programs that don't detach on their own. This option will force start-stop-daemon to fork before starting the process, and force it into the background.
```



```
  -n, --name process-name
     Check for processes with the name process-name. The process-name is usually the process filename, but it could have been changed by the process itself.
```



```
     is ignored.

  -a, --startas pathname
     With --start, start the process specified by pathname.  If not specified, defaults to the argument given to --exec.
```

Tasks

1- write bash script to create project based on Makefile

2- write bash script to generate systemd service file simple example

3- write bash script to download video from youtube

4- write a wiki bash script to help you on development
- give you example about c++ hello world
- give you example about python hello world
- give you example about linux commands
- give you example about bash hello world

5- write bash script to perform calculator operations