

The screenshot shows a dual-pane interface of Visual Studio Code. The left pane is a terminal window displaying the output of a shell script named 'hello.sh'. The right pane is a code editor showing the source code of the same script.

Terminal Output:

```
test@test:~/Desktop$ ./hello.sh
Hello World
Our shell name is /bin/bash
Our shell version name is 4.3.11(1)-release
Our home directory is /home/test
Our current working directory is /home/test/Desktop
The name is Mark
value 10
test@test:~/Desktop$
```

Code Editor Content:

```
1 #! /bin/bash
2 # this is a comment
3 echo "Hello World" # this is also a comment
4
5 echo Our shell name is $BASH
6 echo Our shell version name is $BASH_VERSION
7 echo Our home directory is $HOME
8 echo Our current working directory is $PWD
9
10 name=Mark
11 VALUE=10
12 echo The name is $name
13 echo value $VALUE
```

At the bottom of the code editor, there are status indicators: Line 5, Column 29 (5 selected), Indentation, LF, Shell Script (Bash), and a circular icon.

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a vertical dock containing icons for various applications: a terminal, a file manager, a browser, a file viewer, a terminal, a file manager, a browser, a file viewer, a terminal, a file manager, a browser, a file viewer, and a terminal. The main window has two panes. The left pane is a terminal window titled "Terminal" with the command "test@test:~/Desktop\$./hello.sh" and its output:

```
test@test:~/Desktop$ ./hello.sh
Enter name :
Max
Enterd name : Max
test@test:~/Desktop$ ./hello.sh
*Enter names :
max tom john
Names : max , tom, john
test@test:~/Desktop$ ./hello.sh
Enter names : l
```

The right pane is a Visual Studio Code editor window titled "hello.sh - Desktop - Visual Studio Code" showing the script content:

```
#!/bin/bash
echo "Enter names : "
read name1 name2 name3
echo "Names : $name1 , $name2, $name3"
```

At the bottom of the screen, there is a blue status bar displaying "Ln 3, Col 22" and "Shell Script (Bash)".

The screenshot shows a Linux desktop environment with a terminal window and a code editor.

Terminal:

```
test@test:~/Desktop$ ./hello.sh
Enter name :
Max
Enterd name : Max
test@test:~/Desktop$ ./hello.sh
*Enter names :
max tom john
Names : max , tom, john
test@test:~/Desktop$ ./hello.sh
Enter names :
^C
test@test:~/Desktop$ ^C
test@test:~/Desktop$ ./hello.sh
username : myuser
username : myuser
test@test:~/Desktop$ ./hello.sh
username : myuser
password : username : myuser
password : 12345
test@test:~/Desktop$ ./hello.sh
username : myuser
password :
username : myuser
password : 12345
test@test:~/Desktop$
```

Visual Studio Code:

```
hello.sh
1 #!/bin/bash
2
3 read -p 'username : ' user_var
4 read -sp 'password : ' pass_var
5 echo
6 echo "username : $user_var"
7 echo "password : $pass_var"
```

Bottom status bar: Line 4, Col 32, Spaces: 4, UTF-8, LF, Shell Script (Bash)

The screenshot shows a Linux desktop environment with a terminal window and a code editor.

Terminal:

```
test@test:~/Desktop$ ./hello.sh
Enter name :
Max
Enterd name : Max
test@test:~/Desktop$ ./hello.sh
*Enter names :
max tom john
Names : max , tom, john
test@test:~/Desktop$ ./hello.sh
Enter names :
^C
test@test:~/Desktop$ ^C
test@test:~/Desktop$ ./hello.sh
username : myuser
username : myuser
test@test:~/Desktop$ ./hello.sh
username : myuser
password : username : myuser
password : 12345
test@test:~/Desktop$ ./hello.sh
username : myuser
password :
username : myuser
password : 12345
test@test:~/Desktop$ ./hello.sh
Enter names :
tom max
Names : tom, max
test@test:~/Desktop$
```

Visual Studio Code:

```
hello.sh
1 #!/bin/bash
2
3 echo "Enter names : "
4 read -a names
5 echo "Names : ${names[0]}, ${names[1]}"
```

File statistics: 0 lines, 0 characters, 0 words.

Encoding: UTF-8 LF Shell Script (Bash)

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a vertical dock containing icons for various applications, including a terminal, a file manager, a browser, and other system tools. In the center, there is a terminal window titled "Terminal" with the command "test@test: ~/Desktop\$./hello.sh" and the output "Enter name : max". To the right of the terminal is a Visual Studio Code editor window titled "hello.sh - Desktop - Visual Studio Code" displaying the following shell script:

```
#!/bin/bash
echo "Enter name : "
read
echo "Name : $REPLY"
```

The status bar at the bottom of the VS Code window indicates "Ln 5, Col 20 (5 selected) Spaces: 4 UTF-8 LF Shell Script (Bash)".

The screenshot shows a Linux desktop environment with a terminal window and a code editor.

Terminal:

```
test@test:~/Desktop$ ./hello.sh
Enter name :
Max
Enterd name : Max
test@test:~/Desktop$ ./hello.sh
*Enter names :
max tom john
Names : max , tom, john
test@test:~/Desktop$ ./hello.sh
Enter names : I
^C
test@test:~/Desktop$ ^C
test@test:~/Desktop$ ./hello.sh
username : myuser
username : myuser
test@test:~/Desktop$ ./hello.sh
username : myuser
password : username : myuser
password : 12345
test@test:~/Desktop$ ./hello.sh
username : myuser
password :
username : myuser
password : 12345
test@test:~/Desktop$
```

Visual Studio Code:

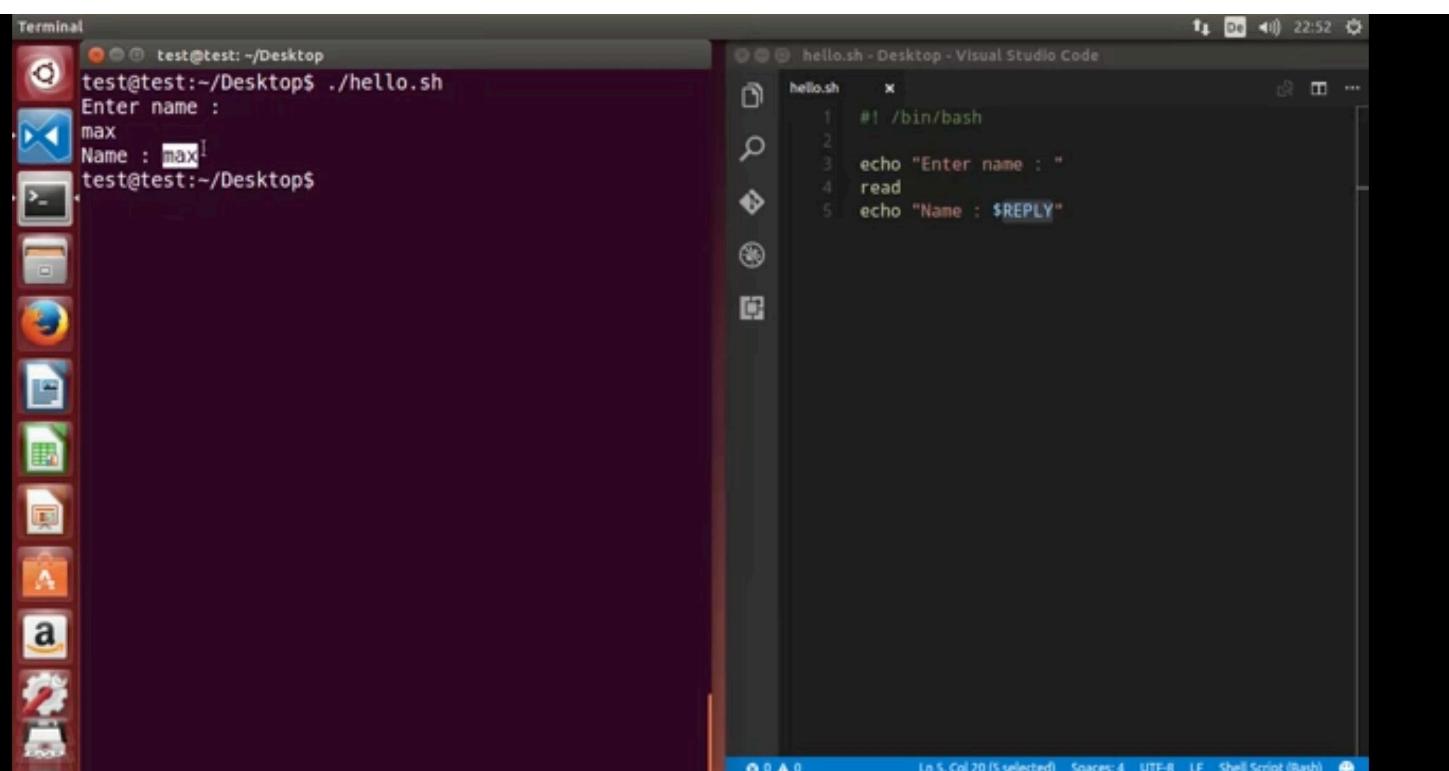
```
hello.sh
1 #!/bin/bash
2
3 read -p 'username : ' user_var
4 read -sp 'password : ' pass_var
5 echo
6 echo "username : $user_var"
7 echo "password : $pass_var"
```

Bottom status bar: Line 7, Col 28, Spaces:4, UTF-8, LF, Shell Script (Bash)

The screenshot shows a split interface of Visual Studio Code. On the left is a terminal window titled "Visual Studio Code" with the command "test@test:~/Desktop\$./hello.sh" and several test runs of the script. On the right is a code editor window titled "hello.sh - Desktop - Visual Studio Code" containing the following shell script:

```
#!/bin/bash
echo "Enter names : "
read -a names
echo "Names : ${names[0]}, ${names[1]}"
```

The terminal output shows various inputs and outputs, including "Max", "max tom john", "myuser", "password", and "tom max". The code editor shows the script's logic for reading two names from input and printing them to the console.



The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a vertical dock containing icons for various applications, including a terminal, file manager, browser, and system tools. In the center, there is a terminal window titled "Terminal" showing command-line output. To the right, there is a Visual Studio Code editor window titled "hello.sh - Desktop - Visual Studio Code" displaying a shell script named "hello.sh".

Terminal Output:

```
test@test:~/Desktop$ ./hello.sh Mark Tom John
Mark Tom John > echo $1 $2 $3
test@test:~/Desktop$ ./hello.sh Mark Tom John
./hello.sh Mark Tom John > echo $1 $2 $3
test@test:~/Desktop$
```

Visual Studio Code Editor Content:

```
hello.sh
1 #!/bin/bash
2
3 echo $0 $1 $2 $3 > echo $1 $2 $3'
```

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a vertical dock containing icons for various applications, including a terminal, file manager, and system settings. The main area features a terminal window titled "Terminal" and a Visual Studio Code editor window titled "hello.sh - Desktop - Visual Studio Code".

The terminal window displays the following command-line session:

```
test@test:~/Desktop$ ./hello.sh Mark Tom John
Mark Tom John > echo $1 $2 $3
test@test:~/Desktop$ ./hello.sh Mark Tom John
./hello.sh Mark Tom John > echo $1 $2 $3
test@test:~/Desktop$ ./hello.sh Mark Tom John
./hello.sh Mark Tom John > echo $1 $2 $3
Mark Tom John
test@test:~/Desktop$ ./hello.sh Mark Tom John
./hello.sh Mark Tom John > echo $1 $2 $3
Mark Tom John
test@test:~/Desktop$
```

The Visual Studio Code editor window shows the content of the "hello.sh" script:

```
#!/bin/bash
echo $0 $1 $2 $3 > echo $1 $2 $3'
args=("$@")
echo ${args[0]} ${args[1]} ${args[2]}
```

The status bar at the bottom of the screen indicates the following information: Line 8, Column 1, Spaces: 4, UTF-8, LF, Shell Script (Bash).

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a vertical dock containing icons for various applications, including a terminal, file manager, browser, and system tools. The main area features a terminal window titled "Terminal" and a Visual Studio Code editor window.

Terminal:

```
test@test:~/Desktop$ ./hello.sh Mark Tom John
Mark Tom John > echo $1 $2 $3
test@test:~/Desktop$ ./hello.sh Mark Tom John
./hello.sh Mark Tom John > echo $1 $2 $3
test@test:~/Desktop$ ./hello.sh Mark Tom John
./hello.sh Mark Tom John > echo $1 $2 $3
Mark Tom John
test@test:~/Desktop$ ./hello.sh Mark Tom John
./hello.sh Mark Tom John > echo $1 $2 $3
Mark Tom John
test@test:~/Desktop$ ./hello.sh Mark Tom John
./hello.sh Mark Tom John > echo $1 $2 $3
Mark Tom John
test@test:~/Desktop$
```

Visual Studio Code:

```
hello.sh  x
1  #!/bin/bash
2
3  echo $0 $1 $2 $3 > echo $1 $2 $3'
4
5  args=("$@")
6
7  #echo ${args[0]} ${args[1]} ${args[2]}
8
9  echo $@ |
```

At the bottom of the screen, there is a blue status bar displaying "Ln 9, Col 8" and "Spaces: 4" along with other file information.

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a dock containing icons for various applications, including a terminal, file manager, browser, and system tools. The main area has two windows open: a terminal window and a Visual Studio Code editor.

Terminal Window:

```
test@test:~/Desktop$ ./hello.sh Mark Tom John
Mark Tom John > echo $1 $2 $3
test@test:~/Desktop$ ./hello.sh Mark Tom John
./hello.sh Mark Tom John > echo $1 $2 $3
test@test:~/Desktop$ ./hello.sh Mark Tom John
./hello.sh Mark Tom John > echo $1 $2 $3
Mark Tom John
test@test:~/Desktop$ ./hello.sh Mark Tom John
./hello.sh Mark Tom John > echo $1 $2 $3
Mark Tom John
test@test:~/Desktop$ ./hello.sh Mark Tom John
./hello.sh Mark Tom John > echo $1 $2 $3
Mark Tom John
3
```

Visual Studio Code Editor:

```
hello.sh x
1 #!/bin/bash
2
3 echo $0 $1 $2 $3' > echo $1 $2 $3'
4
5 args=("$@")
6
7 #echo ${args[0]} ${args[1]} ${args[2]}
8
9 echo $@
10
11 echo $#
```

The status bar at the bottom of the screen indicates the following information: Line 11, Column 8, Spaces: 4, UTF-8, LF, Shell Script (Bash).

Visual Studio Code

test@test:~/Desktop\$

The screenshot shows a dark-themed desktop environment with a vertical dock of icons on the left. A terminal window is open in the top-left corner, displaying a command-line interface with the user 'test' at host 'test'. In the center, a code editor window for 'hello.sh' is open in Visual Studio Code, showing a simple shell script template. The status bar at the bottom right indicates the file has 3 lines, 0 columns, is in UTF-8 encoding, and is a Shell Script (Bash) file.

```
#!/bin/bash
if [ condition ]
then
    statement
fi
```

The image shows a dual-monitor setup displaying two code editors. On the left monitor, the Atom editor is open with a file named 'untitled'. The code in this file is a script demonstrating various bash comparison operators. On the right monitor, the Visual Studio Code editor is open with a file named 'hello.sh'. This file contains a simple shell script with a conditional statement. Both editors show syntax highlighting and line numbers.

Atom Editor (Left):

```
1 integer comparison
2
3 -eq - is equal to - if [ "$a" -eq "$b" ]
4 -ne - is not equal to - if [ "$a" -ne "$b" ]
5 -gt - is greater than - if [ "$a" -gt "$b" ]
6 -ge - is greater than or equal to - if [ "$a" -ge "$b" ]
7 -lt - is less than - if [ "$a" -lt "$b" ]
8 -le - is less than or equal to - if [ "$a" -le "$b" ]
9 < - is less than - (( "$a" < "$b" ))
10 <= - is less than or equal to - (( "$a" <= "$b" ))
11 > - is greater than - (( "$a" > "$b" ))
12 >= - is greater than or equal to - (( "$a" >= "$b" ))
13
14 string comparison
15 == - is equal to - if [ "$a" == "$b" ]
16 === - is equal to - if [ "$a" === "$b" ]
17 != - is not equal to - if [ "$a" != "$b" ]
18 <- is less than, in ASCII alphabetical order - if [[ $a < $b ]]
19 >- is greater than, in ASCII alphabetical order - if [[ $a > $b ]]
20 -z - string is null, that is, has zero length
```

Visual Studio Code Editor (Right):

```
1 #!/bin/bash
2
3 count=10
4
5 if [ $count ]
6 then
7     statement
8 fi
```

The screenshot shows two code editors side-by-side. On the left is Atom, and on the right is Visual Studio Code. Both editors are displaying shell script code.

Atom Editor (Left):

```
1 integer comparison
2
3 -eq - is equal to - if [ "$a" -eq "$b" ]
4 -ne - is not equal to - if [ "$a" -ne "$b" ]
5 -gt - is greater than - if [ "$a" -gt "$b" ]
6 -ge - is greater than or equal to - if [ "$a" -ge "$b" ]
7 -lt - is less than - if [ "$a" -lt "$b" ]
8 -le - is less than or equal to - if [ "$a" -le "$b" ]
9 < - is less than - (( "$a" < "$b" ))
10 <= - is less than or equal to - (( "$a" <= "$b" ))
11 > - is greater than - (( "$a" > "$b" ))
12 >= - is greater than or equal to - (( "$a" >= "$b" ))
13
14 string comparison
15 = - is equal to - if [ "$a" = "$b" ]
16 == - is equal to - if [ "$a" == "$b" ]
17 != - is not equal to - if [ "$a" != "$b" ]
18 < - is less than, in ASCII alphabetical order - if [[ $a < $b ]]
19 > - is greater than, in ASCII alphabetical order - if [[ $a > $b ]]
20 -z - string is null, that is, has zero length
```

Visual Studio Code Editor (Right):

```
1 #!/bin/bash
2
3 count=10
4
5 if [ $count -gt 9 ]
6 then
7     echo "condition is true"
8 fi
```

Visual Studio Code

untitled — Atom

```
1 integer comparison
2
3 -eq - is equal to - if [ "$a" -eq "$b" ]
4 -ne - is not equal to - if [ "$a" -ne "$b" ]
5 -gt - is greater than - if [ "$a" -gt "$b" ]
6 -ge - is greater than or equal to - if [ "$a" -ge "$b" ]
7 -lt - is less than - if [ "$a" -lt "$b" ]
8 -le - is less than or equal to - if [ "$a" -le "$b" ]
9 < - is less than - (( "$a" < "$b" ))
10 <= - is less than or equal to - (( "$a" <= "$b" ))
11 > - is greater than - (( "$a" > "$b" ))
12 >= - is greater than or equal to - (( "$a" >= "$b" ))
13
14 string comparison
15 = - is equal to - if [ "$a" = "$b" ]
16 == - is equal to - if [ "$a" == "$b" ]
17 != - is not equal to - if [ "$a" != "$b" ]
18 < - is less than, in ASCII alphabetical order - if [[ $a < $b ]]
19 > - is greater than, in ASCII alphabetical order - if [[ $a > $b ]]
20 -z - string is null, that is, has zero length
```

hello.sh - Desktop - Visual Studio Code

```
1 #!/bin/bash
2
3 count=10
4
5 if (($count >= 9 ))
6 then
7     echo "condition is true"
8 fi
```

The screenshot shows a Linux desktop environment with a dark theme. On the left is a dock containing icons for various applications, including a terminal, file manager, and web browser. The main area features a terminal window and a Visual Studio Code editor.

Terminal:

```
test@test:~/Desktop$ ./hello.sh
test@test:~/Desktop$ ./hello.sh
condition is true
test@test:~/Desktop$
```

Visual Studio Code:

```
hello.sh
1  #!/bin/bash
2
3  word=abc
4
5  if [ $word == "abc" ]
6  then
7      echo "condition is true"
8  fi
```

Bottom status bar: Ln 8, Col 5 Spaces: 4 UTF-8 LF Shell Script (Bash)

The screenshot shows a split-screen interface of Visual Studio Code. On the left, a terminal window displays the output of running a shell script named 'hello.sh'. The script checks if the variable \$word is equal to "abcccc". If true, it prints "condition is true". The terminal output is as follows:

```
test@test:~/Desktop$ ./hello.sh
test@test:~/Desktop$ ./hello.sh
condition is true
test@test:~/Desktop$
```

On the right, the code editor displays the content of the 'hello.sh' file. The code is a Bash script that sets a variable \$word to 'abc' and then checks if it equals 'abcccc' using an if-then-fi conditional statement.

```
hello.sh
#! /bin/bash
word=abc
if [ $word == "abcccc" ]
then
    echo "condition is true"
fi
```

The status bar at the bottom of the interface shows the file path as 'Desktop - Visual Studio Code', the line count as 'Ln 7, Col 16', the character count as 'Spaces: 4', the encoding as 'UTF-8', the line separator as 'LF', and the language as 'Shell Script (Bash)'.

The screenshot shows two tabs open in Visual Studio Code:

untitled — Atom

```
1 integer comparison
2
3 -eq - is equal to - if [ "$a" -eq "$b" ]
4 -ne - is not equal to - if [ "$a" -ne "$b" ]
5 -gt - is greater than - if [ "$a" -gt "$b" ]
6 -ge - is greater than or equal to - if [ "$a" -ge "$b" ]
7 -lt - is less than - if [ "$a" -lt "$b" ]
8 -le - is less than or equal to - if [ "$a" -le "$b" ]
9 < - is less than - (( "$a" < "$b" ))
10 <= - is less than or equal to - (( "$a" <= "$b" ))
11 > - is greater than - (( "$a" > "$b" ))
12 >= - is greater than or equal to - (( "$a" >= "$b" ))
13
14 string comparison
15 = - is equal to - if [ "$a" = "$b" ]
16 == - is equal to - if [ "$a" == "$b" ]
17 != - is not equal to - if [ "$a" != "$b" ]
18 <- is less than, in ASCII alphabetical order - if [[ $a < $b ]]
19 >- is greater than, in ASCII alphabetical order - if [[ $a > $b ]]
20 -z - string is null, that is, has zero length
```

hello.sh - Desktop - Visual Studio Code

```
1 #!/bin/bash
2
3 word=a
4
5 if [[ $word < "b" ]]
6 then
7     echo "condition is true"
8 fi
```

Visual Studio Code

untitled — Atom

```
1 integer comparison
2
3 -eq - is equal to - if [ "$a" -eq "$b" ]
4 -ne - is not equal to - if [ "$a" -ne "$b" ]
5 -gt - is greater than - if [ "$a" -gt "$b" ]
6 -ge - is greater than or equal to - if [ "$a" -ge "$b" ]
7 -lt - is less than - if [ "$a" -lt "$b" ]
8 -le - is less than or equal to - if [ "$a" -le "$b" ]
9 < - is less than - (( "$a" < "$b" ))
10 <= - is less than or equal to - (( "$a" <= "$b" ))
11 > - is greater than - (( "$a" > "$b" ))
12 >= - is greater than or equal to - (( "$a" >= "$b" ))
13
14 string comparison
15 = - is equal to - if [ "$a" = "$b" ]
16 == - is equal to - if [ "$a" == "$b" ]
17 != - is not equal to - if [ "$a" != "$b" ]
18 < - is less than, in ASCII alphabetical order - if [[ $a < $b ]]
19 > - is greater than, in ASCII alphabetical order - if [[ $a > $b ]]
20 -z - string is null, that is, has zero length
```

hello.sh

```
1 #!/bin/bash
2
3 word=a
4
5 if [[ $word == "b" ]]
6 then
7     echo "condition is true"
8 else
9     echo "condition is false"
10 fi
```

The screenshot shows a Linux desktop environment with a terminal window and a code editor window.

Terminal:

```
test@test:~/Desktop$ ./hello.sh
test@test:~/Desktop$ ./hello.sh
condition is true
test@test:~/Desktop$ ./hello.sh
./hello.sh: line 5: b: No such file or directory
test@test:~/Desktop$ ./hello.sh
condition is true
test@test:~/Desktop$ ./hello.sh
condition is false
test@test:~/Desktop$
```

Code Editor:

```
hello.sh
1  #!/bin/bash
2
3  word=a
4
5  if [[ $word == "b" ]]
6  then
7      echo "condition is true"
8  else
9      echo "condition is false"
10 fi
```

The screenshot shows a dual-pane interface of Visual Studio Code. The left pane is a terminal window displaying a series of command-line executions of a shell script named 'hello.sh'. The right pane is a code editor showing the script's source code.

Terminal Output:

```
test@test:~/Desktop$ ./hello.sh
test@test:~/Desktop$ ./hello.sh
condition is true
test@test:~/Desktop$ ./hello.sh
./hello.sh: line 5: b: No such file or directory
test@test:~/Desktop$ ./hello.sh
condition is true
test@test:~/Desktop$ ./hello.sh
condition is false
test@test:~/Desktop$ ./hello.sh
condition a is true
test@test:~/Desktop$
```

Code Editor Content:

```
hello.sh
1  #!/bin/bash
2
3  word=a
4
5  if [[ $word == "b" ]]
6  then
7      echo "condition b is true"
8  elif [[ $word == "a" ]]
9  then
10     echo "condition a is true"
11 else
12     echo "condition is false"
13 fi
```

Bottom status bar: Ln 10, Col 3 (27 selected) Spaces: 4 UTF-8 LF Shell Script (Bash)

The screenshot shows a split interface of Visual Studio Code. On the left is a terminal window displaying a Linux shell session. On the right is a code editor window showing a Bash script.

Terminal (Left):

```
drwxrwxr-x 2 test test 4096 Mär 11 14:53 dir
-rw-rw-r-- 1 test test 820 Mär 6 21:21 doc.md
-rwxrwxr-x 1 test test 166 Mär 11 14:55 hello.sh
-rw-rw-r-- 1 test test 0 Mär 11 14:50 test
test@test:~/Desktop$ cat > test
I will write something here
test@test:~/Desktop$ ls -l
total 16
-rw-rw-r-- 1 test test 0 Mär 6 21:39 9
drwxrwxr-x 2 test test 4096 Mär 11 14:53 dir
-rw-rw-r-- 1 test test 820 Mär 6 21:21 doc.md
-rwxrwxr-x 1 test test 166 Mär 11 14:55 hello.sh
-rw-rw-r-- 1 test test 28 Mär 11 14:56 test
test@test:~/Desktop$ ./hello.sh
Enter the name of the file : test
test not empty
test@test:~/Desktop$
```

Code Editor (Right):

```
hello.sh
1 #!/bin/bash
2
3 echo -e "Enter the name of the file : \c"
4 read file_name
5
6 if [ -w $file_name ]
7 then
8     echo "$file_name not empty"
9 else
10    echo "$file_name empty"
11 fi
```

Bottom status bar: Ln 6, Col 8 Spaces: 2 UTF-8 LF Shell Script (Bash)

The image shows a dual-pane interface on an Ubuntu desktop. On the left is a terminal window titled "Terminal" with the command "test@test:~/Desktop\$./hello.sh" and its output: "Enter the name of the file : cbbccb", "cbbccb not found", "test@test:~/Desktop\$ touch test", "test@test:~/Desktop\$./hello.sh", "Enter the name of the file : test", "test found", "test@test:~/Desktop\$./hello.sh", "Enter the name of the file : test", "test found", and "test@test:~/Desktop\$". On the right is a Visual Studio Code editor window titled "hello.sh - Desktop - Visual Studio Code" containing the following shell script:

```
1 #!/bin/bash
2
3 echo -e "Enter the name of the file : \c"
4 read file_name
5
6 if [ -f $file_name ]
7 then
8     echo "$file_name found"
9 else
10    echo "$file_name not found"
11
12 fi
```

The status bar at the bottom of the VS Code window indicates "Ln 6, Col 8" and "Spaces: 2, LF, Shell Script (Bash)".

Visual Studio Code

test@test:~/Desktop\$./hello.sh
Enter the name of the file : test
.Type some text data. To quit press ctrl+d.
lets write some data. Hello worldtest@
test@test:~/Desktop\$
test@test:~/Desktop\$ cat test
lets write some data. Hello worldtest@
test@test:~/Desktop\$
test@test:~/Desktop\$./hello.sh
Enter the name of the file : test
Type some text data. To quit press ctrl+d.
Shell Scripting Tutorialtest@test:~/De

hello.sh x doc.md

```
1  #!/bin/bash
2
3  echo -e "Enter the name of the file : \c"
4  read file_name
5
6  if [ -f $file_name ]
7  then
8      if [ -w $file_name ]
9      then
10         echo "Type some text data. To quit press ctrl+d."
11         cat >> $file_name
12     else
13         echo "The file do not have write permissions"
14     fi
15 else
16     echo "$file_name not exists"
17 fi
18
```

0 ▲ 0

Ln 3, Col 32 Spaces: 2 UTF-8 LF Shell Script (Bash)

The screenshot shows a dual-pane interface of Visual Studio Code. The left pane is a terminal window displaying a Linux shell session. The right pane is a code editor showing a shell script named `hello.sh`.

Terminal Output:

```
test@test:~$ cd Desktop/
test@test:~/Desktop$ ./hello.sh
./hello.sh: line 5: [: missing `]'
age not valid
test@test:~/Desktop$ ./hello.sh
valid age
test@test:~/Desktop$ ./hello.sh
age not valid
test@test:~/Desktop$
```

Code Editor Content (`hello.sh`):

```
#!/bin/bash
age=50
if [ "$age" -gt 18 ] && [ "$age" -lt 30 ]
then
    echo "valid age"
else
    echo "age not valid"
fi
```

The screenshot shows a Visual Studio Code interface. On the left is a terminal window displaying a shell session. On the right is a code editor window showing a shell script named 'hello.sh'.

Terminal Output:

```
test@test:~$ cd Desktop/
test@test:~/Desktop$ ./hello.sh
./hello.sh: line 5: [: missing `]'
age not valid
test@test:~/Desktop$ ./hello.sh
valid age
test@test:~/Desktop$ ./hello.sh
age not valid
test@test:~/Desktop$ ./hello.sh
age not valid
test@test:~/Desktop$ ./hello.sh
valid age
test@test:~/Desktop$
```

Code Editor Content:

```
hello.sh
# /bin/bash
age=25
if [ "$age" -gt 18 -a "$age" -lt 30 ]
then
    echo "valid age"
else
    echo "age not valid"
fi
```

The screenshot shows a Linux desktop environment with a terminal window and a code editor window.

Terminal:

```
test@test:~/Desktop$ cd Desktop/
test@test:~/Desktop$ ./hello.sh
./hello.sh: line 5: [: missing `]'
age not valid
test@test:~/Desktop$ ./hello.sh
valid age
test@test:~/Desktop$ ./hello.sh
age not valid
test@test:~/Desktop$ ./hello.sh
age not valid
test@test:~/Desktop$ ./hello.sh
valid age
test@test:~/Desktop$ ./hello.sh
valid age
test@test:~/Desktop$
```

Visual Studio Code:

File: hello.sh

```
1  #!/bin/bash
2
3  age=25
4
5  if [[ "$age" -gt 18 && "$age" -lt 30 ]]
6  then
7      echo "valid age"
8  else
9      echo "age not valid"
10 fi
```

Bottom status bar: In 7 Col 1% Spaces: 4 UTF-8 LF Shell Script (Bash)

The image shows a desktop environment with a terminal window and a code editor window.

Terminal:

```
test@test:~/Desktop$ ./hello.sh
./hello.sh: line 5: [: missing `]'
./hello.sh: line 5: [60: command not found
age not valid
test@test:~/Desktop$ ./hello.sh
'valid age
test@test:~/Desktop$ ./hello.sh
valid age
test@test:~/Desktop$
```

Visual Studio Code:

File hello.sh

```
#!/bin/bash
age=25
if [ "$age" -gt 18 ] || [ "$age" -lt 30 ]
then
    echo "valid age"
else
    echo "age not valid"
fi
```

Line 3, Col 7 Spaces: 2 UTF-8 LF Shell Script (Bash)

A screenshot of a Linux desktop environment. On the left, there is a vertical dock with various application icons. In the center, there is a terminal window titled "Visual Studio Code" showing a command-line session:

```
test@test:~/Desktop$ ./hello.sh
1+1
2
test@test:~/Desktop$ ./hello.sh
25
test@test:~/Desktop$
```

To the right of the terminal is a Visual Studio Code editor window titled "hello.sh - Desktop - Visual Studio Code". It displays the following script content:

```
1 #!/bin/bash
2
3 num1=20
4 num2=5
5
6 echo $(( num1 + num2 ))
```

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a dock containing various application icons, including a terminal, file manager, and system settings. In the center, there is a terminal window titled "Terminal" showing the command-line output of running a script named "hello.sh". On the right, there is a Visual Studio Code window titled "hello.sh - Desktop - Visual Studio Code" displaying the source code of the "hello.sh" script.

Terminal Output:

```
test@test:~/Desktop$ ./hello.sh
1+1
2
test@test:~/Desktop$ ./hello.sh
25
test@test:~/Desktop$ ./hello.sh
*25
15
100
4
0
test@test:~/Desktop$
```

Visual Studio Code Content:

```
hello.sh
1 #!/bin/bash
2
3 num1=20
4 num2=5
5
6 echo $(( num1 + num2 ))
7 echo $(( num1 - num2 ))
8 echo $(( num1 * num2 ))
9 echo $(( num1 / num2 ))
10 echo $(( num1 % num2 ))
```

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a dock containing icons for various applications, including a terminal, file manager, browser, and development tools. The main area features a terminal window and a Visual Studio Code editor.

Terminal:

```
test@test:~/Desktop$ ./hello.sh
1+1
test@test:~/Desktop$ ./hello.sh
25
test@test:~/Desktop$ ./hello.sh
25
15
100
4
0
test@test:~/Desktop$ ./hello.sh
25
15
expr: syntax error
4
0
test@test:~/Desktop$ ./hello.sh
25
15
100
4
0
test@test:~/Desktop$
```

Visual Studio Code:

```
hello.sh
1 #!/bin/bash
2
3 num1=20
4 num2=5
5
6 echo $(expr $num1 + $num2 )
7 echo $(expr $num1 - $num2 )
8 echo $(expr $num1 \* $num2 )
9 echo $(expr $num1 / $num2 )
10 echo $(expr $num1 % $num2 )
```

Visual Studio Code

test@test:~/Desktop\$

The screenshot shows a dual-pane interface of Visual Studio Code. The left pane is a terminal window with a dark background and white text. It displays the command 'test@test:~/Desktop\$' followed by a cursor. The right pane is a code editor with a dark background and white text. It shows a file named 'hello.sh' containing a shell script. The script defines variables num1=20 and num2=5, and then uses echo statements to print the results of addition, subtraction, multiplication, division, and modulus operations. The status bar at the bottom indicates the file is 17 lines long, 29 columns wide, has 4 spaces per tab, is in UTF-8 encoding, and is a Shell Script (Bash). A vertical orange bar is visible between the two panes.

```
hello.sh
1 #!/bin/bash
2
3 num1=20
4 num2=5
5
6 echo $(( num1 + num2 ))
7 echo $(( num1 - num2 ))
8 echo $(( num1 * num2 ))
9 echo $(( num1 / num2 ))
10 echo $(( num1 % num2 ))
11
12
13 echo $(expr $num1 + $num2 )
14 echo $(expr $num1 - $num2 )
15 echo $(expr $num1 \* $num2 )
16 echo $(expr $num1 / $num2 )
17 echo $(expr $num1 % $num2 )
```

Ln 17, Col 29 Spaces: 4 UTF-8 LF Shell Script (Bash)

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a dock with various application icons, including a terminal, file manager, browser, and system tools. The main area has two windows open: a terminal window and a Visual Studio Code editor.

Terminal:

```
test@test:~/Desktop$ ./hello.sh
25.5
test@test:~/Desktop$ ./hello.sh
25.5
15.5
102.5
4
.5
test@test:~/Desktop$ ./hello.sh
25.5
15.5
102.5
4.10
.5
test@test:~/Desktop$
```

Visual Studio Code:

```
hello.sh
1 #!/bin/bash
2
3 num1=20.5
4 num2=5
5
6 echo "20.5+5" | bc
7 echo "20.5-5" | bc
8 echo "20.5*5" | bc
9 echo "scale=2;20.5/5" | bc
10 echo "20.5%5" | bc
11
12
```

At the bottom of the screen, there is a blue status bar displaying file information: Line 12, Col 1, Spaces: 4, UTF-8, LF, Shell Script (Bash).

The screenshot shows a Linux desktop environment with a terminal window and a code editor window.

Terminal:

```
test@test:~/Desktop$ ./hello.sh
25.5
test@test:~/Desktop$ ./hello.sh
25.5
15.5
102.5
4
.5
test@test:~/Desktop$ ./hello.sh
25.5
15.5
102.5
4.10
.5
test@test:~/Desktop$ ./hello.sh
25.5
15.5
102.5
4.10000000000000000000000000000000
.5
test@test:~/Desktop$ ./hello.sh
25.5
15.5
102.5
4.10000000000000000000000000000000
.5
test@test:~/Desktop$
```

Visual Studio Code:

```
hello.sh
1 #!/bin/bash
2
3 num1=20.5
4 num2=5
5
6 echo "$num1+$num2" | bc
7 echo "$num1-$num2" | bc
8 echo "20.5*5" | bc
9 echo "scale=20;20.5/5" | bc
10 echo "20.5%5" | bc
11
12
```

Bottom status bar: Line 7, Col 13, Spaces: 4, UTF-8, LF, Shell Script (Bash)

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a dock containing icons for various applications, including a terminal, file manager, browser, and system tools. The main area features a terminal window and a Visual Studio Code editor.

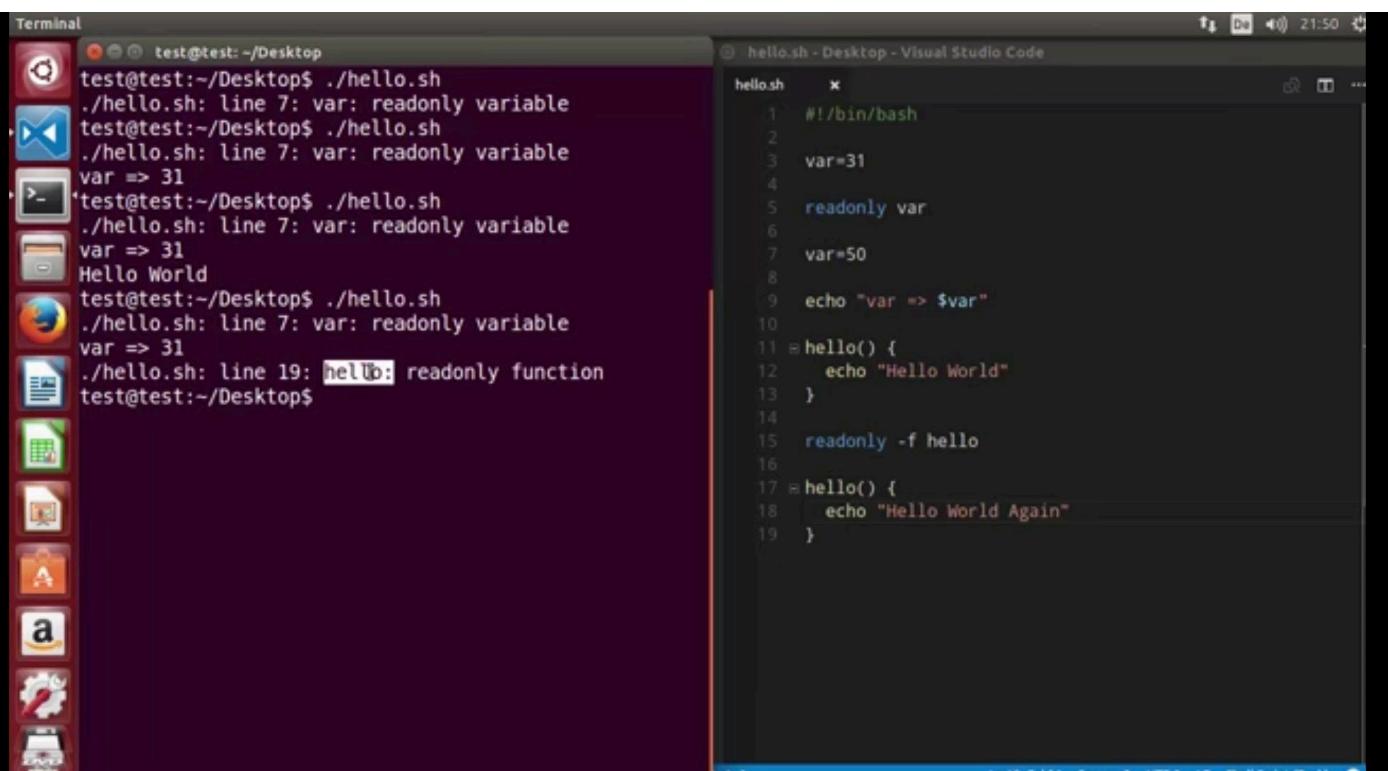
Terminal:

```
test@test:~/Desktop$ ./hello.sh
25.5
15.5
102.5
4.10000000000000000000
.5
5.19
test@test:~/Desktop$
```

Visual Studio Code:

```
hello.sh
1 #!/bin/bash
2
3 num1=20.5
4 num2=5
5
6 echo "$num1+$num2" | bc
7 echo "$num1-$num2" | bc
8 echo "20.5*5" | bc
9 echo "scale=20;20.5/5" | bc
10 echo "20.5%5" | bc
11
12 num=27
13
14 echo "scale=2;sqrt($num)" | bc -l
15
16
17
```

The terminal shows the execution of a shell script named `hello.sh`, which performs arithmetic operations using the `bc` command. The code in the editor also uses `bc` to calculate the sum, difference, product, division, and square root of the numbers 20.5 and 5. The output of the script is displayed in the terminal window.



The screenshot shows a Linux desktop environment with a terminal window and a code editor window.

Terminal:

```
test@test:~/Desktop$ ./hello.sh
25.5
15.5
102.5
4.10000000000000000000000000
.5
5.19
test@test:~/Desktop$ ./hello.sh
25.5
15.5
102.5
4.10000000000000000000000000
.5
2.00
test@test:~/Desktop$ ./hello.sh
25.5
15.5
102.5
4.10000000000000000000000000
.5
2.00
27
test@test:~/Desktop$
```

Visual Studio Code:

```
hello.sh
1 #!/bin/bash
2
3 num1=20.5
4 num2=5
5
6 echo "$num1+$num2" | bc
7 echo "$num1-$num2" | bc
8 echo "20.5*5" | bc
9 echo "scale=20;20.5/5" | bc
10 echo "20.5%5" | bc
11
12 num=4
13
14 echo "scale=2;sqrt($num)" | bc -l
15 echo "scale=2;3^3" | bc -l
16
17
18
```

File status bar: Line 17, Col 1, Spaces: 4, UTF-8, LF, Shell Script (Bash)

Visual Studio Code

test@test:~\$

The screenshot shows a dark-themed instance of Visual Studio Code. On the left is a terminal window with the command 'test@test:~\$' entered. On the right is a code editor window titled 'hello.sh - Desktop - Visual Studio Code'. The code editor displays a shell script named 'hello.sh' with the following content:

```
#!/bin/bash
case expression in
  pattern1 )
    [statements ;;
  pattern2 )
    statements ;;
...
esac
```

The status bar at the bottom of the code editor indicates the following information: Ln 5, Col 9 (13 selected), Spaces: 4, UTF-8, LF, Shell Script (Bash). The sidebar on the left contains various icons for other files and settings.

Visual Studio Code

```
test@test:~$ cd Desktop/
test@test:~/Desktop$ ./hello.sh
.Unknown vehicle
test@test:~/Desktop$ ./hello.sh car
Rent of car is 100 dollar
test@test:~/Desktop$ ./hello.sh van
Rent of van is 80 dollar
test@test:~/Desktop$ ./hello.sh truck
Rent of truck is 150 dollar
test@test:~/Desktop$
```

hello.sh - Desktop - Visual Studio Code

```
hello.sh    x
1  #! /bin/bash
2
3  vehicle=$1
4
5  case $vehicle in
6      "car" )
7          echo "Rent of $vehicle is 100 dollar" ;;
8      "van" )
9          echo "Rent of $vehicle is 80 dollar" ;;
10     "bicycle" )
11         echo "Rent of $vehicle is 5 dollar" ;;
12     "truck" )
13         echo "Rent of $vehicle is 150 dollar" ;;
14     * )
15         echo "Unknown vehicle" ;;
16 esac
17
18
19 |
```

Ln 19, Col 1 Spaces: 4 UTF-8 LF Shell Script (Bash)

The screenshot shows a desktop environment with a terminal window and a code editor window.

Terminal Window:

```
test@test:~/Desktop$ ./hello.sh
Enter some character : f
User entered f a to z
test@test:~/Desktop$ ./hello.sh
Enter some character : K
User entered K a to z
test@test:~/Desktop$ LANG=C
test@test:~/Desktop$ ./hello.sh
Enter some character : K
User entered K A to Z
test@test:~/Desktop$ ./hello.sh
Enter some character : 9
User entered 9 0 to 9
test@test:~/Desktop$ ./hello.sh
Enter some character : 5
User entered 5 0 to 9
test@test:~/Desktop$ ./hello.sh
Enter some character : &
User entered & special character
test@test:~/Desktop$ ./hello.sh
Enter some character : sdsdsdsd
Unknown input I
test@test:~/Desktop$
```

Code Editor Window (Visual Studio Code):

```
hello.sh
1 #!/bin/bash
2
3 echo -e "Enter some character : \c"
4 read value
5
6
7 case $value in
8     [a-z] )
9         echo "User entered $value a to z" ;;
10    [A-Z] )
11        echo "User entered $value A to Z" ;;
12    [0-9] )
13        echo "User entered $value 0 to 9" ;;
14    ? )
15        echo "User entered $value special character" ;;
16    * )
17        echo "Unknown input" ;;
18 esac
19
20
21
```

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a dock containing icons for various applications, including a terminal, file manager, and system tools. The main area features a terminal window and a Visual Studio Code editor.

Terminal:

```
test@test:~/Desktop$ ./hello.sh
ubuntu windows kali
test@test:~/Desktop$ ./hello.sh
ubuntu windows kali
windows
'test@test:~/Desktop$ ./hello.sh
ubuntu windows kali
ubuntu
test@test:~/Desktop$ ./hello.sh
ubuntu windows kali
ubuntu
0 1 2
test@test:~/Desktop$ ./hello.sh
ubuntu windows kali
ubuntu
0 1 2
3
test@test:~/Desktop$
```

Visual Studio Code:

```
hello.sh  x
1  #!/bin/bash
2
3  os=('ubuntu' 'windows' 'kali')
4
5  echo "${os[0]}"
6  echo "${os[0]}"
7  echo "${os[0]}"
8  echo "${#os[@]}"
9
10
```

At the bottom of the screen, there is a status bar displaying file information: "In 10, Col 1 Spaces: 4 UTF-8 LF Shell Script (Bash)".

The screenshot shows a Linux desktop environment with a terminal window and a code editor window.

Terminal:

```
test@test:~/Desktop$ ./hello.sh
ubuntu windows kali
ubuntu
test@test:~/Desktop$ ./hello.sh
ubuntu windows kali
ubuntu
0 1 2
test@test:~/Desktop$ ./hello.sh
ubuntu windows kali
ubuntu
0 1 2
3
test@test:~/Desktop$ ./hello.sh
ubuntu windows kali mac
ubuntu
0 1 2 3
4
test@test:~/Desktop$ ./hello.sh
mac windows kali
mac
0 1 2
3
test@test:~/Desktop$ ./hello.sh
ubuntu windows mac
ubuntu
0 1 3
3
test@test:~/Desktop$
```

Visual Studio Code:

```
hello.sh
1  #!/bin/bash
2
3  os=('ubuntu' 'windows' 'kali')
4  os[3]='mac'
5
6  unset os[2]
7  echo "${os[@]}"
8  echo "${os[0]}"
9  echo "${!os[@]}"
10 echo "${#os[@]}"
11
12
```

Bottom status bar: Line 7, Col 16, Spaces: 4, UTF-8, LF, Shell Script (Bash)

The screenshot shows a dual-pane interface of Visual Studio Code. The left pane is a terminal window titled "test@test:~/Desktop\$". It displays a series of identical command-line sessions, each consisting of a series of numbers (0, 1, 6) followed by the string "dasfdsafsfasdfsdf". The right pane is a code editor titled "hello.sh - Desktop - Visual Studio Code". It contains a shell script named "hello.sh" with the following content:

```
#!/bin/bash
os=('ubuntu' 'windows' 'kali')
os[6]='mac'
unset os[2]
echo "${os[@]}"
echo "${os[0]}"
echo "${!os[@]}"
echo "${#os[@]}"
string=dasfdsafsfasdfsdf
echo "${string[@]}"
echo "${string[0]}"
echo "${string[1]}"
echo "${#string[@]}"
```

The status bar at the bottom indicates "Ln 15, Col 20" and "Shell Script (Bash)".

A screenshot of a Linux desktop environment. On the left, there is a vertical dock with various application icons. In the center, there is a terminal window titled "test@test: ~/Desktop\$". To the right of the terminal is a Visual Studio Code editor window titled "hello.sh - Desktop - Visual Studio Code". The code editor shows a file named "hello.sh" with the following content:

```
1 #!/bin/bash
2 # while loops
3
4 while [ condition ]
5 do
6     command1
7     command2
8     command3
9 done
```

The status bar at the bottom of the Visual Studio Code window displays: Ln 4, Col 6 (5 selected) Spaces: 4 UTF-8 LF Shell Script (Bash)

A screenshot of the Visual Studio Code interface. On the left, there is a terminal window titled "Visual Studio Code" showing the command "test@test:~/Desktop\$./hello.sh" followed by a series of numbers from 1 to 10. On the right, there is a code editor window titled "hello.sh - Desktop - Visual Studio Code" containing the following Bash script:

```
1  #!/bin/bash
2  # while loops
3  n=1
4
5  while [ $n -le 10 ]
6  do
7      echo "$n"
8      n=$(( n+1 ))
9  done
```

The status bar at the bottom indicates "Line 3, Col 3 (1 selected)" and "Spaces: 4".

A screenshot of a Linux desktop environment. On the left, there is a dock with various icons for applications like a web browser, file manager, and system tools. In the center, there is a terminal window titled "terminal" showing the command "test@test: ~/Desktop\$./hello.sh" followed by a series of numbers from 1 to 10. To the right of the terminal is a Visual Studio Code editor window titled "hello.sh - Desktop - Visual Studio Code". The code editor shows a shell script named "hello.sh" with the following content:

```
1 #!/bin/bash
2 # while loops
3 n=1
4
5 while [ $n -le 10 ]
6 do
7     echo "$n"
8     (( n++ ))
9 done
```

The status bar at the bottom of the screen indicates "Ln 8, Col 14 Spaces: 4 UTF-8 LF Shell Script (Bash)".

A screenshot of a Linux desktop environment. On the left, there is a vertical dock with various application icons. In the center, there is a terminal window titled "test@test: ~/Desktop" showing a command prompt. To the right of the terminal is a Visual Studio Code window titled "hello.sh - Desktop - Visual Studio Code" displaying a shell script named "hello.sh". The script contains the following code:

```
1 #!/bin/bash
2 # while loops
3 n=1
4
5 while (( $n <= 10 ))
6 do
7     echo "$n"
8     (( ++n ))
9 done
```

The status bar at the bottom of the Visual Studio Code window shows "Line 9, Col 5" and "Spaces: 4".

A screenshot of a Linux desktop environment. On the left, there is a vertical dock with various application icons. In the center, there is a terminal window titled "terminal" showing the command "test@test: ~/Desktop\$./hello.sh" followed by the output "1 2 3 4 5 6 7 8 9". To the right of the terminal is a Visual Studio Code editor window titled "hello.sh - Desktop - Visual Studio Code". The code editor displays a shell script named "hello.sh" with the following content:

```
1 #!/bin/bash
2 # while loops
3 n=1
4
5 while [ $n -le 10 ]
6 do
7     echo "$n"
8     (( n++ ))
9     sleep 1
10 done
11
12
```

The status bar at the bottom of the VS Code window shows "Ln 9, Col 12" and "Shell Script (Bash)".

The screenshot shows a dual-pane interface of Visual Studio Code. The left pane is a terminal window displaying a bash session. The right pane is a code editor showing a file named 'hello.sh'.

Terminal (Left):

```
test@test:~/Desktop$ ./hello.sh
#!/bin/bash
# while loops

while read p
do
echo $p
done < hello.sh

test@test:~/Desktop$ ./hello.sh
#!/bin/bash
# while loops

cat hello.sh | while read p
do
echo $p
done

test@test:~/Desktop$
```

Code Editor (Right):

```
hello.sh
1 #!/bin/bash
2 # while loops
3
4
5 cat hello.sh | while read p
6 do
7   echo $p
8 done
9
10
```

At the bottom of the code editor, status information is displayed: Ln 5, Col 3 (10 selected), Spaces: 4, UTF-8, LF, Shell Script (Bash).

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a vertical dock containing icons for various applications, including a terminal, file manager, and system settings. The main area features a terminal window and a Visual Studio Code editor.

Terminal Window:

```
test@test:~/Desktop$ cat /etc/host.conf
# The "order" line is only used by old versions of the C
.brary.
order hosts,bind
multi on
test@test:~/Desktop$ ./hello.sh
# The "order" line is only used by old versions of the C
.brary.
order hosts,bind
multi on
test@test:~/Desktop$
```

Visual Studio Code Editor:

```
hello.sh - Desktop - Visual Studio Code
hello.sh x
1 #!/bin/bash
2 # while loops
3
4
5 while IFS= read -r line
6 do
7     echo $line
8 done < /etc/host.conf
9
10
```

The status bar at the bottom of the screen displays the following information: Line 10, Col 1 | Spaces: 4 | UTF-8 | LF | Shell Script (Bash) | ⚡

A screenshot of a Linux desktop environment. On the left, there is a vertical dock with various application icons. In the center, there is a terminal window titled "Visual Studio Code" showing a command-line interface with the prompt "test@test: ~/Desktop\$". To the right of the terminal is a Visual Studio Code editor window titled "hello.sh - Desktop - Visual Studio Code". The code editor displays a shell script named "hello.sh" with the following content:

```
1 #!/bin/bash
2 # until loops
3
4 until [ condition ]
5 do
6     command1
7     command2
8     ...
9     ....
10    commandN
11 done
12
13
14
```

The status bar at the bottom of the Visual Studio Code window shows the file path "hello.sh", line count "Ln 4, Col 8 (1 selected)", character count "Spaces: 4", encoding "UTF-8", line separator "LF", and file type "Shell Script (Bash)".

A screenshot of the Visual Studio Code interface. On the left is a terminal window showing a Linux command-line session:

```
test@test:~/Desktop$ ./hello.sh
1
2
3
4
5
6
7
8
9
test@test:~/Desktop$ ./hello.sh
1
2
3
4
5
6
7
8
9
10
test@test:~/Desktop$
```

On the right is a code editor window titled "hello.sh - Desktop - Visual Studio Code" containing the following shell script:

```
1 #!/bin/bash
2 # until loops
3 n=1
4
5 until $n -gt 10 ]
6 do
7     echo $n
8     n=$(( n+1 ))
9 done
10
11
12
```

The status bar at the bottom indicates the file has 5 lines, 7 columns, 4 spaces, and is in UTF-8 LF format, and is a Shell Script (Bash) file.

A screenshot of the Visual Studio Code interface. On the left is a terminal window showing a bash session:

```
test@test:~/Desktop$ ./hello.sh
1
2
3
4
5
6
7
8
9
10
test@test:~/Desktop$ ./hello.sh
1
2
3
4
5
6
7
8
9
10
test@test:~/Desktop$
```

On the right is a code editor window titled "hello.sh - Desktop - Visual Studio Code" containing the following shell script:

```
#!/bin/bash
# until loops
n=1
until (( $n > 10 ))
do
    echo $n
    n=$(( n+1 ))
done
```

The status bar at the bottom shows: Ln 5, Col 12 (1 selected) Spaces: 4 UTF-8 LF Shell Script (Bash)

A screenshot of a Linux desktop environment. On the left, there is a vertical dock with various application icons. In the center, there is a terminal window titled "terminal" showing the command "test@test: ~/Desktop\$./hello.sh" followed by the output of the script. To the right of the terminal is a Visual Studio Code editor window titled "hello.sh - Desktop - Visual Studio Code". The code editor displays a shell script named "hello.sh" with the following content:

```
1  #!/bin/bash
2  # until loops
3  n=1
4
5  until (( $n > 10 ))
6  do
7      echo $n
8      (( n++ ))
9  done
10
11
12
```

The status bar at the bottom of the screen shows "Ln 8, Col 13" and "Shell Script (Bash)".

Visual Studio Code

test@test:~/Desktop\$

The screenshot shows a dual-pane interface of Visual Studio Code. The left pane is a terminal window with the command 'test@test:~/Desktop\$' entered. The right pane is a code editor displaying a shell script named 'hello.sh'. The script contains several 'for' loops and command substitutions. The status bar at the bottom indicates the file has 28 lines, 0 columns, 0 spaces, and is in UTF-8 LF format, and it is a Shell Script (Bash).

```
hello.sh
1 #!/bin/bash
2 # for loops
3
4 for VARIABLE in 1 2 3 4 5 .. N
5 do
6     command1
7     command2
8     commandN
9 done
#OR-----
11
12 for VARIABLE in file1 file2 file3
13 do
14     command1 on $VARIABLE
15     command2
16     commandN
17 done
#OR-----
19
20 for OUTPUT in $(Linux-Or-Unix-Command-Here)
21 do
22     command1 on $OUTPUT
23     command2 on $OUTPUT
24     commandN
25 done
#OR-----
27 for (( EXP1; EXP2; EXP3 ))
28 do
```

Ln 28, Col 3 Spaces: 4 UTF-8 LF Shell Script (Bash)

Visual Studio Code

test@test:~/Desktop\$

hello.sh - Desktop - Visual Studio Code

```
10 #OR-
11 for VARIABLE in file1 file2 file3
12 do
13     command1 on $VARIABLE
14     command2
15     commandN
16 done
17 #OR-
18
19 for OUTPUT in $(Linux-Or-Unix-Command-Here)
20 do
21     command1 on $OUTPUT
22     command2 on $OUTPUT
23     commandN
24 done
25 #OR-
26 for (( EXP1; EXP2; EXP3 ))
27 do
28     command1
29     command2
30     command3
31 done
32 done
```

Ln 32, Col 5 (64 selected) Spaces: 4 UTF-8 LF Shell Script (Bash)

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a vertical dock containing icons for various applications: a terminal, a file manager, a browser, a file viewer, a calendar, a file editor, a mail client, a system settings icon, and a developer tools icon. The main area has two windows open. The top window is a terminal window titled "Visual Studio Code" with the command "test@test:~/Desktop\$./hello.sh" and its output:

```
test@test:~/Desktop$ ./hello.sh
1
2
3
4
5
test@test:~/Desktop$
```

The bottom window is a code editor titled "hello.sh - Desktop - Visual Studio Code" displaying the following shell script:

```
#!/bin/bash
# for loops
for i in 1 2 3 4 5
do
    echo $i
done
```

The status bar at the bottom of the screen shows "Ln 4, Col 19" and "Spaces: 4".

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a dock containing icons for various applications, including a terminal, file manager, browser, and system tools. In the center, there is a terminal window titled "Terminal" with the command "test@test:~/Desktop\$./hello.sh" and its output (the numbers 1 through 10). To the right of the terminal is a Visual Studio Code editor window titled "hello.sh - Desktop - Visual Studio Code". The code in the editor is:

```
1 #!/bin/bash
2 # for loops
3
4 for i in {1..10}
5 do
6     echo $i
7 done
```

The status bar at the bottom of the screen shows "Ln 5, Col 3" and "Shell Script (Bash)".

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a dock with various icons for applications like a web browser, file manager, terminal, and system tools. The main area has two windows open: a terminal window and a Visual Studio Code editor.

Terminal Window:

```
test@test:~/Desktop$ ./hello.sh
1
2
3
4
5
test@test:~/Desktop$ ./hello.sh
1
2
3
4
5
6
7
8
9
10
test@test:~/Desktop$ ./hello.sh
1
2
3
4
5
6
7
8
9
test@test:~/Desktop$
```

Visual Studio Code Editor:

```
hello.sh x
1 #!/bin/bash
2 # for loops
3
4 for i in {1..10..2}
5 do
6     echo $i
7 done
```

At the bottom of the screen, there is a blue status bar displaying file information: Line 6, Col 11, Spaces: 4, UTF-8, LF, Shell Script (Bash).

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a dock containing icons for various applications: a terminal, a file manager, a browser, a file viewer, a calendar, a document editor, a media player, a file manager, a web browser, a file viewer, a document editor, a media player, and a system settings icon.

The main area features two windows:

- Terminal:** A terminal window titled "test@test: ~/Desktop". It displays the command "test@test:~/Desktop\$./hello.sh" followed by the output "4.3.11(1)-release" and a sequence of numbers from 0 to 4.
- Visual Studio Code:** An editor window titled "hello.sh - Desktop - Visual Studio Code". It shows a shell script named "hello.sh" with the following content:

```
1 #!/bin/bash
2 # for loops
3 echo ${BASH_VERS[ON]}
4 for (( i=0; i<5; i++ ))
5 do
6     echo $i
7 done
```

The status bar at the bottom of the screen indicates the current file is "hello.sh", has 6 lines and 11 columns, 4 spaces per tab, uses UTF-8 encoding, and is a Shell Script (Bash).

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a dock containing icons for various applications, including a terminal, file manager, and system settings. In the center, a terminal window is open with the command `./hello.sh` entered, which results in an error message: `./hello.sh: line 5: [: too many arguments`. To the right of the terminal is a Visual Studio Code editor window titled "hello.sh - Desktop - Visual Studio Code". The code editor displays a shell script named "hello.sh" with the following content:

```
#!/bin/bash
# for loops
for item in *
do
    if [ -d $item ]
    then
        echo $item
    fi
done
```

The status bar at the bottom of the screen indicates the file has 9 lines and 5 columns, is in UTF-8 encoding, and is a Shell Script (Bash).

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a dock containing icons for various applications, including a terminal, file manager, browser, and system tools. In the center, there is a terminal window titled "Terminal" with the command "test@test:~\$./hello.sh" and its output: "doc.md", "examples.desktop", and "hello.sh". To the right of the terminal is a Visual Studio Code editor window titled "hello.sh - Desktop - Visual Studio Code". The code editor displays a shell script named "hello.sh" with the following content:

```
#!/bin/bash
# for loops
for item in *
do
    if [ -f $item ]
    then
        echo $item
    fi
done
```

The status bar at the bottom of the screen indicates "Ls 3, Col 14" and "Shell Script (Bash)".

The image shows a Linux desktop environment with a dark theme. On the left, there is a vertical dock containing icons for various applications, including a terminal, file manager, browser, and productivity tools. The main workspace has two windows open: a terminal window and a Visual Studio Code editor.

Terminal Window:

- Title: Visual Studio Code
- Content: test@test: ~/Desktop
- Bottom status bar: Ln 3, Col 7 (6 selected) Spaces: 4 UTF-8 LF Shell Script (Bash)

Visual Studio Code Editor:

- Title: hello.sh - Desktop - Visual Studio Code
- Content:

```
1 #!/bin/bash
2
3 select varName in list
4 do
5     command1
6     command2
7     ...
8     .....
9     commandN
10 done
```
- Bottom status bar: Ln 3, Col 7 (6 selected) Spaces: 4 UTF-8 LF Shell Script (Bash)

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a vertical dock containing icons for various applications, including a terminal, file manager, browser, and system tools. The main area features two windows: a terminal window and a code editor.

Terminal Window:

```
test@test:~/Desktop$ ./hello.sh
1) mark
2) john
3) tom I
4) ben
#?
```

Code Editor (Visual Studio Code):

```
hello.sh  x
1 #!/bin/bash
2
3 select name in mark john tom ben
4 do
5     echo "$name selected"
6 done
```

The status bar at the bottom of the code editor indicates the file is 6 lines long, 5 columns wide, uses 4 spaces per tab, and is in UTF-8 encoding, with the language set to Shell Script (Bash).

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a dock containing icons for various applications, including a terminal, file manager, and system tools. The main area features two windows: a terminal window and a code editor window.

Terminal Window:

```
test@test:~/Desktop$ ./hello.sh
1) mark
2) john
3) tom
4) ben
#? 2
john selected
#? 
```

Code Editor Window:

```
hello.sh - Desktop - Visual Studio Code
hello.sh  x
1 #!/bin/bash
2
3 select name in mark john tom ben
4 do
5     echo "$name selected"
6 done
```

The code editor window shows a shell script named "hello.sh". The script uses a select loop to prompt the user for a name, with "john" being the selected option. The status bar at the bottom of the code editor indicates the file has 6 lines, 5 columns, and is a Shell Script (Bash).

The screenshot shows a dual-pane interface of Visual Studio Code. The left pane is a terminal window displaying the output of running a shell script named 'hello.sh'. The right pane is a code editor showing the source code of the 'hello.sh' script.

Terminal Output:

```
test@test:~/Desktop$ ./hello.sh
1) mark
2) john
3) tom
4) ben
#? 2
john selected
#? 3
tom selected
#? 1
mark selected
#? ^C
test@test:~/Desktop$ ./hello.sh
1) mark
2) john
3) tom
4) ben
#? 1
mark selected
#? 2
john selected
#? 3
tom selected
#? 4
ben selected
#? 
```

Code Editor Content:

```
hello.sh  x
1 #!/bin/bash
2
3 select name in mark john tom ben
4 do
5   case $name in
6     mark)
7       echo mark selected;;
8     john)
9       echo john selected;;
10    tom)
11      echo tom selected;;
12    ben)
13      echo ben selected;;
14    *)
15      echo "Error please provide the no. between 1..4"
16  esac
17 done
```

The status bar at the bottom indicates the file is 7 lines long, 25 columns wide, with 4 spaces per tab, using UTF-8 encoding, and is a Shell Script (Bash).

The screenshot shows a Linux desktop environment with a terminal window and a code editor window.

Terminal:

```
test@test:~/Desktop$ ./hello.sh
./hello.sh: line 5: [: missing `]'
1
./hello.sh: line 5: [: missing `]'
2
./hello.sh: line 5: [: missing `]'
3
./hello.sh: line 5: [: missing `]'
4
./hello.sh: line 5: [: missing `]'
5
./hello.sh: line 5: [: missing `]'
6
./hello.sh: line 5: [: missing `]'
7
./hello.sh: line 5: [: missing `]'
8
./hello.sh: line 5: [: missing `]'
9
./hello.sh: line 5: [: missing `]'
10
test@test:~/Desktop$ ./hello.sh
1
2
3
4
5
```

Code Editor:

```
hello.sh - Desktop - Visual Studio Code
hello.sh  x
1  #!/bin/bash
2
3  for (( i=1 ; i<=10 ; i++ ))
4  do
5      if [ $i -gt 5 ]
6      then
7          break
8      fi
9      echo "$i"
10 done
```

Visual Studio Code status bar: Line 5 Col 19 Spaces: 2 UFT-8 (F Shell Script (Bash))

A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window and a Visual Studio Code editor.

The terminal window on the left shows the command `./hello.sh` being run, with the output being the numbers 1 through 10, except for 3 and 6 which are skipped by the script's logic.

The Visual Studio Code editor on the right displays the source code for `hello.sh`. The code is a shell script that loops from 1 to 10, skipping the values 3 and 6, and printing the other values.

```
hello.sh - Desktop - Visual Studio Code
hello.sh    x
1 #!/bin/bash
2
3 for (( i=1 ; i<=10 ; i++ ))
4 do
5     if [ $i -eq 3 -o $i -eq 6 ]
6     then
7         continue
8     fi
9     echo "$i"
10 done
```

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a dock with various icons for applications like a terminal, file manager, and system tools. The main area features a terminal window and a Visual Studio Code editor.

Terminal Window:

```
test@test:~/Desktop$ ./hello.sh
Hello
test@test:~/Desktop$ ./hello.sh
Hello
test@test:~/Desktop$ ./hello.sh
Hello
test@test:~/Desktop$ ./hello.sh
Hello
foo
test@test:~/Desktop$
```

Visual Studio Code Editor:

```
hello.sh
1 #!/bin/bash
2
3  function Hello(){
4      echo "Hello"
5  }
6
7  quit () {
8      exit
9  }
10
11 Hello| 1
12
13
14 echo "foo"
15 quit
```

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a dock with various icons for applications like a terminal, file manager, and system tools. On the right, there is a terminal window titled "Terminal" showing the command-line output of a shell script named "hello.sh". The script prints "Hello", "World", and "Again" when run, and "foo" when run with an argument. To the right of the terminal is a Visual Studio Code editor window titled "hello.sh - Desktop - Visual Studio Code" displaying the source code of the "hello.sh" script.

```
#!/bin/bash
function print(){
    echo $1
}
quit () {
    exit
}
print Hello
print World
print Again
echo "foo"
quit
```

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a dock containing icons for various applications, including a terminal, file manager, browser, and system tools. The main area features a terminal window and a Visual Studio Code editor.

Terminal:

```
test@test:~/Desktop$ ./hello.sh
Hello
test@test:~/Desktop$ ./hello.sh
Hello
test@test:~/Desktop$ ./hello.sh
Hello
test@test:~/Desktop$ ./hello.sh
Hello
foo
test@test:~/Desktop$ ./hello.sh
Hello
foo
test@test:~/Desktop$ ./hello.sh
Hello
World
foo
test@test:~/Desktop$ ./hello.sh
Hello
World
Again
foo
test@test:~/Desktop$ ./hello.sh
Hello World Again
foo
test@test:~/Desktop$
```

Visual Studio Code:

```
hello.sh
1 #!/bin/bash
2
3 function print(){
4     echo $1 $2 $3
5 }
6
7 quit () {
8     exit
9 }
10
11 print Hello World Again
12
13 echo "foo"
14 quit
```

The screenshot shows a dual-pane interface of Visual Studio Code. The left pane is a terminal window with a dark background, displaying a Linux shell session. The right pane is a code editor window with a light background, displaying a Bash script named `hello.sh`.

Terminal Output:

```
test@test:~/Desktop$ ./hello.sh
the name is Max
foo
test@test:~/Desktop$ ./hello.sh
The name is Tom
the name is Max
foo
test@test:~/Desktop$ ./hello.sh
The name is Tom : Before
the name is Max
The name is Max : After
test@test:~/Desktop$
```

Code Editor Content (`hello.sh`):

```
1  #!/bin/bash
2
3  function print(){
4      name=$1
5      echo "the name is $name"
6  }
7
8  name="Tom"
9
10 echo "The name is $name : Before"
11 print Max
12 echo "The name is $name : After"
```

The code editor shows syntax highlighting for the Bash script. A cursor is positioned at the start of the `function print()` line. The status bar at the bottom indicates the file is 3 lines long, has 1 character selected, and is in UTF-8 encoding.

Terminal

```
test@test:~/Desktop$ ./hello.sh
the name is Max
foo
test@test:~/Desktop$ ./hello.sh
The name is Tom
'the name is Max
foo
test@test:~/Desktop$ ./hello.sh
The name is Tom : Before
the name is Max
The name is Max : After
test@test:~/Desktop$ ./hello.sh
The name is Tom : Before
the name is Max
The name is Tom : After
test@test:~/Desktop$
```

hello.sh - Desktop - Visual Studio Code

```
hello.sh    x
1  #!/bin/bash
2
3  function print(){
4      local name=$1
5      echo "the name is $name"
6  }
7
8  name="Tom"
9
10 echo "The name is $name : Before"
11
12 print Max
13
14 echo "The name is $name : After"
```

Ln 9 Col 1 Spaces: 2 UTF-8 LF Shell Script (Bash)

The screenshot shows a dual-pane interface of Visual Studio Code. The left pane is a terminal window displaying a Linux shell session. The right pane is a code editor showing a shell script named `hello.sh`.

Terminal Output:

```
test@test:~/Desktop$ ls
hello.sh
test@test:~/Desktop$ touch file1.txt
test@test:~/Desktop$ ls
file1.txt  hello.sh
test@test:~/Desktop$ ./hello.sh
You need to provide an argument :
usage : ./hello.sh file_name
./hello.sh: line 9: local: command not found
File not found
test@test:~/Desktop$ ./hello.sh file1.txt
File found
test@test:~/Desktop$
```

Code Editor Content (`hello.sh`):

```
#!/bin/bash
usage() {
    echo "You need to provide an argument : "
    echo "usage : $0 file_name"
}
is_file_exist() {
    local file="$1"
    [[ -f "$file" ]] && return 0 || return 1
}
[[ $# -eq 0 ]] && usage
if ( is_file_exist "$1" )
then
    echo "File found"
else
    echo "File not found"
fi
```

Bottom status bar: Ln 10, Col 35 (2 selected) Spaces: 2 UTF-8 LF Shell Script (Bash)

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a vertical dock containing icons for various applications: a terminal, a file manager, a browser (Firefox), a file viewer, a calendar, a document editor, a file manager, a developer tools icon, a settings gear, and a system tray icon.

The main area features two windows:

- Terminal Window:** Titled "test@test: ~/Desktop". It shows the command `./hello.sh` being run twice. The first run succeeds, but the second run fails with the error message "var: readonly variable".

```
test@test:~/Desktop$ ./hello.sh
./hello.sh: line 7: var: readonly variable
test@test:~/Desktop$ ./hello.sh
./hello.sh: line 7: var: readonly variable
var => 31
test@test:~/Desktop$
```
- Visual Studio Code Editor:** Titled "hello.sh - Desktop - Visual Studio Code". It displays the content of the `hello.sh` script.

```
hello.sh  x
1  #!/bin/bash
2
3  var=31
4
5  readonly var
6
7  var=50 ]
8
9  echo "var => $var"
```

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a vertical dock containing icons for various applications: a terminal, a file manager, a browser (Firefox), a file viewer, a calendar, a file manager, a file viewer, and a file manager.

The main window area contains two panes. The left pane is a terminal window titled "Visual Studio Code" with the command line "test@test:~/Desktop\$./hello.sh" and its output "Hello". The right pane is a code editor titled "hello.sh - Desktop - Visual Studio Code" displaying the following Bash script:

```
#!/bin/bash
hello() {
    echo "Hello"
}
readonly -f hello
```

The screenshot shows a Linux desktop environment with a terminal window and a code editor.

Terminal:

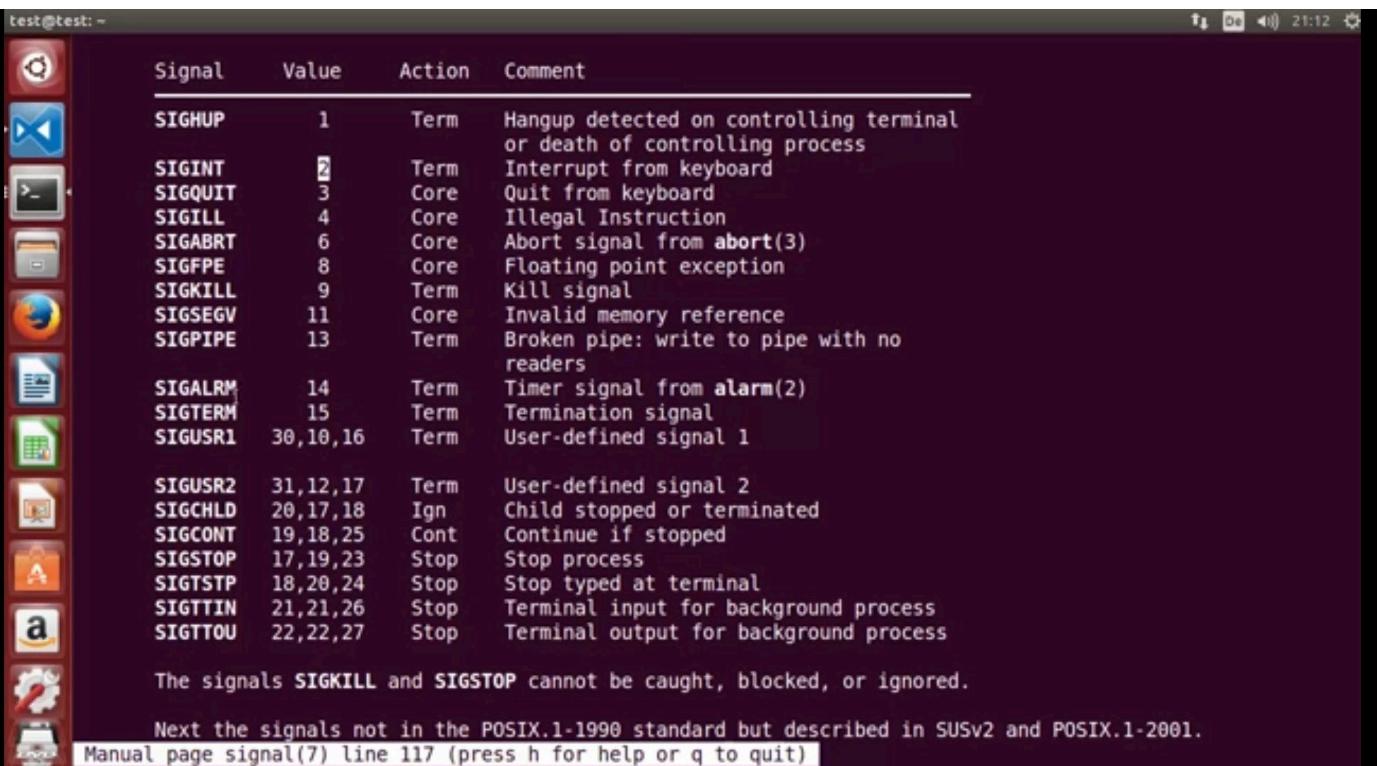
```
test@test: ~/Desktop
1
2
^Z
[1]+  Stopped                  ./hello.sh
test@test:~/Desktop$ ./hello.sh
pid is 3784
1
2
3
4
5
6
Killed
test@test: ~
test@test:~$ kill -9 3784
test@test:~$
```

Visual Studio Code:

hello.sh

```
1 #!/bin/bash
2 echo "pid is $$"
3 while (( COUNT < 10 ))
4 do
5     sleep 10
6     (( COUNT ++ ))
7     echo $COUNT
8 done
9 exit 0
```

test@test: ~



The screenshot shows a terminal window on an Ubuntu desktop environment. The terminal displays a table of signals with their values and actions, followed by a note about SIGKILL and SIGSTOP, and a message about non-POSIX signals.

Signal	Value	Action	Comment
SIGHUP	1	Term	Hangup detected on controlling terminal or death of controlling process
SIGINT	2	Term	Interrupt from keyboard
SIGQUIT	3	Core	Quit from keyboard
SIGILL	4	Core	Illegal Instruction
SIGABRT	6	Core	Abort signal from <code>abort(3)</code>
SIGFPE	8	Core	Floating point exception
SIGKILL	9	Term	Kill signal
SIGSEGV	11	Core	Invalid memory reference
SIGPIPE	13	Term	Broken pipe: write to pipe with no readers
SIGNALRM	14	Term	Timer signal from <code>alarm(2)</code>
SIGTERM	15	Term	Termination signal
SIGUSR1	30,10,16	Term	User-defined signal 1
SIGUSR2	31,12,17	Term	User-defined signal 2
SIGCHLD	20,17,18	Ign	Child stopped or terminated
SIGCONT	19,18,25	Cont	Continue if stopped
SIGSTOP	17,19,23	Stop	Stop process
SIGTSTP	18,20,24	Stop	Stop typed at terminal
SIGTTIN	21,21,26	Stop	Terminal input for background process
SIGTTOU	22,22,27	Stop	Terminal output for background process

The signals **SIGKILL** and **SIGSTOP** cannot be caught, blocked, or ignored.

Next the signals not in the POSIX.1-1990 standard but described in SUSv2 and POSIX.1-2001.
Manual page `signal(7)` line 117 (press h for help or q to quit)

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a vertical dock containing icons for various applications: a terminal, a file manager, a browser, a code editor, a file viewer, a terminal, a file manager, a browser, a file viewer, a terminal, a file manager, a browser, a file viewer, and a terminal.

The main area features two windows of the Visual Studio Code application:

- Terminal Window:** The title bar says "Visual Studio Code". It displays a terminal session:

```
test@test:~/Desktop$ ./hello.sh
pid is 4296
1
Killed
test@test:~/Desktop$
```
- Code Editor Window:** The title bar says "hello.sh - Desktop - Visual Studio Code". It shows the source code for a shell script:

```
#!/bin/bash
trap "echo Exit signal is detected" SIGKILL SIGSTO
echo "pid is $$"
while (( COUNT < 10 ))
do
    sleep 10
    (( COUNT ++ ))
    echo $COUNT
done
exit 0
```

At the bottom of the screen, the system tray shows the date and time as "In 11, Col 7" and "21:21".

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a dock with various icons for applications like a web browser, file manager, terminal, and system tools. In the center, a terminal window is open with the command `stty raw -echo` running, which is capturing keyboard input and echoing it back. The output shows multiple instances of the message '^CExit signal is detected'. To the right of the terminal is a Visual Studio Code editor window displaying a shell script named `hello.sh`. The script contains the following code:

```
#!/bin/bash
trap "echo Exit signal is detected" SIGINT
echo "pid is $$"
while (( COUNT < 10 ))
do
    sleep 10
    (( COUNT ++ ))
    echo $COUNT
done
exit 0
```

The screenshot shows a Linux desktop environment with a terminal window and a code editor window.

Terminal:

```
test@test:~/Desktop$ ./hello.sh
pid is 5633
1
2
test@test:~/Desktop$ ls
hello.sh
test@test:~/Desktop$ touch file.txt
test@test:~/Desktop$ ls
file.txt hello.sh
test@test:~/Desktop$ ./hello.sh
pid is 5667
file deleted
file deleted
test@test:~/Desktop$ kill -15 5633
test@test:~/Desktop$ kill -15 5667
test@test:~/Desktop$
```

Code Editor:

```
hello.sh
1 #!/bin/bash
2  file=/home/test/Desktop/file.txt
3  trap "rm -f $file && echo file deleted; exit" 0 2
4
5  echo "pid is $$"
6  while (( COUNT < 10 ))
7  do
8      sleep 10
9      (( COUNT ++ ))
10     echo $COUNT
11 done
12 exit 0
```

The screenshot shows a Linux desktop environment with a terminal window and a Visual Studio Code editor.

Terminal:

```
test@test:~/Desktop
4
5
^Cfile deleted
file deleted
test@test:~/Desktop$ ^C
^Ctest@test:~/Desktop$ ^C
test@test:~/Desktop$ ^C
test@test:~/Desktop$ ^C
test@test:~/Desktop$ ./hello.sh
pid is 5697
^Cfile deleted
file deleted
test@test:~/Desktop$ ^C
test@test:~
trap -- '' SIGTTIN
trap -- '' SIGTTOU
test@test:~$ trap
trap -- 'rm /home/test/Desktop/file.txt ; exit' EXIT
trap -- 'rm /home/test/Desktop/file.txt ; exit' SIGINT
trap -- 'rm /home/test/Desktop/file.txt ; exit' SIGTERM
trap -- '' SIGTSTP
trap -- '' SIGTTIN
trap -- '' SIGTTOU
test@test:~$ trap - 0 2 15
test@test:~$ trap
trap -- '' SIGTSTP
trap -- '' SIGTTIN
trap -- '' SIGTTOU
test@test:~$
```

Visual Studio Code:

```
hello.sh - Desktop - Visual Studio Code
hello.sh
1 #!/bin/bash
2 rm -f $file && echo file deleted; exit 0 2 15
3
4 pid is $$"
5 (( COUNT < 10 ))
6
7 sleep 10
8 COUNT ++
9
10 echo $COUNT
11
12
```

The screenshot shows a Linux desktop environment with a dark theme. On the left, there is a vertical dock containing icons for various applications, including a terminal, file manager, browser, and system tools. The main area has two windows open: a terminal window and a Visual Studio Code editor.

Terminal:

```
test@test:~/Desktop$ ./hello.sh
+ file=/home/test/Desktop/file.txt
+ trap 'rm -f /home/test/Desktop/file.txt && echo file deleted; exit' 0 2 15
deleted; exit' 0 2 15
+ echo 'pid is 3669'
pid is 3669
+ (( COUNT < 10 ))
+ sleep 10
```

Visual Studio Code:

```
hello.sh
1 #!/bin/bash -x
2
3
4 file=/home/test/Desktop/file.txt
5 trap "rm -f $file && echo file deleted; exit" 0 2
6
7 echo "pid is $$"
8 while (( COUNT < 10 ))
9 do
10     sleep 10
11     (( COUNT ++ ))
12     echo $COUNT
13 done
14 exit 0
```

The terminal window shows the execution of a shell script named `hello.sh`. The script creates a file at `/home/test/Desktop/file.txt`, traps signals 0, 2, and 15 to delete the file and exit, prints the process ID, and then enters a loop that sleeps for 10 seconds, increments a counter, and prints the count until it reaches 10. The Visual Studio Code window shows the source code for this script, which matches the terminal output.

The screenshot shows a Linux desktop environment with a dark theme. On the left is a dock containing icons for various applications, including a terminal, file manager, browser, and system tools. The main area has two windows: a terminal window and a code editor.

Terminal:

```
test@test:~/Desktop$ ./hello.sh
+ file=/home/test/Desktop/file.txt
+ set +x
pid is 3706
1
```

Code Editor:

```
hello.sh  x
1 #!/bin/bash
2  set -x
3
4  file=/home/test/Desktop/file.txt
5
6  set +x
7  trap "rm -f $file && echo file deleted; exit" 0 2
8
9  echo "pid is $$"
10 while (( COUNT < 10 ))
11 do
12     sleep 10
13     (( COUNT ++ ))
14     echo $COUNT
15 done
16 exit 0
```

At the bottom of the screen, there is a status bar with the following information: In 6, Col 7, Spaces: 4, UTE-B, LF, Shell Script (Bash).