Operating Systems Laboratory (UGCA1926)

GIAN JYOTI INSTITUTE OF MANAGEMENT & TECHNOLOGY

PHASE 2, MOHALI

AFFILIATED TO

PUNJAB TECHNICAL UNIVERSITY, JALANDHAR

FOR THE PARTIAL FULFILLMENT FOR QUALIFYING BCA DEGREE

<u>SUBMITTED TO</u>:

Prof. Tarandeep Singh

<u>SUBMITTED BY:</u>

Name –Suraj

Class - BCA IV (Sec-B)

Roll No. - 2111840

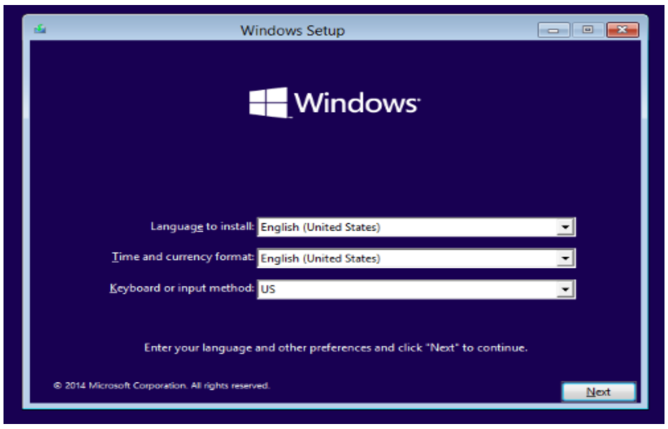
INDEX

1. INSTALLATION OF WINDOWS OS

• To start the Windows install or upgrade process, you need to configure your computer to boot from a CD or DVD before booting to the hard drive. Changing the boot process forces the computer to look for the Windows installation disc before booting from the hard drive.

• Open the CMOS setup.

How to enter and exit the BIOS or CMOS setup.
Change the computer's boot order. Set the CD, DVD, or disc drive as the first boot device if you are trying to boot from a disc. Or, set the first boot device to your USB drive if you're trying to boot from a USB thumb drive. If the drive is not shown, keep the disc is inserted and reboot the computer. With the disc in the drive, BIOS should recognize and include it in the list.
Save the settings change and exit BIOS.
Once you have updated the boot order, you can begin the Windows installation process.



Language to install: English (United States)
Time and currency format: English (United States)
Keyboard or input method: US

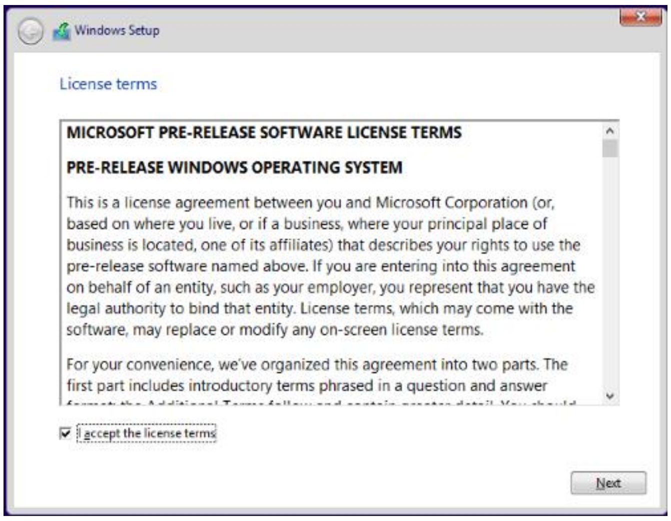
Enter your language and other preferences and click "Next" to continue.

© 2014 Microsoft Corporation. All rights reserved.

Steps to install Windows 10

Step 1: Download Windows 10 ISO image from http://windows.microsoft.com/en-us/windows/preview-iso.
Step 2: Mount the ISO image as a drive or burn it into a DVD. Optionally, you can create a Windows 10 bootable USB drive.
Step 5: In the welcome screen, choose your language settings and click the "Next" button to proceed.

Step 6: Accept the license terms and click "Next" to proceed. Step 7: In the next step, you have to select the drive to install. If you have multiple drives on your computer, you will see them all there. Choose the correct drive to install. If you are installing Windows 10 over an existing Windows, you may install side by side on another drive.



MICROSOFT PRE-RELEASE SOFTWARE LICENSE TERMS

PRE-RELEASE WINDOWS OPERATING SYSTEM

This is a license agreement between you and Microsoft Corporation (or, based on where you live, or if a business, where your principal place of business is located, one of its affiliates) that describes your rights to use the pre-release software named above. If you are entering into this agreement on behalf of an entity, such as your employer, you represent that you have the legal authority to bind that entity. License terms, which may come with the software, may replace or modify any on-screen license terms.

For your convenience, we've organized this agreement into two parts. The first part includes introductory terms phrased in a question and answer

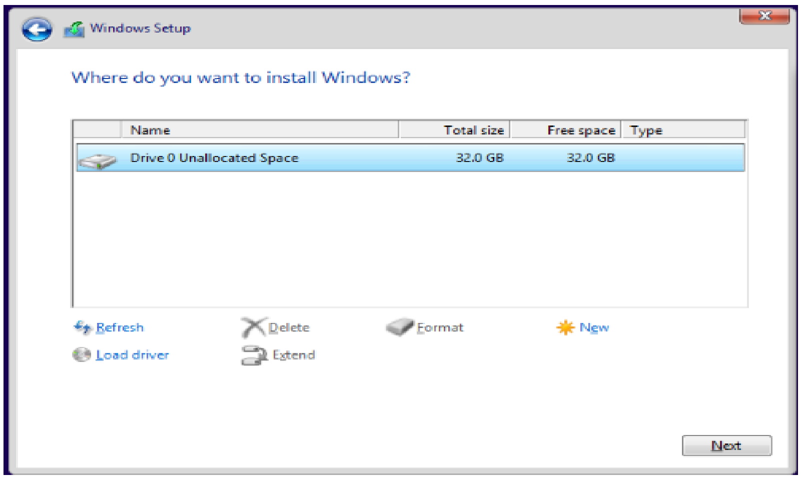
☑ I accept the license terms

Step 8: Installer will copy all the necessary files to the computer and continue with the installation. Depending on your system configuration, it may take a while (10-20 minutes) to complete the installation.

Step 9: After successful installation, you will see the Windows welcome screen.

Step 10: The computer may need to restart several times during the Windows install process. The restarts are normal and if prompted to restart, select the Yes option.

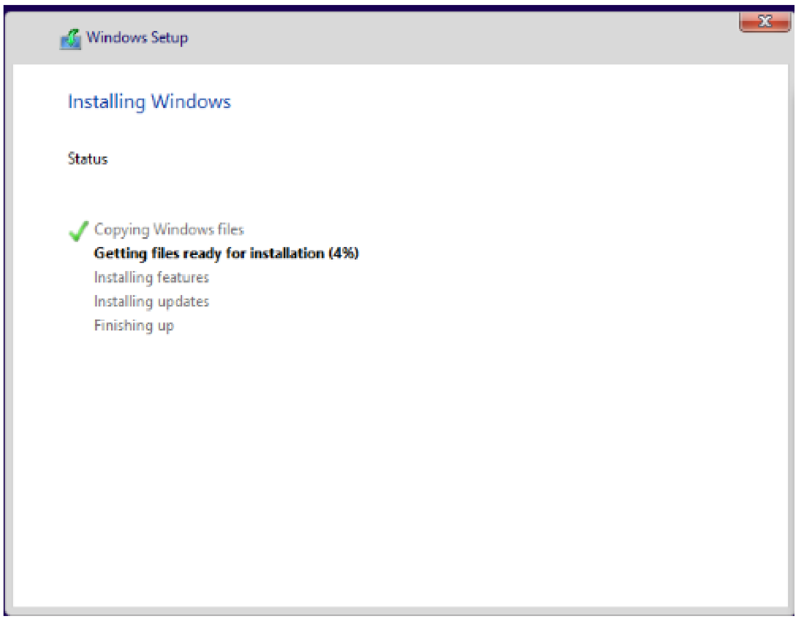
Step 11:When the install process is nearly complete, the Windows configuration option screens are shown. On these screens, you may be asked to select the time zone you live in, your preferred language, and the account's name you use to access Windows. Select the appropriate options and enter the appropriate information on each configuration screen. The Windows install process is completed when the computer prompts you to log in or when it loads into Windows.



Windows Setup

Where do you want to install Windows?

Name	Total size	Free space	Type
Drive 0 Unallocated Space	32.0 GB	32.0 GB	

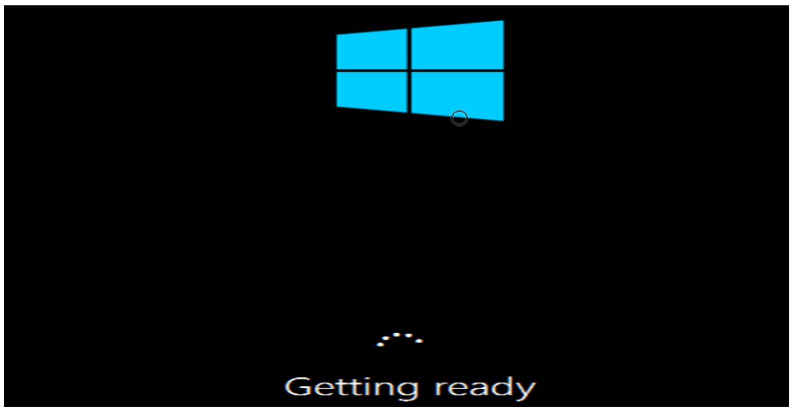


Windows Setup

Installing Windows

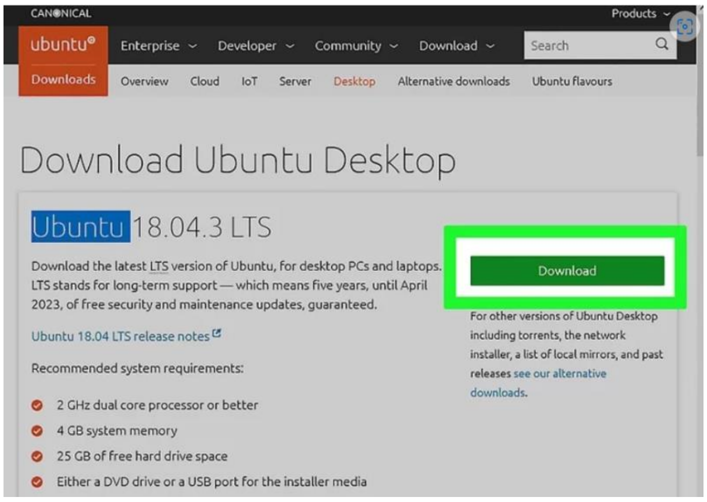
Status

✓ Copying Windows files
Getting files ready for installation (4%)
Installing features
Installing updates
Finishing up



2. INSTALLATION OF LINUX OS

Steps to install Linux OS.
Step 1:Download a disk image (ISO) for the operating system you want to install.



Download Ubuntu Desktop

Ubuntu 18.04.3 LTS

Download the latest LTS version of Ubuntu, for desktop PCs and laptops. LTS stands for long-term support — which means five years, until April 2023, of free security and maintenance updates, guaranteed.

Ubuntu 18.04 LTS release notes

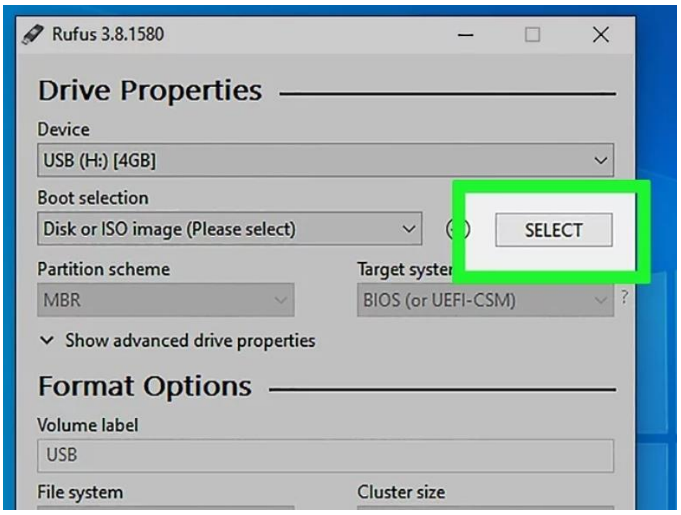
Recommended system requirements:

- 2 GHz dual core processor or better
- 4 GB system memory
- 25 GB of free hard drive space
- Either a DVD drive or a USB port for the installer media

For other versions of Ubuntu Desktop including torrents, the network installer, a list of local mirrors, and past releases see our alternative downloads.

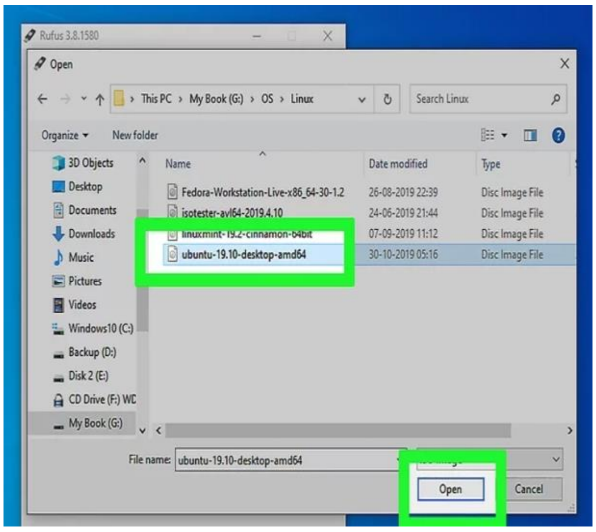
Step 2:Insert a blank USB flash drive.
Step 3:Open Rufus. It has an icon that resembles a USB flash drive. Click the Rufus icon in your Windows Start menu to open Rufus.

Step 4:Select your USB flash drive. Use the drop-down menu below "Devices" to select your USB flash drive. And than Click Select.



Device: USB (H:) [4GB]
Boot selection: Disk or ISO image (Please select)
Partition scheme: MBR
Target system: BIOS (or UEFI-CSM)
Volume label: USB
File system
Cluster size

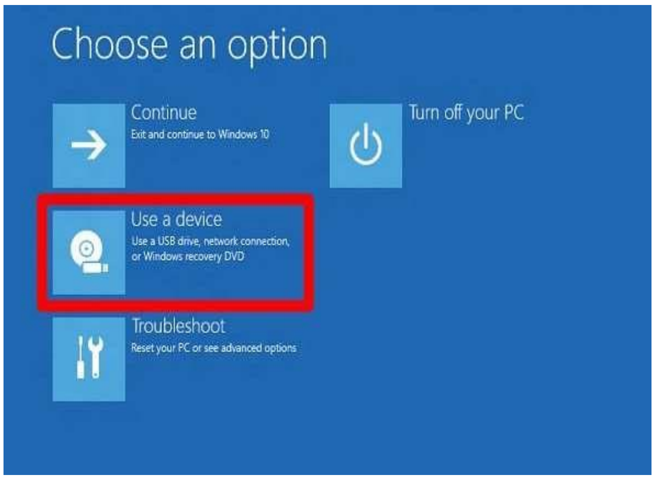
Step 5:Select the operating system ISO file and click Open. This loads the ISO file into Rufus. And than Click Select.



Name	Date modified	Type
Fedora-Workstation-Live-x86_64-30-1.2	26-08-2019 22:39	Disc Image File
isotester-avl64-2019.4.10	24-06-2019 21:44	Disc Image File
linuxmint-19.2-cinnamon-64bit	07-09-2019 11:12	Disc Image File
ubuntu-19.10-desktop-amd64	30-10-2019 05:16	Disc Image File

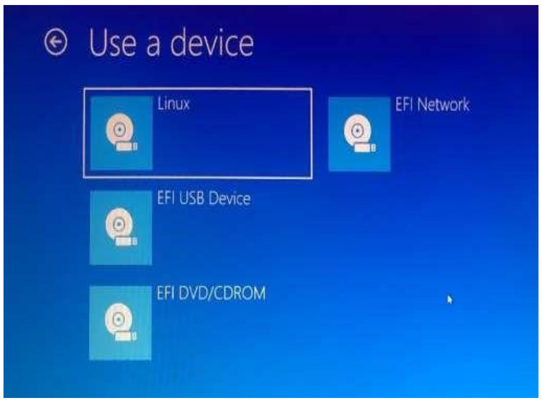
File name: ubuntu-19.10-desktop-amd64

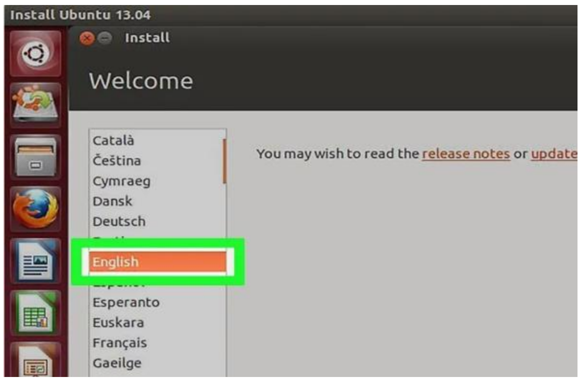
Step 6: Click Start
Step 7: Insert the install disk/Bootable usb. Restart your computer.
. Then select Use a Device.


Choose an option
Continue
Exit and continue to Windows 10
Turn off your PC
Use a device
Use a USB drive, network connection, or Windows recovery DVD
Troubleshoot
Reset your PC or see advanced options

. Find your device in the list. If you don't see your drive, choose EFI USB Device, then pick your drive from the next screen.

Step 8: Wait for the setup program to load.
Step 9: Select your language and keyboard layout.

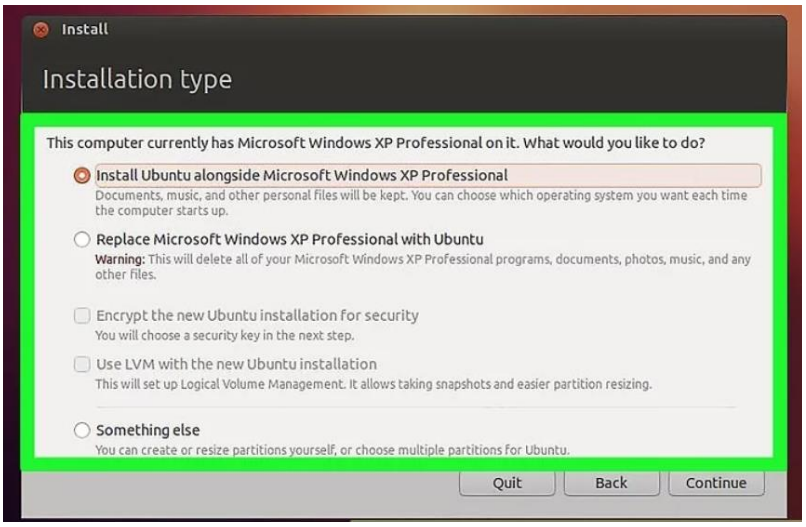




You may wish to read the release notes or update

Step 10:
Select the "Custom" or "Other" installation option.

Step 11: Format the drive you want to install the operating system on.



This computer currently has Microsoft Windows XP Professional on it. What would you like to do?

(●) Install Ubuntu alongside Microsoft Windows XP Professional
Documents, music, and other personal files will be kept. You can choose which operating system you want each time the computer starts up.

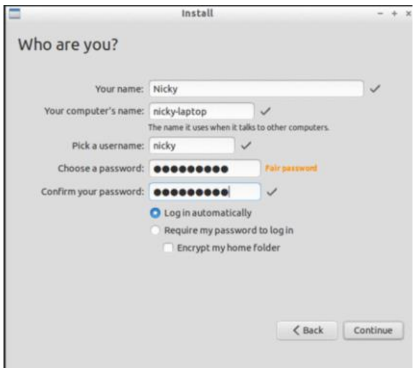
( ) Replace Microsoft Windows XP Professional with Ubuntu
Warning: This will delete all of your Microsoft Windows XP Professional programs, documents, photos, music, and any other files.

☐ Encrypt the new Ubuntu installation for security
You will choose a security key in the next step.

☐ Use LVM with the new Ubuntu installation
This will set up Logical Volume Management. It allows taking snapshots and easier partition resizing.

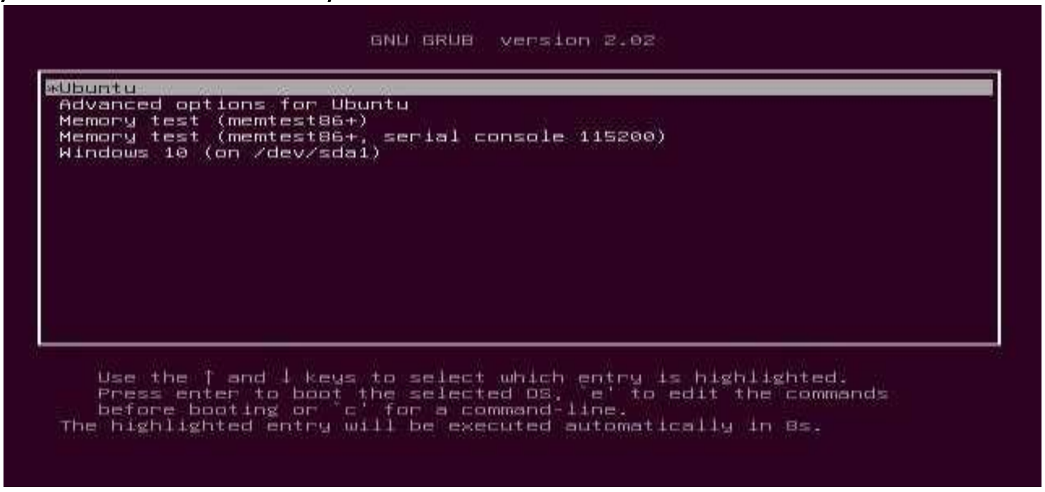
( ) Something else
You can create or resize partitions yourself, or choose multiple partitions for Ubuntu.

Step 12: Now you write your username and password and click continue:



Your name: Nicky ✓
Your computer's name: nicky-laptop ✓
The name it uses when it talks to other computers.
Pick a username: nicky ✓
Choose a password: ●●●●●●●●● Fair password
Confirm your password: ●●●●●●●●● ✓
[✓ Log in automatically]
Require my password to log in
☐ Encrypt my home folder

Step 13: Follow the instructions to complete the installation.
Step14:Reboot your computer when prompted. If you have more than one OS in your system, you will be taken to a GNU GRUB screen after rebooting. This screen allows you to select which OS you want to boot

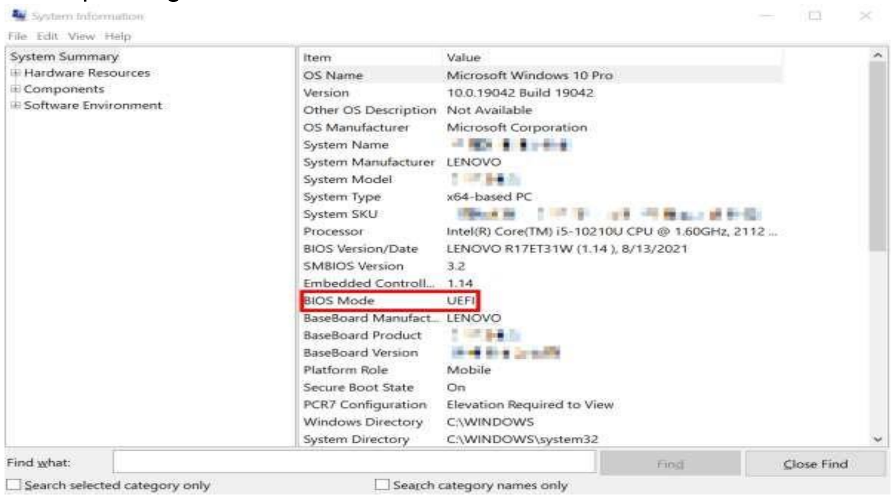


```
                    GNU GRUB  version 2.02

*Ubuntu
 Advanced options for Ubuntu
 Memory test (memtest86+)
 Memory test (memtest86+, serial console 115200)
 Windows 10 (on /dev/sda1)

     Use the ↑ and ↓ keys to select which entry is highlighted.
     Press enter to boot the selected OS, `e' to edit the commands
     before booting or `c' for a command-line.
  The highlighted entry will be executed automatically in 8s.
```

3. DUAL BOOT INSTALLATION OF OPERATING SYSTEM

Step 1: Prepare Windows

First, check whether your system is set up for dual booting at all. For this UEFI is needed, the **further development of BIOS**. Newer PCs and laptops should be equipped with the interface in any case. To be sure, you can simply call up the system information. You should find the corresponding information under the item “BIOS Mode”.



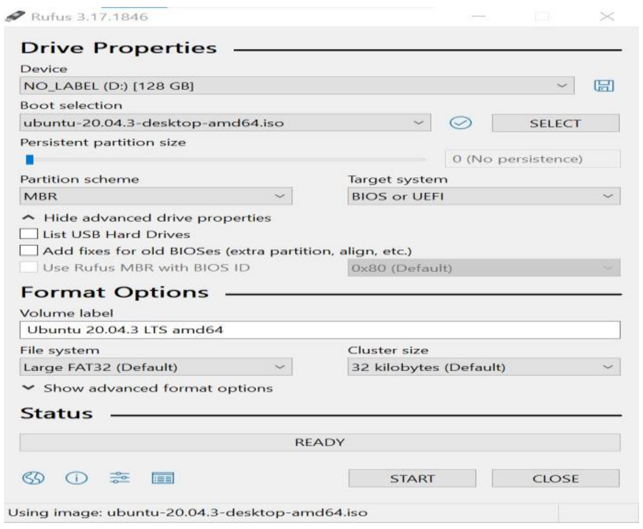
Item	Value
OS Name	Microsoft Windows 10 Pro
Version	10.0.19042 Build 19042
Other OS Description	Not Available
OS Manufacturer	Microsoft Corporation
System Name	[redacted]
System Manufacturer	LENOVO
System Model	[redacted]
System Type	x64-based PC
System SKU	[redacted]
Processor	Intel(R) Core(TM) i5-10210U CPU @ 1.60GHz, 2112 ...
BIOS Version/Date	LENOVO R17ET31W (1.14), 8/13/2021
SMBIOS Version	3.2
Embedded Controll...	1.14
BIOS Mode	UEFI
BaseBoard Manufact...	LENOVO
BaseBoard Product	[redacted]
BaseBoard Version	[redacted]
Platform Role	Mobile
Secure Boot State	On
PCR7 Configuration	Elevation Required to View
Windows Directory	C:\WINDOWS
System Directory	C:\WINDOWS\system32

In the system information you can check if your system uses UEFI.

If you have been working with your system for a long time, i.e. you have not just set it up again, make a data backup to be on the safe side. If an unexpected error occurs during the installation of the second operating system, important data can also be lost. A **backup in the cloud** is a good option. Finally, you will also need storage for the Ubuntu installation. It’s best if the second operating system gets its own hard drive. If only one is available, you’ll need to **create a new partition**. Plan at least 20 GB here, so that you can also work well with Ubuntu. More is definitely better, though.

Step 2: Prepare a boot stick

Now you create a bootable USB stick. First you need an empty USB stick. Then download the ISO image of Ubuntu (or a comparable distribution). Now you need some software to make the USB stick bootable. The tool called Rufus is often used for this. With this, only a few clicks are needed to load the ISO file correctly onto the stick.



Drive Properties

Device: NO_LABEL (D:) [128 GB]

Boot selection: ubuntu-20.04.3-desktop-amd64.iso

Persistent partition size: 0 (No persistence)

Partition scheme: MBR

Target system: BIOS or UEFI

☐ List USB Hard Drives

☐ Add fixes for old BIOSes (extra partition, align, etc.)

☐ Use Rufus MBR with BIOS ID: 0x80 (Default)

Format Options

Volume label: Ubuntu 20.04.3 LTS amd64

File system: Large FAT32 (Default)

Cluster size: 32 kilobytes (Default)

Status

READY

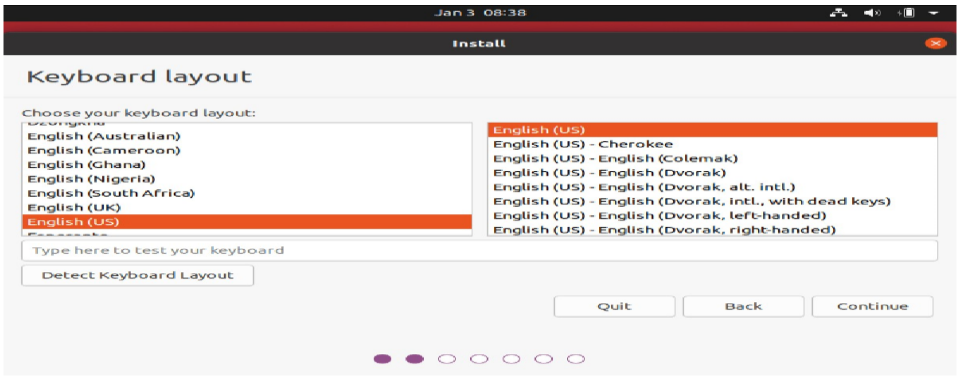
Using image: ubuntu-20.04.3-desktop-amd64.iso

Tools like Rufus can be used to make a USB stick bootable.

You may also need to set the **correct boot order in UEFI**. To do this, start the BIOS and set it to boot from a USB stick first.

Step 3: Install Ubuntu

Now that you have made all the preparations, insert the USB stick into the PC or laptop and restart the system. The BIOS will now access the USB stick and launch the setup application of Ubuntu.



Keyboard layout

Choose your keyboard layout:

English (Australian)
English (Cameroon)
English (Ghana)
English (Nigeria)
English (South Africa)
English (UK)
English (US)

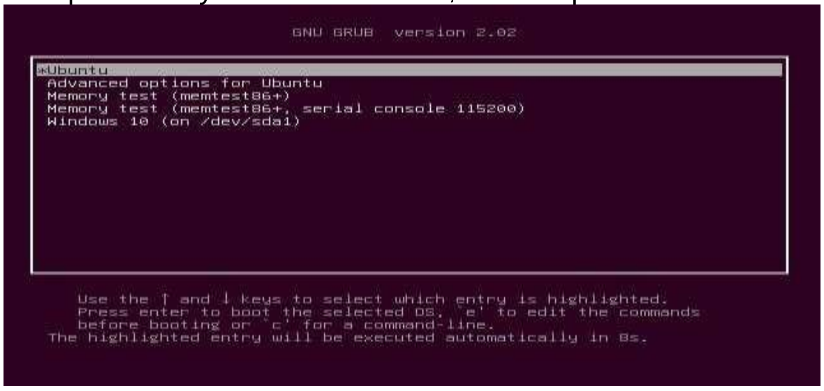
English (US)
English (US) - Cherokee
English (US) - English (Colemak)
English (US) - English (Dvorak)
English (US) - English (Dvorak, alt. intl.)
English (US) - English (Dvorak, intl., with dead keys)
English (US) - English (Dvorak, left-handed)
English (US) - English (Dvorak, right-handed)

The wizard guides you through the installation of Ubuntu.

Follow the instructions of the installer until the software asks you if you want to install Ubuntu in addition to Windows 10 (or Windows Boot Manager). Confirm this. In the next step, you just need to verify that the software has chosen the correct partition for the installation. When you have completed the setup, **both operating systems are installed in parallel**. Now remove the USB stick again, so that the system will not try to boot from it at the next startup.

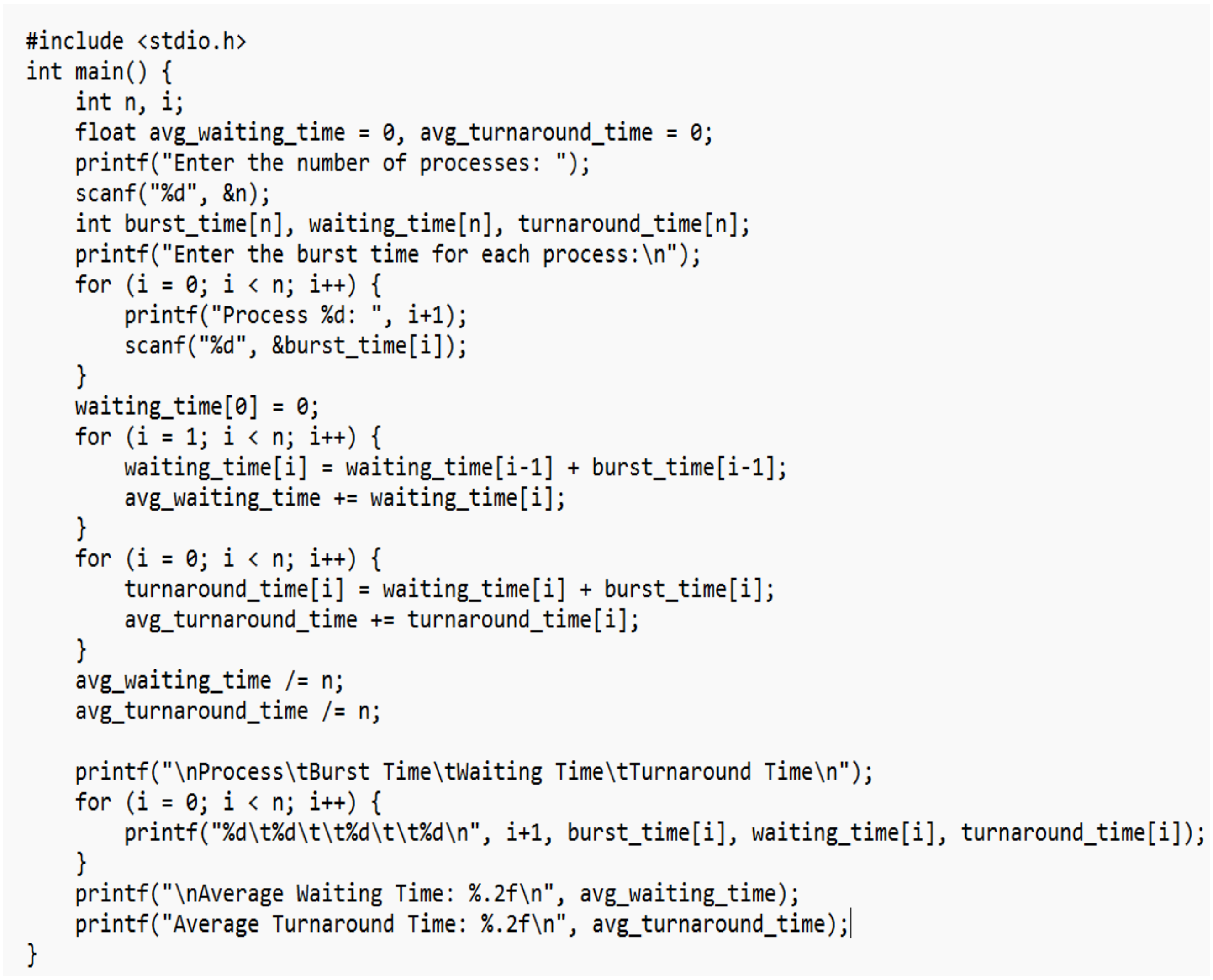
Step 4: Select operating system

Now that you have two equivalent operating systems installed on one PC or laptop, you have to select whether you want to work with Windows or Ubuntu every time you (re)start. This menu is provided by the **Grub** software, which is part of Ubuntu.



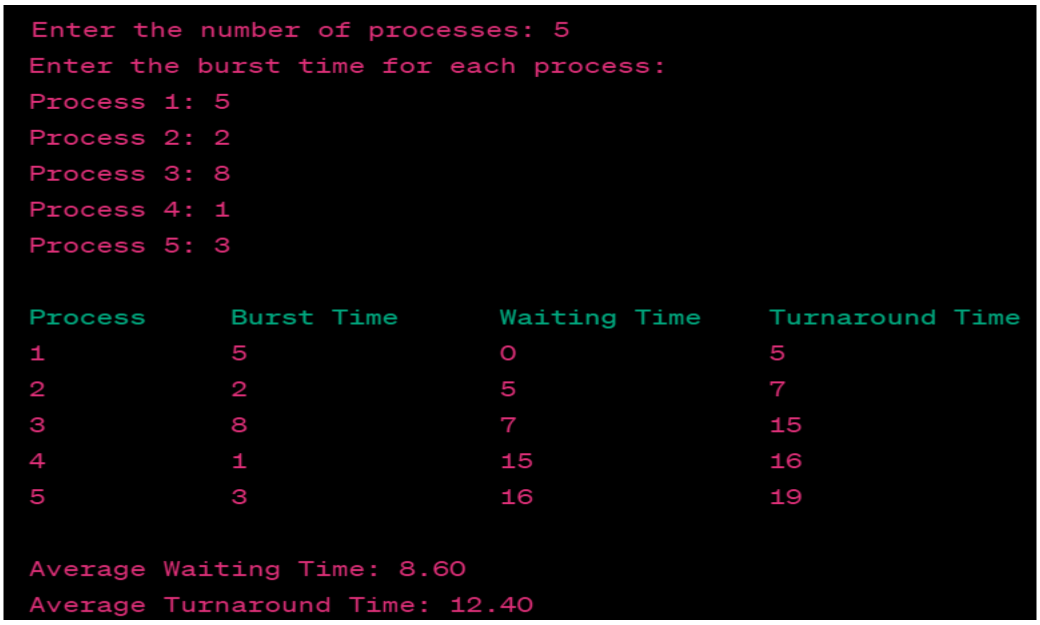
```
GNU GRUB  version 2.02

*Ubuntu
 Advanced options for Ubuntu
 Memory test (memtest86+)
 Memory test (memtest86+, serial console 115200)
 Windows 10 (on /dev/sda1)

    Use the ↑ and ↓ keys to select which entry is highlighted.
    Press enter to boot the selected OS, `e' to edit the commands
    before booting or `c' for a command-line.
 The highlighted entry will be executed automatically in 8s.
```

4.Implementation of FCFS Scheduling algorithm



```c
#include <stdio.h>
int main() {
    int n, i;
    float avg_waiting_time = 0, avg_turnaround_time = 0;
    printf("Enter the number of processes: ");
    scanf("%d", &n);
    int burst_time[n], waiting_time[n], turnaround_time[n];
    printf("Enter the burst time for each process:\n");
    for (i = 0; i < n; i++) {
        printf("Process %d: ", i+1);
        scanf("%d", &burst_time[i]);
    }
    waiting_time[0] = 0;
    for (i = 1; i < n; i++) {
        waiting_time[i] = waiting_time[i-1] + burst_time[i-1];
        avg_waiting_time += waiting_time[i];
    }
    for (i = 0; i < n; i++) {
        turnaround_time[i] = waiting_time[i] + burst_time[i];
        avg_turnaround_time += turnaround_time[i];
    }
    avg_waiting_time /= n;
    avg_turnaround_time /= n;

    printf("\nProcess\tBurst Time\tWaiting Time\tTurnaround Time\n");
    for (i = 0; i < n; i++) {
        printf("%d\t%d\t\t%d\t\t%d\n", i+1, burst_time[i], waiting_time[i], turnaround_time[i]);
    }
    printf("\nAverage Waiting Time: %.2f\n", avg_waiting_time);
    printf("Average Turnaround Time: %.2f\n", avg_turnaround_time);
}
```

OUTPUT :

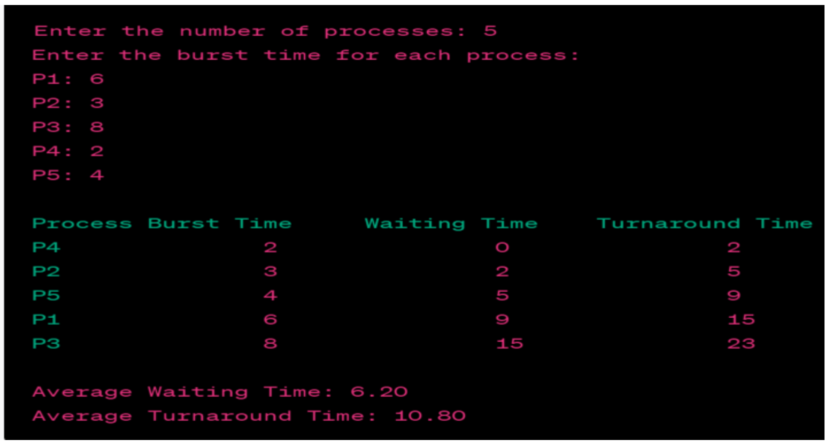


```
Enter the number of processes: 5
Enter the burst time for each process:
Process 1: 5
Process 2: 2
Process 3: 8
Process 4: 1
Process 5: 3

Process     Burst Time      Waiting Time    Turnaround Time
1           5               0               5
2           2               5               7
3           8               7               15
4           1               15              16
5           3               16              19

Average Waiting Time: 8.60
Average Turnaround Time: 12.40
```

5.Implementation of SJF Scheduling algorithm

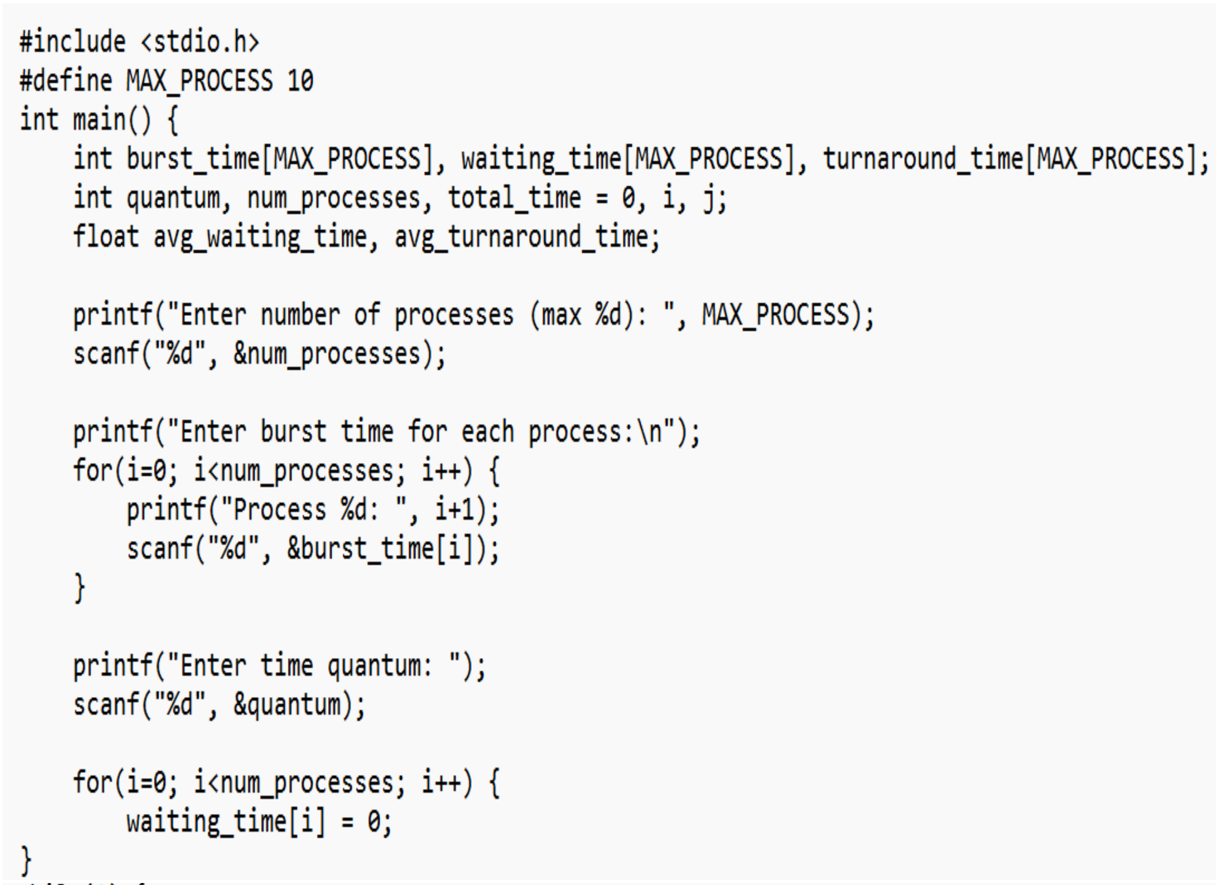
```
#include <stdio.h>
int main() {
    int n, i, j, temp;
    float avg_wt = 0, avg_tat = 0;
    printf("Enter the number of processes: ");
    scanf("%d", &n);
    int bt[n], wt[n], tat[n], p[n];
    printf("Enter the burst time for each process:\n");
    for (i = 0; i < n; i++) {
        printf("P%d: ", i + 1);
        scanf("%d", &bt[i]);
        p[i] = i + 1; }
    for (i = 0; i < n; i++) {
        for (j = i + 1; j < n; j++) {
            if (bt[i] > bt[j]) {
                temp = bt[i];
                bt[i] = bt[j];
                bt[j] = temp;

                temp = p[i];
                p[i] = p[j];
                p[j] = temp;
            } }}
    wt[0] = 0;
    for (i = 1; i < n; i++) {
        wt[i] = 0;
        for (j = 0; j < i; j++) {
            wt[i] += bt[j];
        } }
    printf("\nProcess\tBurst Time\tWaiting Time\tTurnaround Time");
    for (i = 0; i < n; i++) {
        tat[i] = bt[i] + wt[i];
        avg_wt += wt[i];
        avg_tat += tat[i];
        printf("\nP%d\t\t%d\t\t%d\t\t%d", p[i], bt[i], wt[i], tat[i]);
    }
  avg_wt /= n;
    avg_tat /= n;
    printf("\n\nAverage Waiting Time: %.2f", avg_wt);
    printf("\nAverage Turnaround Time: %.2f", avg_tat);

}
```

Output :

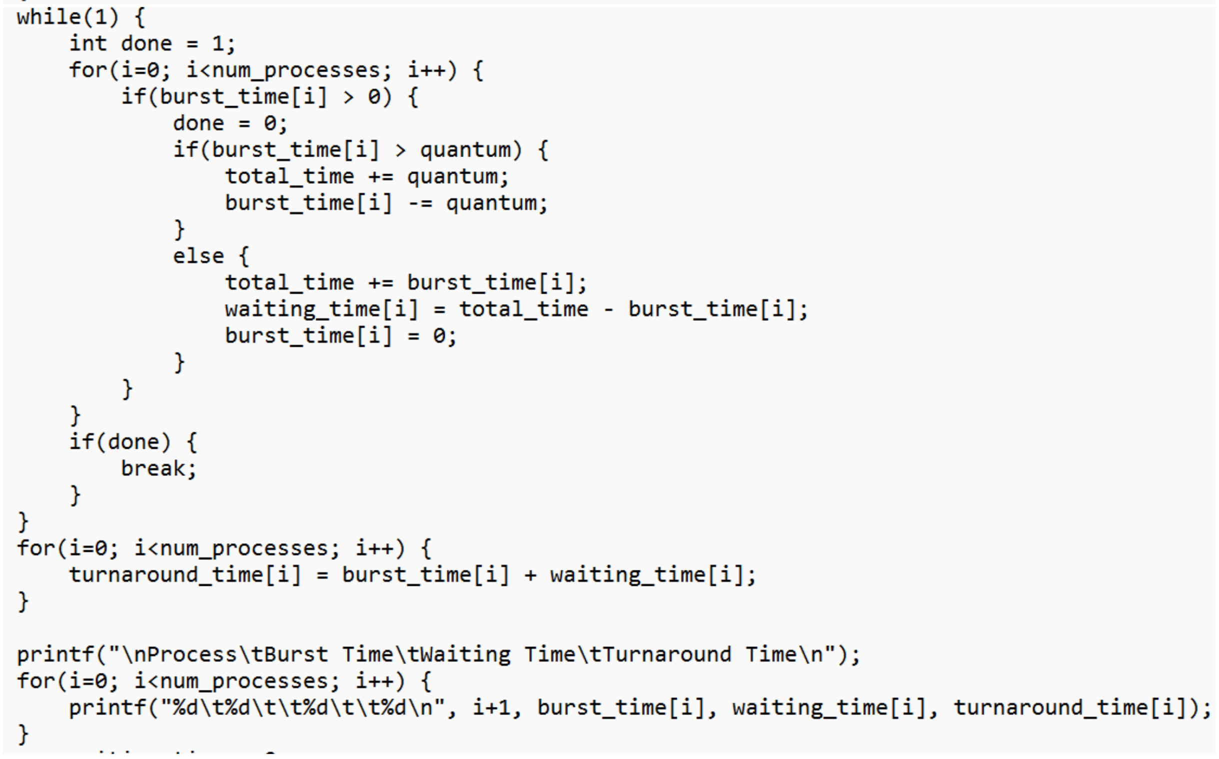


```
Enter the number of processes: 5
Enter the burst time for each process:
P1: 6
P2: 3
P3: 8
P4: 2
P5: 4

Process Burst Time      Waiting Time     Turnaround Time
P4              2               0               2
P2              3               2               5
P5              4               5               9
P1              6               9               15
P3              8               15              23

Average Waiting Time: 6.20
Average Turnaround Time: 10.80
```

6.Implementation of Round-Robin Scheduling algorithm



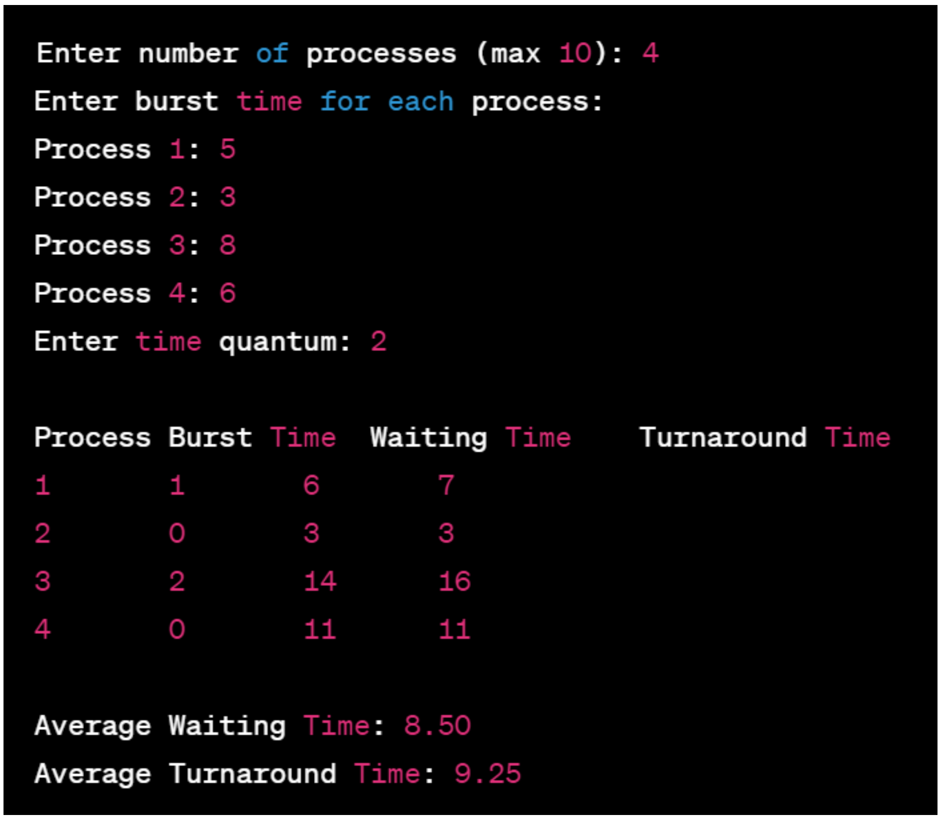
```
#include <stdio.h>
#define MAX_PROCESS 10
int main() {
    int burst_time[MAX_PROCESS], waiting_time[MAX_PROCESS], turnaround_time[MAX_PROCESS];
    int quantum, num_processes, total_time = 0, i, j;
    float avg_waiting_time, avg_turnaround_time;

    printf("Enter number of processes (max %d): ", MAX_PROCESS);
    scanf("%d", &num_processes);

    printf("Enter burst time for each process:\n");
    for(i=0; i<num_processes; i++) {
        printf("Process %d: ", i+1);
        scanf("%d", &burst_time[i]);
    }

    printf("Enter time quantum: ");
    scanf("%d", &quantum);

    for(i=0; i<num_processes; i++) {
        waiting_time[i] = 0;
}
```



```
while(1) {
    int done = 1;
    for(i=0; i<num_processes; i++) {
        if(burst_time[i] > 0) {
            done = 0;
            if(burst_time[i] > quantum) {
                total_time += quantum;
                burst_time[i] -= quantum;
            }
            else {
                total_time += burst_time[i];
                waiting_time[i] = total_time - burst_time[i];
                burst_time[i] = 0;
            }
        }
    }
    if(done) {
        break;
    }
}
for(i=0; i<num_processes; i++) {
    turnaround_time[i] = burst_time[i] + waiting_time[i];
}

printf("\nProcess\tBurst Time\tWaiting Time\tTurnaround Time\n");
for(i=0; i<num_processes; i++) {
    printf("%d\t%d\t\t%d\t\t%d\n", i+1, burst_time[i], waiting_time[i], turnaround_time[i]);
}
```

```
avg_waiting_time = 0;
avg_turnaround_time = 0;
for(i=0; i<num_processes; i++) {
     avg_waiting_time += waiting_time[i];
     avg_turnaround_time += turnaround_time[i];
}
avg_waiting_time /= num_processes;
avg_turnaround_time /= num_processes;
printf("\nAverage Waiting Time: %.2f", avg_waiting_time);
printf("\nAverage Turnaround Time: %.2f", avg_turnaround_time);
}
```

Output :



```
Enter number of processes (max 10): 4
Enter burst time for each process:
Process 1: 5
Process 2: 3
Process 3: 8
Process 4: 6
Enter time quantum: 2

Process Burst Time  Waiting Time    Turnaround Time
1       1       6       7
2       0       3       3
3       2       14      16
4       0       11      11

Average Waiting Time: 8.50
Average Turnaround Time: 9.25
```

7. VI EDITOR AND ITS COMMANDS

The vi editor is a widely used text editor in the Unix/Linux operating systems. It provides a wide range of commands for text manipulation, search, and replace. Here are some of the basic commands:

1 Opening and saving files:

- To open a file: vi filename
- To save a file: Press the Esc key to enter command mode, then type :w and press Enter
- To save and exit: Press Esc to enter command mode, then type :wq and press Enter

2.Moving the cursor: Use the arrow keys to move the cursor around the file. Alternatively, you can use the following commands:
- `h` - Move left
- `j` - Move down
- `k` - Move up
- `l` - Move right

3. Editing text: To insert text, move the cursor to the location where you want to insert text and press `i`. This will put you into "insert mode". Type the text you want to insert, and when you're done, press `Esc` to return to "command mode".

4. Saving and quitting: To save changes and quit Vi, press `Esc` to enter "command mode", then type `:wq` and press `Enter`. If you want to quit without saving changes, type `:q!` instead.

5. Searching for text: To search for text within the file, press `Esc` to enter "command mode", then type `/` followed by the text you want to search for and press `Enter`. Vi will search for the text and highlight the first occurrence.

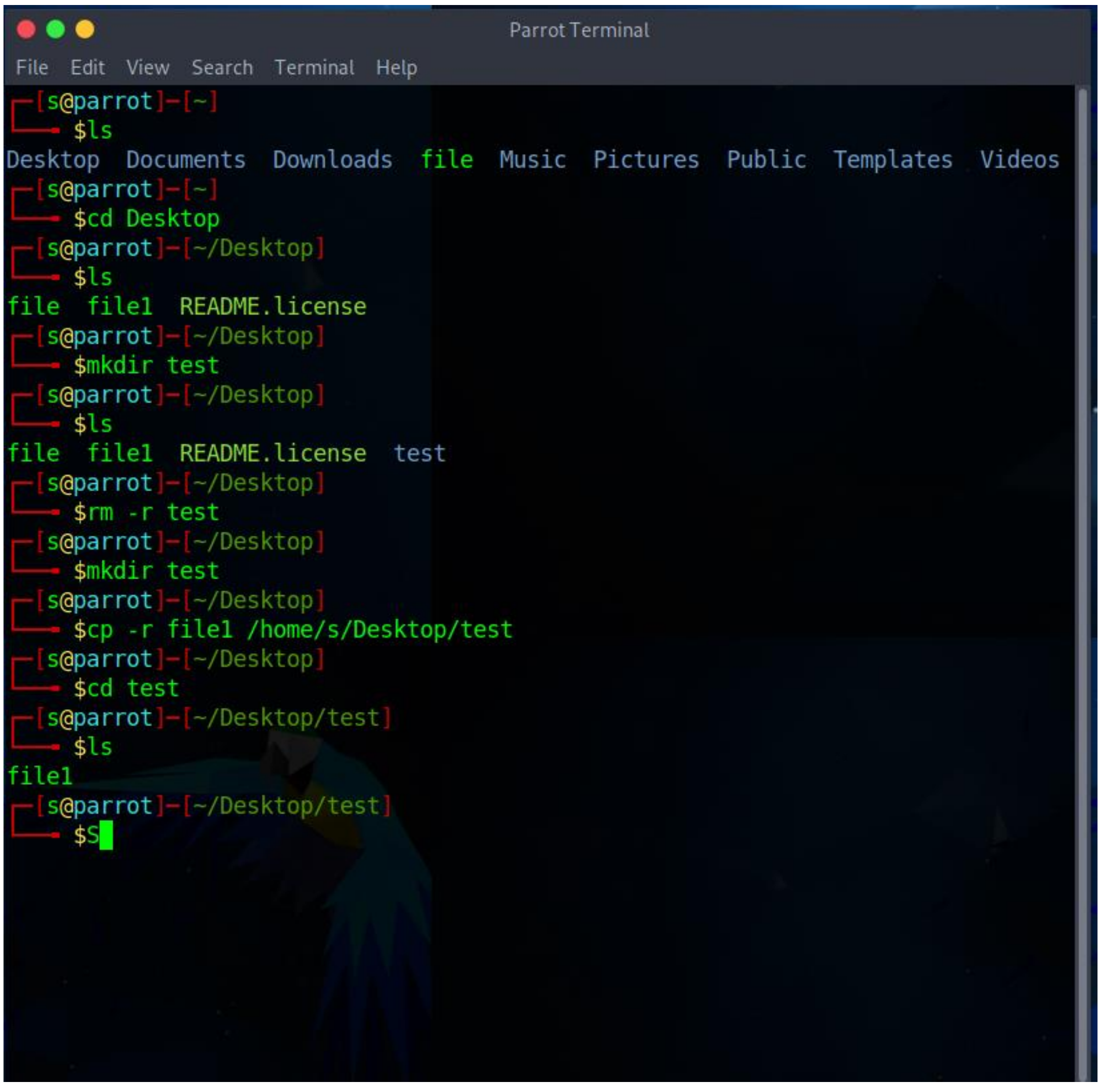
6. Undo and redo: To undo the last change, press `Esc` to enter "command mode", then type `u`. To redo the last change, press `Ctrl`+`r`.

These are just a few basic commands to get you started with Vi. There are many more advanced commands and features available, but mastering the basics will get you a long way.

8. SHELL COMMANDS

Shell commands are used to interact with the shell or command-line interface of an operating system. Here are some basic shell commands that are commonly used:

1. `cd` - Change directory. This command is used to change the current working directory.
2. `ls` - List directory contents. This command is used to list the files and directories in the current directory.
3. `mkdir` - Make directory. This command is used to create a new directory.
4. `rm` - Remove file. This command is used to delete a file.
5. `rmdir` - Remove directory. This command is used to delete a directory.
6. `cp` - Copy file. This command is used to copy a file from one location to another.

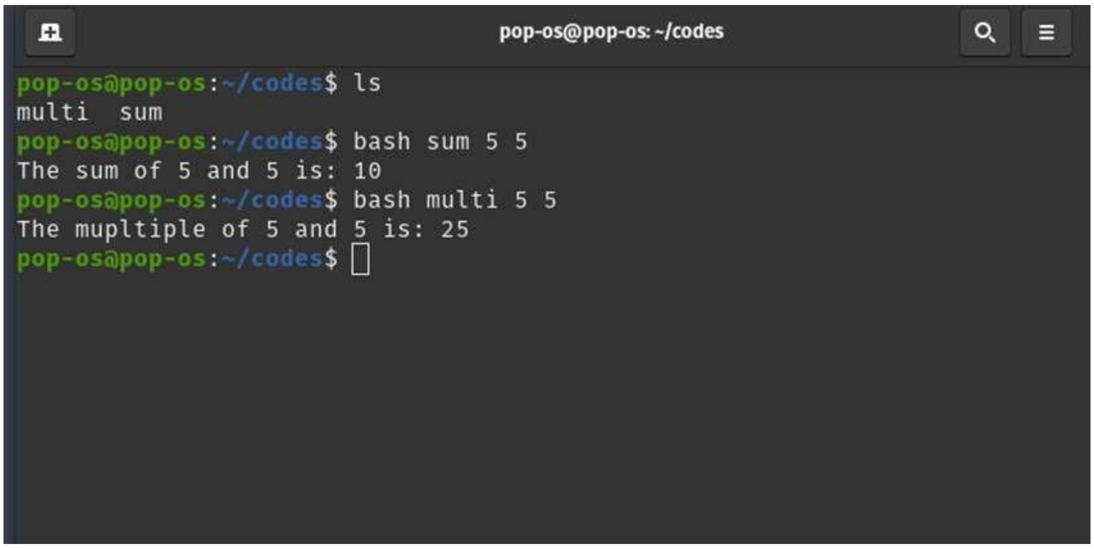


```
┌─[s@parrot]─[~]
└──╼ $ls
Desktop  Documents  Downloads  file  Music  Pictures  Public  Templates  Videos
┌─[s@parrot]─[~]
└──╼ $cd Desktop
┌─[s@parrot]─[~/Desktop]
└──╼ $ls
file  file1  README.license
┌─[s@parrot]─[~/Desktop]
└──╼ $mkdir test
┌─[s@parrot]─[~/Desktop]
└──╼ $ls
file  file1  README.license  test
┌─[s@parrot]─[~/Desktop]
└──╼ $rm -r test
┌─[s@parrot]─[~/Desktop]
└──╼ $mkdir test
┌─[s@parrot]─[~/Desktop]
└──╼ $cp -r file1 /home/s/Desktop/test
┌─[s@parrot]─[~/Desktop]
└──╼ $cd test
┌─[s@parrot]─[~/Desktop/test]
└──╼ $ls
file1
┌─[s@parrot]─[~/Desktop/test]
└──╼ $S
```

9. SHELL SCRIPTING – USING VARIABLES

```
if [ $# -ne 2 ]; then   echo
"Usage: $0 num1 num2"   exit
1 fi  num1=$1 num2=$2
sum=$((num1 + num2))
 echo "The sum of $num1 and $num2 is:
$sum"
```

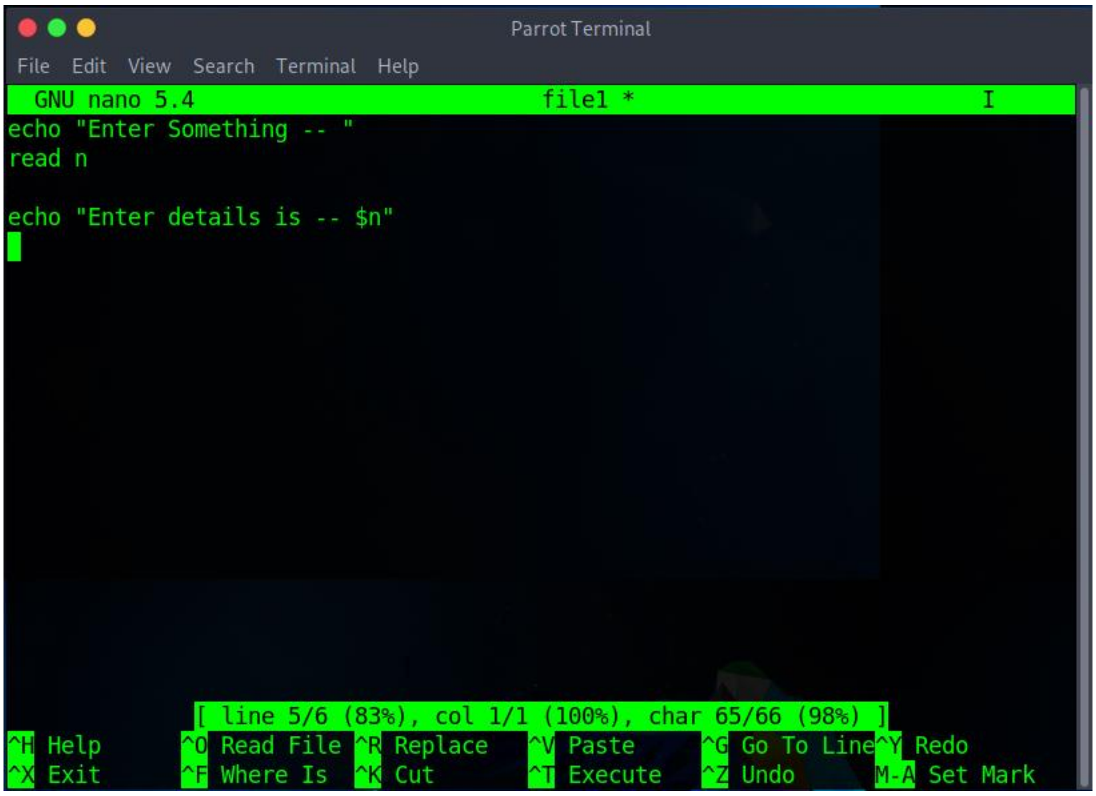
OUTPUT :



pop-os@pop-os: ~/codes

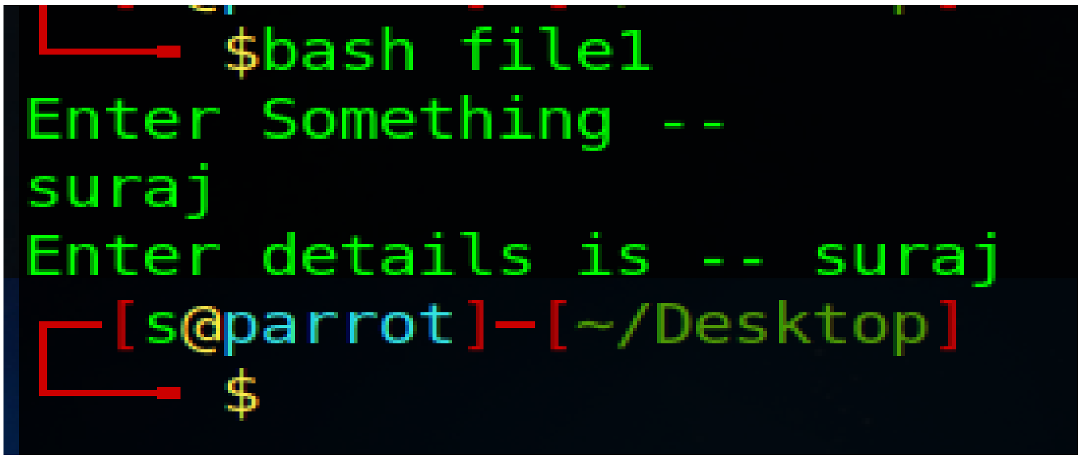
```
pop-os@pop-os:~/codes$ ls
multi  sum
pop-os@pop-os:~/codes$ bash sum 5 5
The sum of 5 and 5 is: 10
pop-os@pop-os:~/codes$ bash multi 5 5
The mupltiple of 5 and 5 is: 25
pop-os@pop-os:~/codes$ 
```

10. SHELL SCRIPTING – INPUT & OUTPUT



```
  GNU nano 5.4                       file1 *
echo "Enter Something -- "
read n

echo "Enter details is -- $n"

              [ line 5/6 (83%), col 1/1 (100%), char 65/66 (98%) ]
^H Help      ^O Read File ^R Replace   ^V Paste     ^G Go To Line^Y Redo
^X Exit      ^F Where Is  ^K Cut       ^T Execute   ^Z Undo      M-A Set Mark
```

OUTPUT :



```
└──╼ $bash file1
Enter Something --
suraj
Enter details is -- suraj
┌─[s@parrot]─[~/Desktop]
└──╼ $
```

11. SHELL SCRIPTING – DATA TYPES

Here are some of the data types used in shell scripting:

String: A sequence of characters enclosed in quotes, either single quotes or double quotes. Example: string="Hello World"

Integer: A numerical value that can be used in mathematical operations. There is no need to declare the variable type in shell scripting. Example: integer=10

Float: A numerical value that can contain a decimal point. Shell scripting does not have built-in support for floating point arithmetic, but it can be accomplished using external tools or libraries. Example: float=3.14

Array: A collection of values that can be accessed using an index number. To create an array, use the syntax array=(value1 value2 value3). To access a specific element of the array, use the syntax ${array[index]}. Example: fruits=("apple" "banana" "orange") and ${fruits[1]} would output "banana".

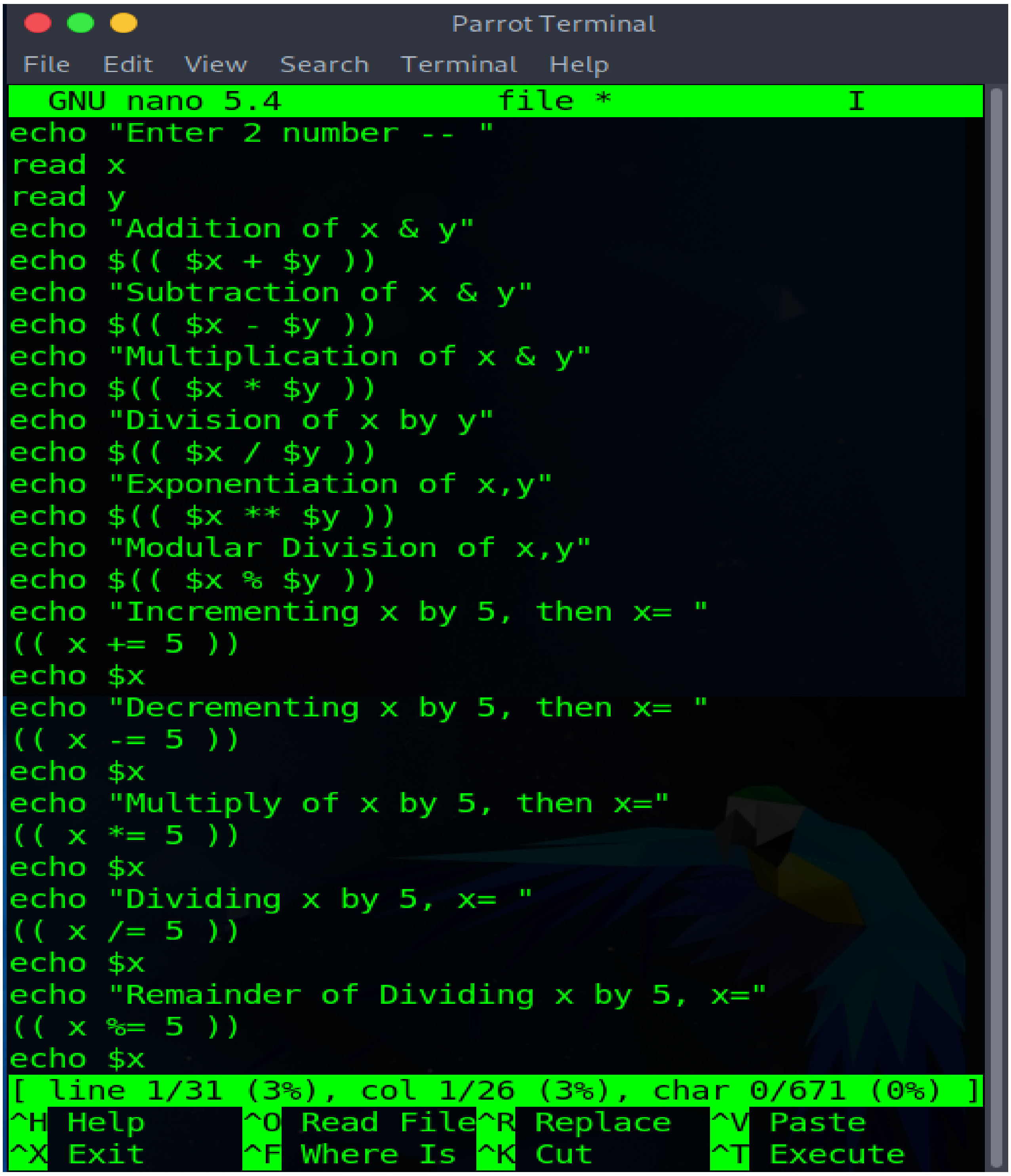
Boolean: A data type that can only have one of two values, either true or false. In shell scripting, true is represented by a non-zero value and false is represented by a zero value. Example: boolean=true

Null: A data type that represents the absence of a value. In shell scripting, a variable can be assigned a null value using the syntax variable=null.

12. SHELL SCRIPTING – USE ARITHMETIC OPERATORS

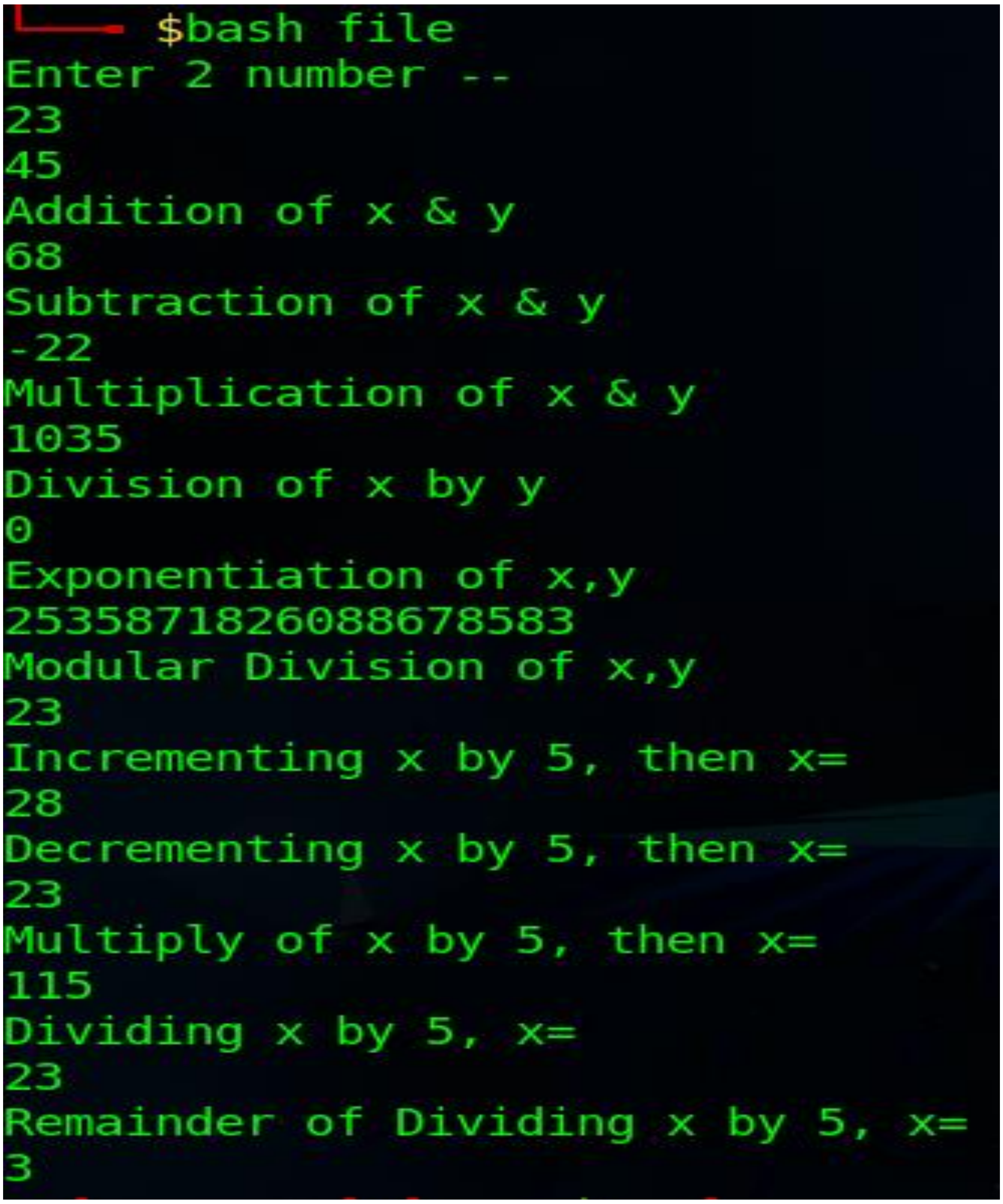
In shell scripting, arithmetic operators are used to perform mathematical operations on numeric values. The shell provides several arithmetic operators, including:

Here's an example that demonstrates the use of arithmetic operators in shell scripting:



```
  GNU nano 5.4                file *                  
echo "Enter 2 number -- "
read x
read y
echo "Addition of x & y"
echo $(( $x + $y ))
echo "Subtraction of x & y"
echo $(( $x - $y ))
echo "Multiplication of x & y"
echo $(( $x * $y ))
echo "Division of x by y"
echo $(( $x / $y ))
echo "Exponentiation of x,y"
echo $(( $x ** $y ))
echo "Modular Division of x,y"
echo $(( $x % $y ))
echo "Incrementing x by 5, then x= "
(( x += 5 ))
echo $x
echo "Decrementing x by 5, then x= "
(( x -= 5 ))
echo $x
echo "Multiply of x by 5, then x="
(( x *= 5 ))
echo $x
echo "Dividing x by 5, x= "
(( x /= 5 ))
echo $x
echo "Remainder of Dividing x by 5, x="
(( x %= 5 ))
echo $x
[ line 1/31 (3%), col 1/26 (3%), char 0/671 (0%) ]
^H Help        ^O Read File^R Replace     ^V Paste
^X Exit        ^F Where Is ^K Cut         ^T Execute
```

OUTPUT:

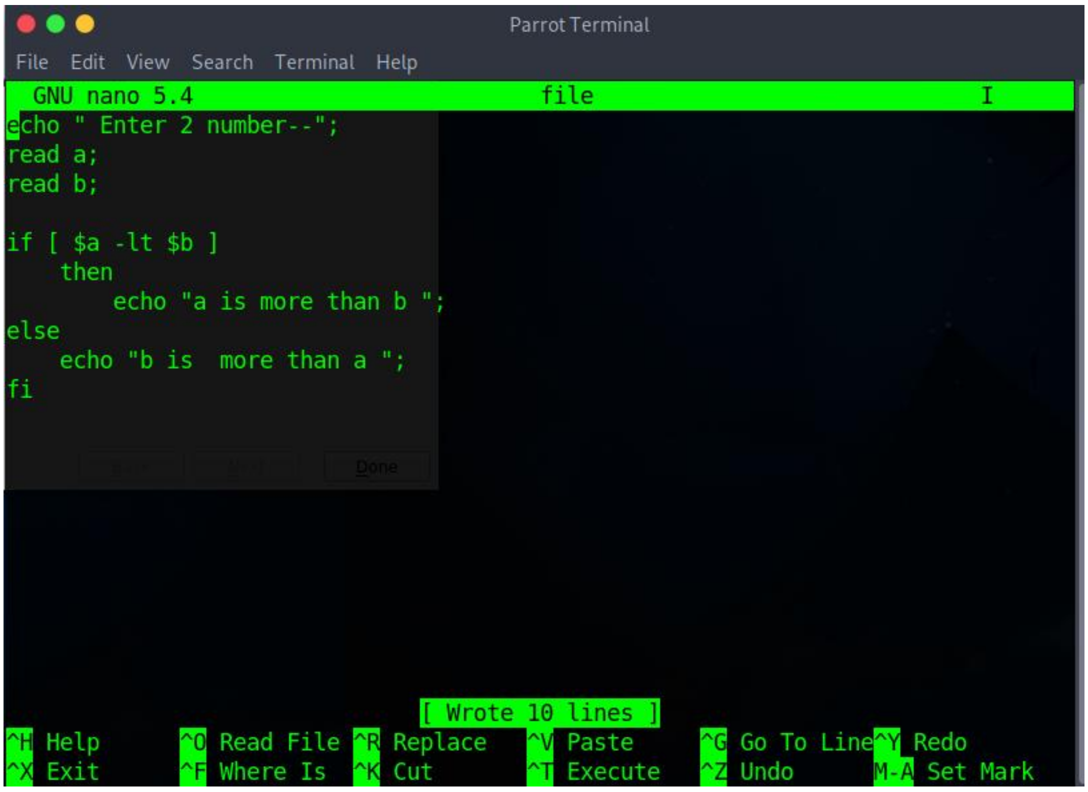


```
└─── $bash file
Enter 2 number --
23
45
Addition of x & y
68
Subtraction of x & y
-22
Multiplication of x & y
1035
Division of x by y
0
Exponentiation of x,y
2535871826088678583
Modular Division of x,y
23
Incrementing x by 5, then x=
28
Decrementing x by 5, then x=
23
Multiply of x by 5, then x=
115
Dividing x by 5, x=
23
Remainder of Dividing x by 5, x=
3
```

13. SHELL SCRIPTING – IF CONTROL STATEMENT PROGRAM

The `if` control statement in shell scripting is used to execute a block of code based on a particular condition. It allows for branching logic in a script, making it possible to execute different code paths based on the result of a condition.

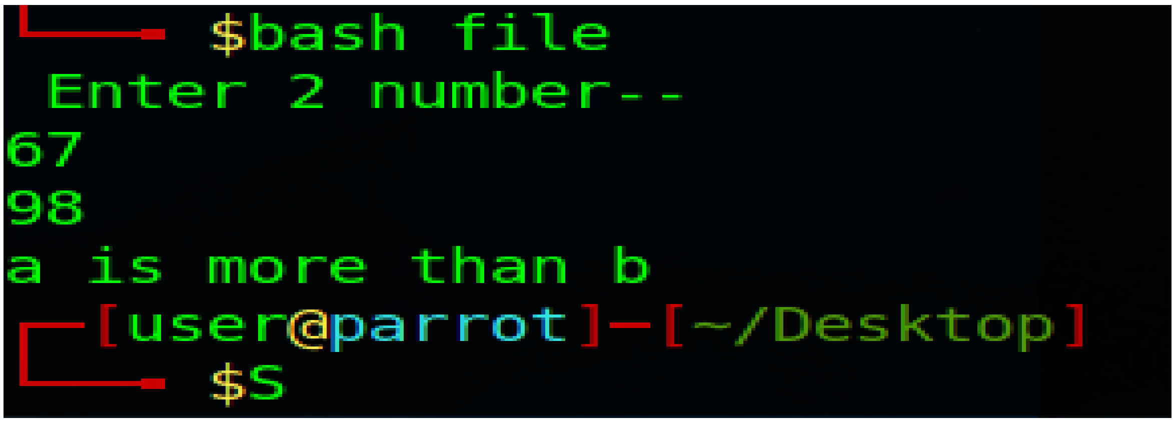
For example



```
echo " Enter 2 number--";
read a;
read b;

if [ $a -lt $b ]
    then
        echo "a is more than b ";
else
    echo "b is  more than a ";
fi
```

```
[ Wrote 10 lines ]
^H Help      ^O Read File ^R Replace  ^V Paste    ^G Go To Line^Y Redo
^X Exit      ^F Where Is  ^K Cut      ^T Execute  ^Z Undo      M-A Set Mark
```

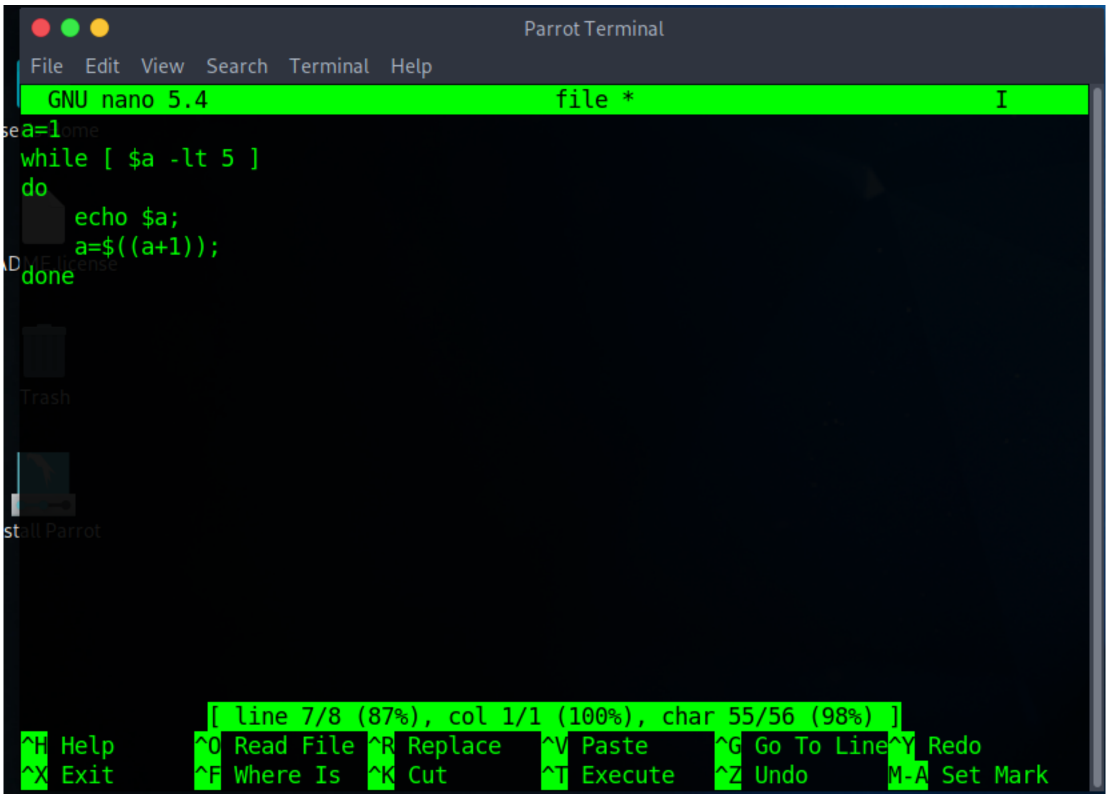
OUTPUT :



```
└──╼ $bash file
 Enter 2 number--
67
98
a is more than b
┌─[user@parrot]─[~/Desktop]
└──╼ $S
```

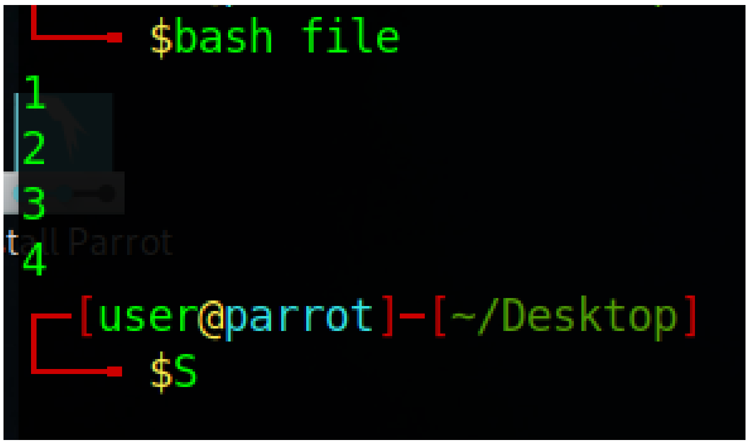
14. SHELL SCRIPTING – WHILE CONTROL STATEMENT

`while` control statement is used in shell scripting to execute a block of code repeatedly while a particular condition is true. Here is an example of a shell script that uses the `while` control statement:



```
GNU nano 5.4                         file *
a=1
while [ $a -lt 5 ]
do
        echo $a;
        a=$((a+1));
done

[ line 7/8 (87%), col 1/1 (100%), char 55/56 (98%) ]
^H Help     ^O Read File ^R Replace  ^V Paste    ^G Go To Line^Y Redo
^X Exit     ^F Where Is  ^K Cut      ^T Execute  ^Z Undo      M-A Set Mark
```

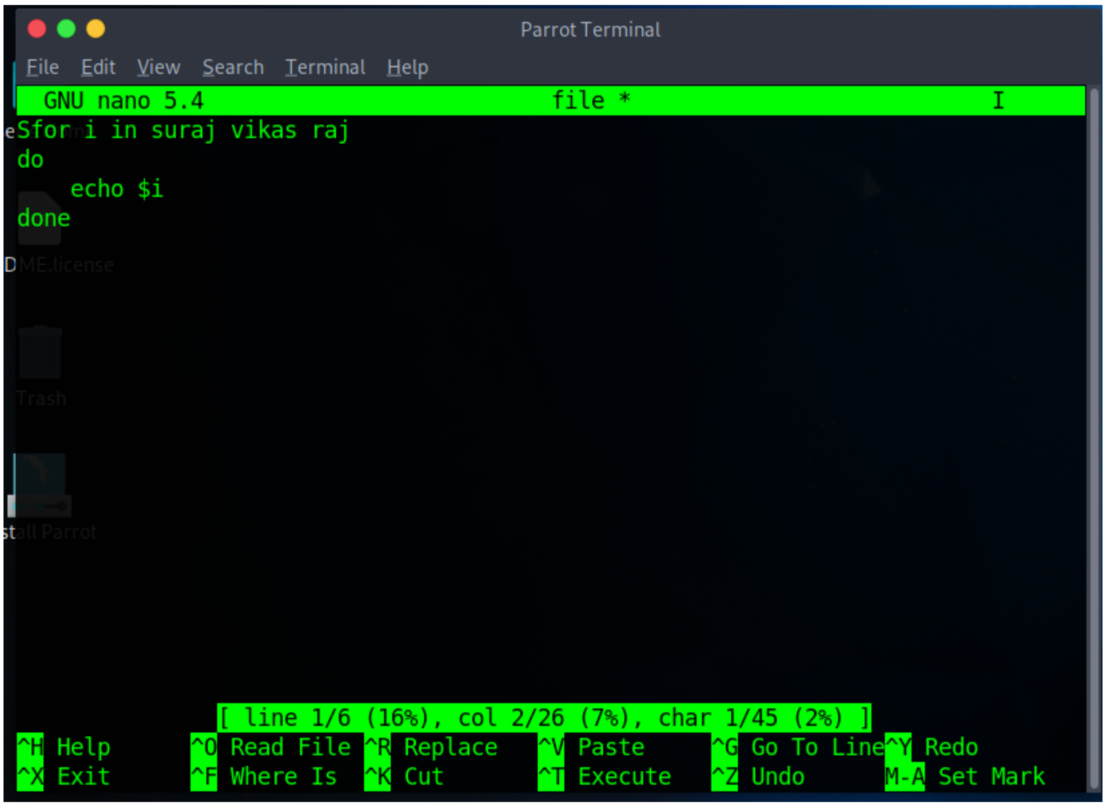
OUTPUT :



```
└──╼ $bash file
1
2
3
4
┌─[user@parrot]─[~/Desktop]
└──╼ $S
```

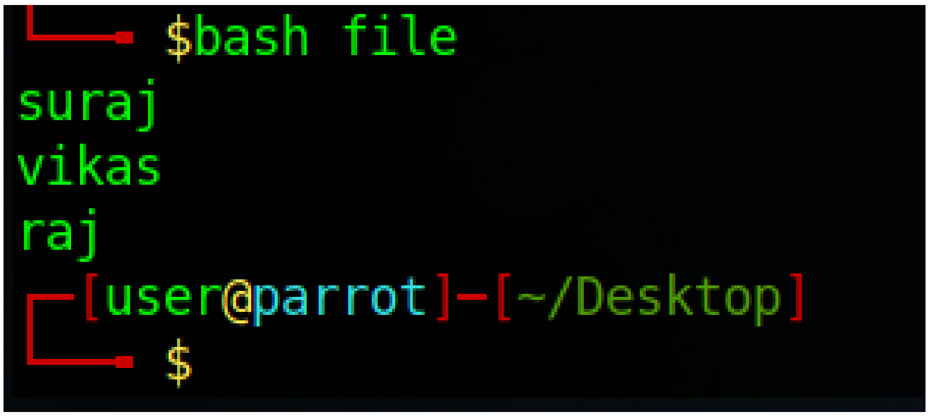
15. SHELL SCRIPTING – FOR CONTROL STATEMENT

The `for` control statement is used in shell scripting to execute a block of code for each item in a list. Here is an example of a shell script that uses the `for` control statement:



```
  GNU nano 5.4                        file *
for i in suraj vikas raj
do
    echo $i
done

                 [ line 1/6 (16%), col 2/26 (7%), char 1/45 (2%) ]
^H Help      ^O Read File ^R Replace   ^V Paste     ^G Go To Line ^Y Redo
^X Exit      ^F Where Is  ^K Cut       ^T Execute   ^Z Undo       M-A Set Mark
```

OUTPUT :



```
└──╼ $bash file
suraj
vikas
raj
┌─[user@parrot]─[~/Desktop]
└──╼ $
```