* [1read Command Syntax](https://bash.cyberciti.biz/guide/Getting_User_Input_Via_Keyboard#read_Command_Syntax)
  + [1.1Handling Input](https://bash.cyberciti.biz/guide/Getting_User_Input_Via_Keyboard#Handling_Input)
  + [1.2Examples](https://bash.cyberciti.biz/guide/Getting_User_Input_Via_Keyboard#Examples)
    - [1.2.1Multiple Input (number.sh)](https://bash.cyberciti.biz/guide/Getting_User_Input_Via_Keyboard#Multiple_Input_.28number.sh.29)
    - [1.2.2Display Domain Owner Information](https://bash.cyberciti.biz/guide/Getting_User_Input_Via_Keyboard#Display_Domain_Owner_Information)
    - [1.2.3Timeout Input](https://bash.cyberciti.biz/guide/Getting_User_Input_Via_Keyboard#Timeout_Input)
    - [1.2.4Handling Passwords](https://bash.cyberciti.biz/guide/Getting_User_Input_Via_Keyboard#Handling_Passwords)
* [2Handling multiple values](https://bash.cyberciti.biz/guide/Getting_User_Input_Via_Keyboard#Handling_multiple_values)
  + [2.1$IFS](https://bash.cyberciti.biz/guide/Getting_User_Input_Via_Keyboard#.24IFS)
    - [2.1.1But how do I use $IFS and read command together?](https://bash.cyberciti.biz/guide/Getting_User_Input_Via_Keyboard#But_how_do_I_use_.24IFS_and_read_command_together.3F)
    - [2.1.2How do I change the IFS separator value?](https://bash.cyberciti.biz/guide/Getting_User_Input_Via_Keyboard#How_do_I_change_the_IFS_separator_value.3F)
* [3See also](https://bash.cyberciti.biz/guide/Getting_User_Input_Via_Keyboard#See_also)

read Command Syntax

read -p "Prompt" variable1 variable2 variableN

Where,

* **-p "Prompt"** : Display prompt to user without a newline.
* **variable1** : The first input (word) is assigned to the variable1.
* **variable2** : The second input (word) is assigned to the variable2.

**Handling Input**

Create a script called greet.sh as follows:

*#!/bin/bash*

read -p "Enter your name : " name

echo "Hi, $name. Let us be friends!"

Save and close the file. Run it as follows:

chmod +x greet.sh

./greet.sh

Sample Outputs:

Enter your name : Vivek Gite

Hi, Vivek Gite. Let us be friends!

**Examples**

Try the following examples.

**Multiple Input (number.sh)**

*#!/bin/bash*

*# read three numbers and assigned them to 3 vars*

read -p "Enter number one : " n1

read -p "Enter number two : " n2

read -p "Enter number three : " n3

*# display back 3 numbers - punched by user.*

echo "Number1 - $n1"

echo "Number2 - $n2"

echo "Number3 - $n3"

**Display Domain Owner Information**

A shell script to display the Internet domain name owner information (domain.sh):

*#!/bin/bash*

read -p "Enter the Internet domain name (e.g. nixcraft.com) : " domain\_name

whois $domain\_name

**Timeout Input**

You can time out read command using the -t option. It causes read to time out and return failure if a complete line of input is not read within TIMEOUT seconds. For example, if no input provided within 10 second, program will be aborted (domain2.sh):

*#!/bin/bash*

read -t 10 -p "Enter the Internet domain name (e.g. nixcraft.com) : " domain\_name

whois $domain\_name

**Handling Passwords**

The -s option causes input coming from a terminal do not be displayed on the screen. This is useful for password handling (readpass.sh):

*#!/bin/bash*

read -s -p "Enter Password : " my\_password

echo

echo "Your password - $my\_password"

Handling multiple values

Consider the following example:

read -p "Enter directory to delete : " dirname

echo "$dirname"

Sample outputs:

Enter directory to delete : foo bar /tmp/data

foo bar /tmp/data

The user supplied three values instead of one. The string is now made of three different fields. All three words are assigned to dirname using [$IFS](https://bash.cyberciti.biz/guide/$IFS) internal field separator. The [$IFS](https://bash.cyberciti.biz/guide/$IFS) determines how shell recognizes fields.

**$IFS**

To display default value of [$IFS](https://bash.cyberciti.biz/guide/$IFS), enter:

echo "$IFS"

You will see a whitespace which is nothing but a space, a tab, and a newline (default). You can print actual values of IFS using the following command (see [Here strings](https://bash.cyberciti.biz/guide/Here_strings)):

cat -etv <<<"$IFS"

Sample outputs:

^I$

$

Where,

* **$** - end of line i.e. newline
* **^I$** - tab and newline

**But how do I use $IFS and read command together?**

Create a variable called nameservers and give it total 3 values as follows (note all values are separated by a whitespace):

nameservers="ns1.nixcraft.net ns2.nixcraft.net ns3.nixcraft.net"

Display the value of a variable nameservers with [echo command](https://bash.cyberciti.biz/guide/Echo_command) or [printf command](https://bash.cyberciti.biz/guide/Printf_command):

echo "$nameservers"

OR

printf "%s" $nameservers

Now, you can simply split $nameservers using the [read command](https://bash.cyberciti.biz/guide/Read_command) as follows (see [Here strings](https://bash.cyberciti.biz/guide/Here_strings)):

read -r ns1 ns2 ns3 <<< "$nameservers"

Where,

* The [read command](https://bash.cyberciti.biz/guide/Read_command) reads input from $nameservers variable.
* The default value of [$IFS](https://bash.cyberciti.biz/guide/$IFS) is used to assign values to three separate variables. Your input is broken into tokens using [$IFS](https://bash.cyberciti.biz/guide/$IFS) and assigned to three variables.
* In other words, the [IFS variable](https://bash.cyberciti.biz/wiki/index.php?title=IFS_variable&action=edit&redlink=1) worked as token delimiter or separator.
* The first token (ns1.nixcraft.net) is saved as the value of the first variable ($ns1)
* The second token (ns2.nixcraft.net) is saved as the value of the second variable ($ns2).
* The third token (ns3.nixcraft.net) is saved as the value of the third variable ($ns3).
* To display the value of each variable use [echo command](https://bash.cyberciti.biz/guide/Echo_command) or [printf command](https://bash.cyberciti.biz/guide/Printf_command) as follows:

echo "DNS Server #1 $ns1"

echo " #2 $ns2"

echo " #3 $ns2"

OR use the [printf command](https://bash.cyberciti.biz/guide/Printf_command)

printf "DNS Server #1 %s\n #2 %s\n #3 %s\n" $ns1 $ns2 $ns3

Sample outputs:

DNS Server #1 ns1.nixcraft.net

#2 ns2.nixcraft.net

#3 ns3.nixcraft.net

**How do I change the IFS separator value?**

Consider the following [/etc/passwd](https://bash.cyberciti.biz/guide/etc/passwd) line:

gitevivek:x:1002:1002::/home/gitevivek:/bin/sh

Assign the above line to a variable called pwd:

pwd="gitevivek:x:1002:1002::/home/gitevivek:/bin/sh"

Save the Internal Field Separator to a variable called old:

old="$IFS"

Set the Internal Field Separator to a colon (i.e. change the Internal Field Separator):

IFS=:

Read $pwd and generate tokens using $IFS and store them into respective fields:

read -r login password uid gid info home shell <<< "$pwd"

printf "Your login name is %s, uid %d, gid %d, home dir set to %s with %s as login shell\n" $login $uid $gid $home $shell

Sample outputs:

Your login name is gitevivek, uid 1002, gid 1002, home dir set to /home/gitevivek with /bin/sh as login shell

Finally, restore the Internal Field Separator value using $old:

IFS="$old"

Where,

* **:** - act as token separator on $pwd i.e. the contents of the IFS variable are used as token delimiters.
* **login** - Field # 1 is generated using the first token and is saved as the value of the first variable ($login)
* **password** - Field # 2 is generated using the second token and is saved as the value of the second variable ($password)
* **uid** - Field # 3 and so on...
* **gid** - Field # 4
* **info** - Field # 5
* **home** - Field # 6
* **shell** - Field # 7