03 Array, getopt, File

Moatasem Elsayed

Content

1- Array

2-getopt

3-File

4-systemd

Array

Accessing Array

```
## Define Array
declare -a ARRAY_NAME2=("Java" "Python" "HTML" "CSS" "JavaScript")
ARRAY_NAME2=(3 "Java" "Python" "HTML" "CSS")
ARRAY_NAME2[0]="cpp"

#Indexing
# ${ARRAY_NAME[index]} // Get Value
# ${ARRAY_NAME[@]}  //All Elements
# ${#ARRAY_NAME2[@]} //Length

# echo ${ARRAY_NAME2[0]}
echo "${ARRAY_NAME2[@]}"                                #cpp Java Python HTML CSS
echo "The array contains ${#ARRAY_NAME2[@]} elements" #5
```

Delete Array

```
#cpp Java Python HTML CSS
# delete
# unset "ARRAY_NAME[index]"
unset "ARRAY_NAME2[2]"
echo "${ARRAY_NAME2[@]}" #cpp Java HTML CSS
echo "The array contains ${#ARRAY_NAME2[@]} elements"
echo "${ARRAY_NAME2[3]}"
```

```
moatsem@moatsem-IdeaPad-Gaming-3-15IA
cpp Java HTML CSS
The array contains 4 elements
HTML
```

Slicing

```
25  ## Slicing
26  SLICED_ARRAY=("${ARRAY_NAME2[@]:1:3}") #e included
27  echo "${SLICED_ARRAY[@]}"
28  example_array=("Java" 2 "HTML" "CSS" "JavaScript")
29  sliced_array=("${example_array[@]:1:3}")
30  echo "sliced array is "${sliced_array[@]}""    Double quote array expansions to avoid re-spli
31
```

```
PROBLEMS 6   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   GITLENS

moatsem@moatsem-IdeaPad-Gaming-3-15IAH7:~/Diploma/mypresetation/04bash/Bash_Basics/Basic_commands$ ./16_Array.sh
Java Python HTML
sliced array is 2 HTML CSS
moatsem@moatsem-IdeaPad-Gaming-3-15IAH7:~/Diploma/mypresetation/04bash/Bash_Basics/Basic_commands$
```

map

```
3  #declare -a arr=(BMW MERCEDES TOTYTA)
4  arr=(BMW MERCEDESE TOTYTA)
5  echo "${arr[1]}"
6
7  declare -A login=([user]=Moatsem [password]=password)
8  echo "${login[user]}"
9  echo "${login[password]}"
10
```

Looping

```
39
40  for i in "${ARRAY_NAME2[@]}"; do
41      echo "$i"
42  done
43
44  # what if i want to get index value
45  for i in "${!ARRAY_NAME2[@]}"; do
46      echo "$i - ${ARRAY_NAME2[i]}  "
47  done
48  ##############################################################################
49
```

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

```
moatsem@moatsem-IdeaPad-Gaming-3-15IAH7:~/Diploma/mypresetation/04bash/Bash_Basics/Basic_commands$ ./16_Array.sh
cpp
Java
Python
HTML
CSS
0 - cpp
1 - Java
2 - Python
3 - HTML
4 - CSS
moatsem@moatsem-IdeaPad-Gaming-3-15IAH7:~/Diploma/mypresetation/04bash/Bash_Basics/Basic_commands$
```

getopt

```
2
3  usage() { echo "Usage: $0 [-s <45|90>] [-p <string>]" 1>&2; exit 1; }
4
5  while getopts ":s:p:" o; do
6      case "${o}" in
7          s)
8              s=${OPTARG}
9              ((s == 45 || s == 90)) || usage
10             ;;
11         p)
12             p=${OPTARG}
13             ;;
14         *)
15             usage
16             ;;
17     esac
18 done
19 shift $((OPTIND-1))
20
21 if [ -z "${s}" ] || [ -z "${p}" ]; then
22     usage
23 fi
24
25 echo "s = ${s}"
26 echo "p = ${p}"
```

File

```
# read Files #1 ##

value=$(cat text.txt)
echo "$value"

value=$(<text.txt)
echo "$value"
```

```
# read File  line by line #2 ##

while read line; do    read without
    echo "$line"
done <values.txt
echo "$line"
```

```
#Write use to out >direct
echo "hello " >file.txt
```

```
#file exist
if [ -f text.txt ]; then
    echo " file exist"
fi
```

read file into array

Basic_commands > $ 22_readfile_MaxArray.sh > ...

```
1   #!/bin/bash
2   set -x
3   value=$(cat values.txt)
4   set +x
5   echo "${value[@]}"
6   declare -a ARRAY_NAME
7   index=0
8   for i in $value; do
9       ARRAY_NAME[index]=$i
10      ((index++))
11  done
12  echo "new array is ${ARRAY_NAME[1]}"
```

```
max=${ARRAY_NAME[0]}
for i in "${!ARRAY_NAME[@]}"; do
    if [ "${ARRAY_NAME[i]}" -gt "$max" ]; then
        max=${ARRAY_NAME[$i]}
    fi
done
echo "$max"
```

[[-e FILE]]	Exists
[[-r FILE]]	Readable
[[-h FILE]]	Symlink
[[-d FILE]]	Directory
[[-w FILE]]	Writable
[[-s FILE]]	Size is > 0 bytes
[[-f FILE]]	File
[[-x FILE]]	Executable

Tips

Working with dictionaries

```
echo "${sounds[dog]}" # Dog's sound
echo "${sounds[@]}"   # All values
echo "${!sounds[@]}"  # All keys
echo "${#sounds[@]}"  # Number of elements
unset sounds[dog]     # Delete dog
```

Declare integer

```
declare -i count # Declare as type integer
count+=1         # Increment
count=count+10    Use $((..)) for arithmetics, e.g. i=$((i + 2))
echo "$count" #11
```

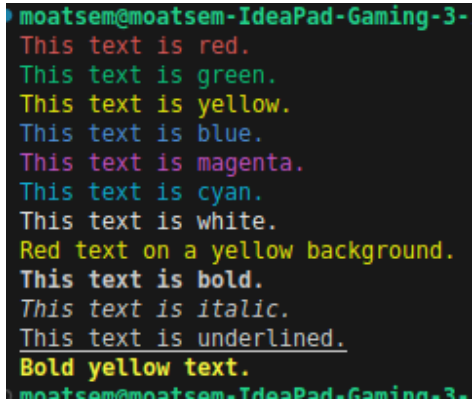
Python inside script

```
python3 -c "print(\"hello\")
print(\"world\")
"
```

Change color of text

To change the color of the text, you can use escape codes in the form of `\e[Xm`, where `X` represents the color code. Here are some common foreground text colors:

- Black: `\e[30m`
- Red: `\e[31m`
- Green: `\e[32m`
- Yellow: `\e[33m`
- Blue: `\e[34m`
- Magenta: `\e[35m`
- Cyan: `\e[36m`
- White: `\e[37m`




```bash
#!/bin/bash

# Define color codes
RED="\e[31m"
GREEN="\e[32m"
YELLOW="\e[33m"
BLUE="\e[34m"
MAGENTA="\e[35m"
CYAN="\e[36m"
WHITE="\e[37m"
RESET="\e[0m"

# Display colored text
echo -e "${RED}This text is red.${RESET}"
echo -e "${GREEN}This text is green.${RESET}"
echo -e "${YELLOW}This text is yellow.${RESET}"
echo -e "${BLUE}This text is blue.${RESET}"
echo -e "${MAGENTA}This text is magenta.${RESET}"
echo -e "${CYAN}This text is cyan.${RESET}"
echo -e "${WHITE}This text is white.${RESET}"

# You can also apply background colors
echo -e "${RED}${YELLOW}Red text on a yellow background.${RESET}"

# Additional formatting
BOLD="\e[1m"
ITALIC="\e[3m"
UNDERLINE="\e[4m"

# Apply formatting
echo -e "${BOLD}This text is bold.${RESET}"
echo -e "${ITALIC}This text is italic.${RESET}"
echo -e "${UNDERLINE}This text is underlined.${RESET}"

# Combined formatting and color
echo -e "${BOLD}${YELLOW}Bold yellow text.${RESET}"
```

Lab:Colors

```
> hoalsem > scripts > dmenu_based > $ color_code.sh > ...
#!/bin/bash

# Create an array of ANSI color codes and color names
declare -A colors=([Red]="\e[31m"
    [Green]="\e[32m"
    [Yellow]="\e[33m"
    [Blue]="\e[34m"
    [Magenta]="\e[35m"
    [Cyan]="\e[36m"
    [White]=:"\e[37m"
    [BOLD]="\e[1m"
    [ITALIC]="\e[3m"
    [UNDERLINE]="\e[4m"
    [RESET]="\e[0m"
)
color_names=("Red" "Green" "Yellow" "Blue" "Magenta" "Cyan" "White" "BOLD" "ITALIC" "UNDERLINE" "RESET")

# Use rofi -dmenu to select a color
selected_color=$(printf "%s\n" "${color_names[@]}" | rofi -dmenu -p "Select a color:")

# echo "${colors[$selected_color]}" | xclip -sel clip
xdotool type -delay 10 "${colors[$selected_color]}"
```

Startup : systemd

```
moatsem@moatsem-IdeaPad-Gaming-3-15IAH7:~/Diploma/mypresetation/04bash/startup/sd$ sudo vim /usr/lib/systemd/system/hello.service
moatsem@moatsem-IdeaPad-Gaming-3-15IAH7:~/Diploma/mypresetation/04bash/startup/sd$ sudo systemctl enable hello
Created symlink /etc/systemd/system/multi-user.target.wants/hello.service → /lib/systemd/system/hello.service.
moatsem@moatsem-IdeaPad-Gaming-3-15IAH7:~/Diploma/mypresetation/04bash/startup/sd$ sudo systemctl start  hello
moatsem@moatsem-IdeaPad-Gaming-3-15IAH7:~/Diploma/mypresetation/04bash/startup/sd$ cat hello.sh `
> ^C
moatsem@moatsem-IdeaPad-Gaming-3-15IAH7:~/Diploma/mypresetation/04bash/startup/sd$ cat hello.sh
#!/bin/bash

date > /home/moatsem/uptime.txt
moatsem@moatsem-IdeaPad-Gaming-3-15IAH7:~/Diploma/mypresetation/04bash/startup/sd$ cat /home/moatsem/uptime.txt
22 2023 ,أكّ EEST 12:00:56 ص
moatsem@moatsem-IdeaPad-Gaming-3-15IAH7:~/Diploma/mypresetation/04bash/startup/sd$ cat /usr/lib/systemd/system/hello.service
[Unit]
Description=hello wolrd

[Service]
ExecStart="/home/moatsem/Diploma/mypresetation/04bash/startup/sd/hello.sh"

[Install]
WantedBy=multi-user.target

moatsem@moatsem-IdeaPad-Gaming-3-15IAH7:~/Diploma/mypresetation/04bash/startup/sd$
```

```
/etc/systemd/system$ find /lib/systemd/system -iname default* -exec ls -l  {} \;
lib/systemd/system/default.target -> graphical.target
/etc/systemd/system$
```

[Unit]
Man systemd.unit
[Service]
Man systemd.service
[Install]

systemd targets	SystemV runlevel	target aliases	Description
default.target			This target is always aliased with a symbolic link to either **multi-user.target** or **graphical.target**. systemd always uses the **default.target** to start the system. The **default.target** should never be aliased to **halt.target**, **poweroff.target**, or **reboot.target**.
graphical.target	5	runlevel5.target	**Multi-user.target** with a GUI
	4	runlevel4.target	Unused. Runlevel 4 was identical to runlevel 3 in the SystemV world. This target could be created and customized to start local services without changing the default **multi-user.target**.
multi-user.target	3	runlevel3.target	All services running, but command-line interface (CLI) only
	2	runlevel2.target	Multi-user, without NFS, but all other non-GUI services running
rescue.target	1	runlevel1.target	A basic system, including mounting the filesystems with only the most basic services running and a rescue shell on the main console
emergency.target	S		Single-user mode—no services are running; filesystems are not mounted. This is the most basic level of operation with only an emergency shell running on the main console for the user to interact with the system.
halt.target			Halts the system without powering it down
reboot.target	6	runlevel6.target	Reboot
poweroff.target	0	runlevel0.target	Halts the system and turns the power off

commands

```
moatsem@moatsem-IdeaPad-Gaming-3-15IAH7:/etc/systemd/system$ systemctl list-dependencies pc-control.service
pc-control.service
● ├─system.slice
● └─sysinit.target
●   ├─apparmor.service
●   ├─dev-hugepages.mount
●   ├─dev-mqueue.mount
●   ├─keyboard-setup.service
●   ├─kmod-static-nodes.service
```

Application management

- `systemctl enable`
- `systemctl restart`
- `systemctl start`
- `systemctl status`
- `systemctl stop`

Control over computers and virtual machines

- `systemctl poweroff`
- `systemctl reboot`

System information

- `journalctl`
- `systemctl list-sockets`
- `systemctl list-units`
- `systemctl list-unit-files`

```
graphical.target
moatsem@moatsem-IdeaPad-Gaming-3-15IAH7:/etc/systemd/system$ systemd-analyze
Startup finished in 6.390s (firmware) + 4.547s (loader) + 4.341s (kernel) + 46.000s (userspace) = 1min 1.279s
graphical.target reached after 15.187s in userspace
moatsem@moatsem-IdeaPad-Gaming-3-15IAH7:/etc/systemd/system$
```

Cpp,Rust Lab

1 examples.sh

```
#!/bin/bash

wiki="/home/moatsem/scripts/wiki.sh"

app=$(echo -e "cpp\nrust" | rofi -dmenu -p "select app")
function cppHandler() {
    list=$(${wiki} cpp list)
    selceted=$(echo "$list" | rofi -dmenu)

    if [ -n "$selceted" ]; then
        examples=$(${wiki} cpp "$selceted")
        # echo "${examples}" >~/deletedvimfile.cpp
        echo "$examples" >/home/moatsem/deletedvimfile.cpp
        # terminator -e "/bin/bash -i -c 'cat /home/moatsem/deletedvimfile.cpp && read -p 'thank you ' test '"
        gnome-terminal --window --full-screen -- bash -c "cat /home/moatsem/deletedvimfile.cpp && read -p 'thank you' "
        rm ~/deletedvimfile.cpp
    fi
}
function rustHandler() {
    list=$(${wiki} rust list)
    selceted=$(echo "$list" | rofi -dmenu)

    if [ -n "$selceted" ]; then
        examples=$(${wiki} rust "$selceted")
        # echo "${examples}" >~/deletedvimfile.rs
        echo "$examples" >/home/moatsem/deletedvimfile.rs
        gnome-terminal --window --full-screen -- bash -c "cat /home/moatsem/deletedvimfile.rs && read -p 'thank you' "
        rm ~/deletedvimfile.rs
    fi
}
case $app in
cpp)
    cppHandler
    ;;
rust)
    rustHandler
    ;;

esac
```

~

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