A) Simple shell script.

Create a program what does following:

Hello there <your username>
Your home directory is:
Path to programs is:
The files in current folder are:
Including hidden files:
Date and time is:
Thanks so far. Bye

Program will use shell variables and commands to show content.

```
:~$ mkdir bin
:~$ cd bin
:~/bin$ nano hello_there
```

```
#!/bin/bash

echo “Hello there $USER”
echo “Your home directory is: $HOME”
echo “Current search path is: $PATH”
echo “Here are files from current folder:”
pwd | ls
echo “Files including hidden files and folders:”
pwd | ls -a
echo “Date and time: $(date)”
echo “Thanks so far. Bye”
```

```
ctrl^x
:~/bin$ bash hello_there
```

*** here comes output

```
:~/bin$ chmod a+x hello_there
```

- adding execute permission to file, so we don’t need to use bash before command

We can run this command from another place:

```
:/$ ~/bin/hello_there
```

*** here comes output

We can also add this new folder to environmental variable PATH:

```
:~$ PATH=$PATH”:/home/ik0v/bin”
```

And run this command without using absolute path:

```
:/$ hello_there
```

*** here comes output

B) Standard variables in shell.

Crate a shell programm called revrs. Program shall take 3 ingoing parameters and then write them out in opposite sequence.
Example: with command reverse ole dole doffen result is: doffen dole ole

```
:~/bin$ nano revrs

#!/bin/bash

echo $3 $2 $1

ctrl^x

:~/bin$ cd ..
:~$ revrs ole dole doffen
doffen dole ole
```

C) Command distribution.

Use command date and check what it prints out. Try also option +%F, and see what is result. Create a file with touch command that contains year, month and date as a part of file name. Check if file was created with that name.

```
:~$ date
Tue 03 Nov 2020 08:54:01 PM CEST

:~$ date +%F
2020-11-03

:~$ touch f_created_$(date +%F)
:~$ ls
… f_created_2020-11-03 …
```

Go to a subfolder and add this path to a PATH variable by using pwd. Make sure that value of PATH was indeed changed.

```
:~/Desktop/bin$ PATH=$PATH:$(pwd)
:~/bin$ echo $PATH
....... :/home/ik0v/Desktop/bin
```

Try command touch myfile’uname -r’.kernel. It creates a file. How looks this file name? Try command uname -r. Check files in /boot folder. Can you find a connection with uname -r command?

```
:~/bin$ touch myfile$(uname -r).kernel
:~/bin$ ls
… myfile4.19.0-12-amd64.kernel   - that is a kernel release number.
```

Inside /boot folder there are several files, each of them has 4.19.0-12-amd 64 part in file name. That is same kernel number as I get from uname -r command.

D)…more about shell scripts.

With shell script you can check whether it is file or folder by its name.
Try following script:

```
#!/bin/bash
FILENAME=~/.bashrc
if [ -f $FILENAME ]
then
   echo "yes, $FILNAVN is a file"
   echo "Here is file's content:"
   cat $FILENAME
else
   echo "file $FILNAVN wasn't found"
fi
```

Save program in file_check. Try it out. Try first a real, existing file, for example .bashrc, then try a non-existing file name. Check how program reacts on these changes.

```
:~/bin$ nano file_check
```
… - writing given code, saving file, giving it execute permission.

```
:~/bin$ file_chek
```

… - firstly we get approve about file, then cat command shows content of that file.

```
:~/bin$ nano file_check
FILENAME=~/.bbachrc
```
- changing only this line

ctrl^x – saving file

```
:~/bin$ file_check

file /home/ik0v/.bbashrc  wasn't found
```
- now else part is activated.

Create a new script using same code as sample above. A new program shall get file name from command line. Command to run program is:

```
./file_check myfile
```
- where myfile is a name of file to be checked

Now you need to check if that is a file or folder (hint: use -d instead of -f). If that was a folder list out its content.

```
:~/bin$ cp file_check file_check_org
```
- saving original version

```
:~/bin$ nano file_chek
```

Then comes code for a new script:

```
#!/bin/bash
FILENAME=$1
if [ -d $FILENAME ]
then
   echo "$FILENAME is a folder"
   echo "Content of that folder:"
   ls $FILENAME
else
   echo "$FILENAME is a file"
fi
```

```
:~/bin$ cd ..
:~$ file_check bin

bin is a folder
Content of that folder:
... file_check file_check_org ...
```

Program works fine.

E) A tiny shell script.

Try this program:

```
#!/bin/bash
configfiles=`ls /etc/*.conf`
cat $configfiles
```

Try out program, and explain what you get in result.

First line creates a variable configfiles, and gives it a value with all conf files from etc folder.
Second line prints out content from all these files with cat command, one by one.

F) Finding files created between two given dates.

Create a program that finds a files created between two given dates. Here are three lines which can be used in your script.

```
touch --date "2016-01-10" /tmp/start
touch --date "2016-01-15" /tmp/end
find /data/images -type f -newer /tmp/start -not -newer /tmp/end
```

Send output to a file.

```
:~/bin$ nano oct_files

#!/bin/bash
touch --date "2020-10-01" /tmp/start
touch --date "2020-10-31" /tmp/end

find ~/ -type f -newer /tmp/start -not -newer /tmp/end > ~/Desktop/oct_files.txt

#ctrl^x   – saving file

:~/bin$ cd
:~$ oct_files
:~$ cd Desktop
 :~/Desktop$ ls
... oct_files.txt ...   – files from home directory, created during october 2020 are saved here
```