In order to have the right to execute a file you first need to do that: chmod +x {nameOfTheFile}

First script:

```
#!/bin/bash

now="$(date)"
printf "Current date and time %s\n" "$now"

user="$(whoami)"
printf "Current user is: %s\n" "$user"

printing_working_directory="$(pwd)"
printf "Current directory is: %s\n" "$printing_working_directory"
```

On the first line: the program loader is instructed to use the /bin/bash.
On the second line we are creating a variable now=date.
On the third line we are printing the current date and time.
On the forth line we are setting the value of user to the (whoami) command which will give us the user
On the fifth line we are printing the result.
On the sixth line we are assigning the value of printing_working_directory to (pwd) which will give us the working directory
On the last line we are printing the variable.

Second script:

```
#!/bin/bash
variable="I love learning about DevOps"

printf "The variable is: %s\n" "$variable"
```

On the first line: the program loader is instructed to use the /bin/bash.
On the second line we are assigning the value “I love…” to the variable
On the third line we are printing it

Third script:

```
#!/bin/bash
input=" plan\n code\n build\n test\n release\n deploy\n"

printf "$input"
```

On the first line: the program loader is instructed to use the /bin/bash.
On the second line we are assigning the value “Plan, code….” to the variable
On the third line we are printing it

Forth script:

```
#!/bin/bash

printf "Please input a name of a file or directory: \n"
read input

if [ -f $input ];
then
printf "%s is a regular file" "$input"
elif [ -d $input ];
then
printf "%s is a directory" "$input"
else
printf "%s is another type of file" "$input"
fi

ls -l "$input"
```

On the first line: the program loader is instructed to use the /bin/bash.
On the second line we are printing to the “please input…”
On the third line we are reading the input(this is a variable)
On the forth line we are checking if the variable input is an ordinary file.
And if yes, printing it
On the elif part we are checking if the file is a directory and if yes, we are printing it, and if it’s not any of the above, we are printing “another type of file”
On the last line we are gonna list the file or directory with ls -l which will display additional information.

Fifth script:

```
#!/bin/bash

if [ $# -ne 3 ];
then
printf "Error: must provide exactly 3 arguments.\n"
exit 1
fi

printf "My nickname is: %s\n" "$1"
printf "My age is: %d\n" "$2"
printf "My full name is: %s\n" "$3"
```

To pass the arguments: ./fifthExersice.sh "Stanislav" 20 "Stanislav Nikolov"
On the first line: the program loader is instructed to use the /bin/bash.
On the second line we are we are checking if $# which reports the number of parameters passed to a script is not equal to 3, than it prints an error, and if not, then continues to the next line in which we are printing the my nickname, my age and my full name. To execute that command we will need to pass the arguments: ./fifthExersice.sh "some string here" {some number in here} "some string here"

Sixth script:

```
#!/bin/bash

printf "Please input your name \n"
read input

printf "$input"

uptime
```

On the first line: the program loader is instructed to use the /bin/bash.
Than we are printing to the user to input a name,
which we are storing in input variable with read command
than we are printing it, and the uptime command is
prints out how long the server has been running as well as other information.