Basic Of Shell Scripting
II

read Command

- Syntax for taking multiple user input using read keyword.

```
#!/bin/bash
echo "Enter three course name"
read name1 name2 name3
echo "names are: $name1, $name2, $name3"
```

Example

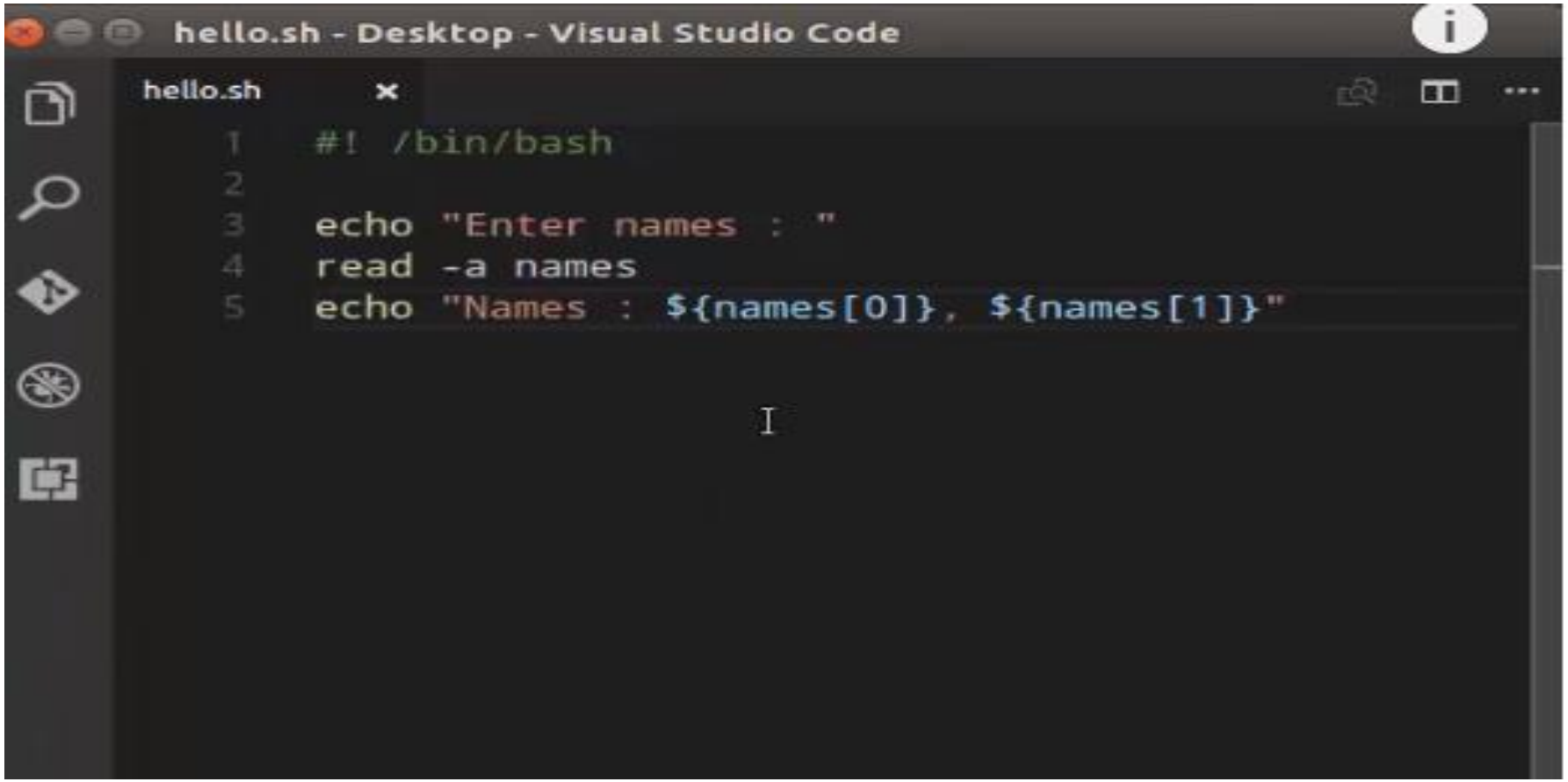
hello.sh

```
#! /bin/bash

read -p 'username : ' user_var
read -sp 'password : ' pass_var
echo
echo "username : $user_var"
echo "password : $pass_var"
```

```
test@test:~/Desktop$ ./hello.sh
username : myuser
password :
username : myuser
password : 12345
```

Example


hello.sh - 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]}"

Adding Basic Options

- Let’s take a look at some of the most basic options we can use:
- *-a array*: stores the results of the word split operation in an array rather than separate variables
- *-s:* does not echo the input line to the standard output stream
- *–p prompt*: print the prompt text before requesting the input from the standard input stream without a *<newline>* character
- *-t timeout:* attempt to read the input for a given period of seconds
- *–N:* read exactly N characters from the input unless a timeout occurs or *EOF* is reached

Conditional Statement

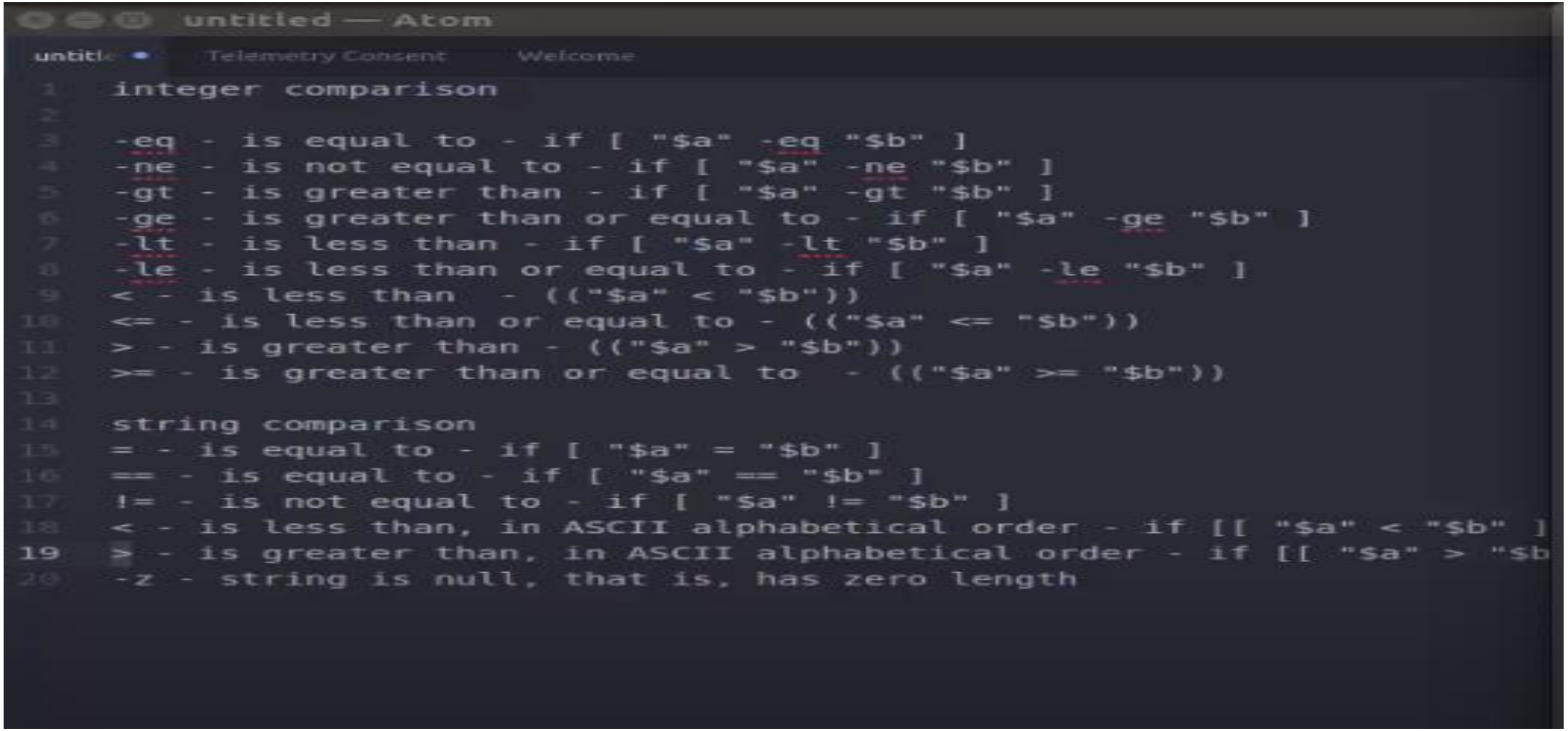
If-else Statement #

- Bash if conditionals can have different forms. The most basic if statement takes the following form:

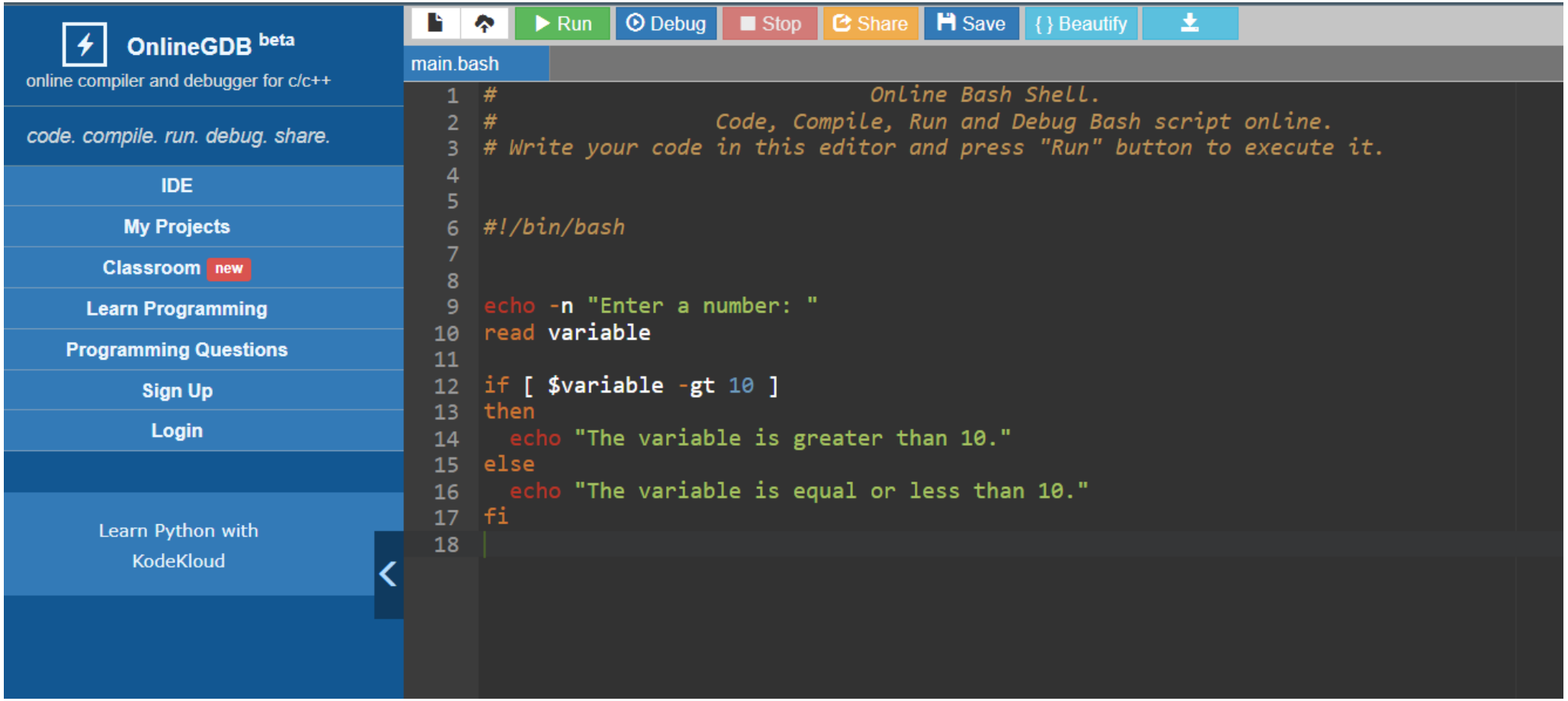
```
if TEST-COMMAND
then
  STATEMENTS
Else
STATEMENTS
fi
```

- The if statement starts with the if keyword followed by the conditional expression and the then keyword. The statement ends with the fi keyword.

Syntax


untitled — Atom
untitl
Telemetry Consent
Welcome
integer comparison
-eq - is equal to - if ["$a" -eq "$b"]
-ne - is not equal to - if ["$a" -ne "$b"]
-gt - is greater than - if ["$a" -gt "$b"]
-ge - is greater than or equal to - if ["$a" -ge "$b"]
-lt - is less than - if ["$a" -lt "$b"]
-le - is less than or equal to - if ["$a" -le "$b"]
< - is less than - (("$a" < "$b"))
<= - is less than or equal to - (("$a" <= "$b"))
> - is greater than - (("$a" > "$b"))
>= - is greater than or equal to - (("$a" >= "$b"))
string comparison
= - is equal to - if ["$a" = "$b"]
== - is equal to - if ["$a" == "$b"]
!= - is not equal to - if ["$a" != "$b"]
< - is less than, in ASCII alphabetical order - if [["$a" < "$b"]
> - is greater than, in ASCII alphabetical order - if [["$a" > "$b
-z - string is null, that is, has zero length

Example Code



if..elif..else Statement

```
if TEST-COMMAND1
then
  STATEMENTS1
elif TEST-COMMAND2
then
  STATEMENTS2
else
  STATEMENTS3
fi
```

If the TEST-COMMAND1 evaluates to True, the STATEMENTS1 will be executed. If the TEST-COMMAND2 evaluates to True, the STATEMENTS2 will be executed. If none of the test commands evaluate to True, the STATEMENTS2 is executed.

Example Code