Name: Shubham .B. Shigavan ***TE ETRX***
UID : 2018110055

Lab 6

Aim: Virtual Memory.

Objectives: This lab introduces simple tools available in Linux which helps in understanding some fundamental concepts related to disk utilisation using the concept of process management and virtual memory.

Part A

Code in cpu.c:

Below code contains a while loop and a for loop which assigns value to a variable repetatively.

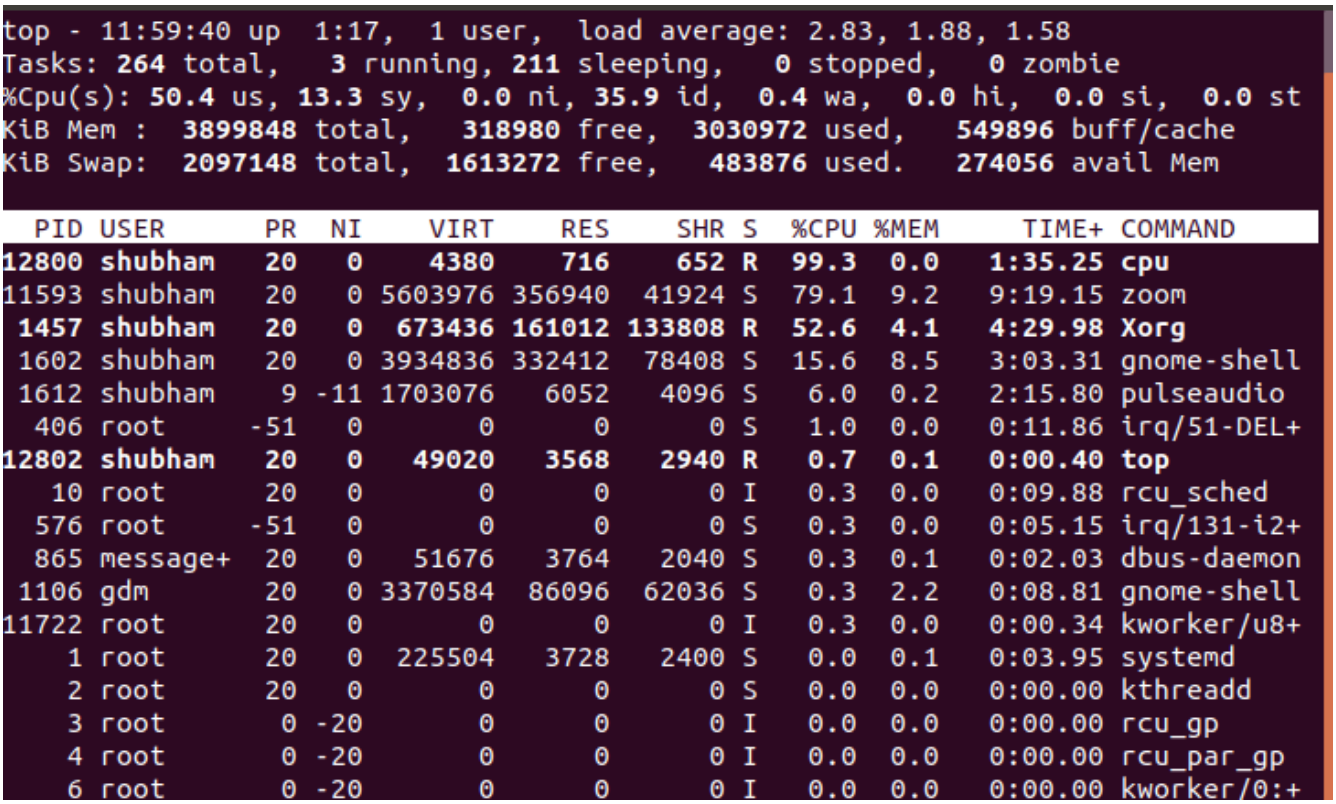
```
#include <unistd.h>
#include <stdio.h>

int main(int argc, char *argv[])

{
  unsigned int i,j;
  while(1)
    {
      j = 1;
     for(i = 1; i <= 10; i++)
          {
            j = j*i;
          }
    }
}
```

Program compiling and execution:

```
shubham@shubham-Vostro-15-3568:~/Desktop$ cd lab_2_related_files
shubham@shubham-Vostro-15-3568:~/Desktop/lab_2_related_files$ gcc cpu.c -o cpu
shubham@shubham-Vostro-15-3568:~/Desktop/lab_2_related_files$ ./cpu
```

Top command execution in other terminal:

In below image you can see the process cpu running at the top.



Questions:

1. What is the PID of the process running the cpu command?

Ans: In the above image you can see the process id (PID) of the cpu.c program execution in the first column which is 12800

2. How much CPU and memory does this process consume?

Ans: In the above image, in the %CPU column we can see the CPU consumption which is around 99.3% and in the %MEM we can see Memory consumption is around 0%.

3. What is the current state of the process? For example, is it running or in a blocked state ?

Ans: Current state of the process can be found out from the “S” column. For cpu process it is R which represents that process is in running state. Here S represents the sleeping and B represents blocked state.

Code in memory1.c :

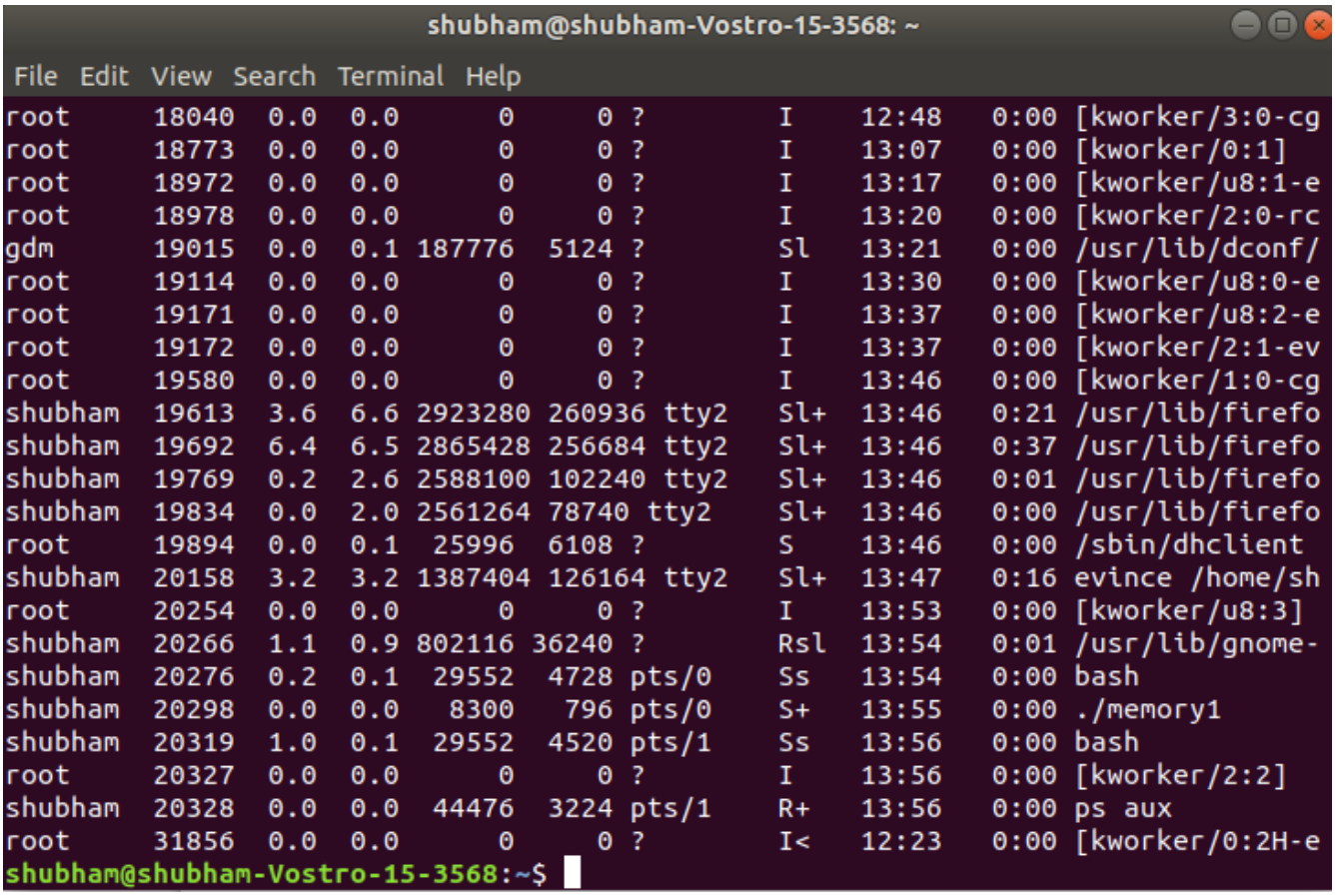
In this program, we are creating array of size 1000000 but not accessing it. We are simply printing size of int and process id of this process.

```
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>

#define ARRAY_SIZE 1000000
int main()
{
        int array[ARRAY_SIZE];
        int i;

        printf("\n\nProgram : 'memory_1'\n");
        printf("_____\n");
        printf ("\n\nPID : %d \n",getpid());
        printf( "Size of int : %ld \n",sizeof(int));

        printf("\nPress Enter Key to exit.\n");

        getchar();
        return 0;
}
```

Program compiling and execution:

```
shubham@shubham-Vostro-15-3568:~$ cd Desktop
shubham@shubham-Vostro-15-3568:~/Desktop$ cd lab_2_related_files
shubham@shubham-Vostro-15-3568:~/Desktop/lab_2_related_files$ gcc memory1.c -o m
emory1
shubham@shubham-Vostro-15-3568:~/Desktop/lab_2_related_files$ ./memory1


Program : 'memory_1'
____________________


PID : 20298
Size of int : 4

Press Enter Key to exit.
```

Checking memory usage of memory1.c:



Here to find the memory consumption of this process we are running program memory1.c in one terminal and running ps aux command in another terminal.

Here-

• The a option tells ps to display the processes of all users. Only the processes that not associated with a terminal and processes of group leaders are not shown.

• u stands for a user-oriented format that provides detailed information about the processes.

• The x option instructs ps to list the processes without a controlling terminal. Those are mainly processes that are started on boot time and running in the background.

So this can be used for determining different parameters of all the processes.
All the columns shows following things:

- ***USER*** - The user who runs the process.
- ***%CPU*** - The cpu utilization of the process.
- ***%MEM*** - The percentage of the process's resident set size to the physical memory on the machine.

- ***VSZ*** - Virtual memory size of the process in KiB.
- ***RSS*** - The size of the physical memory that the process is using.
- ***STAT*** - The the process state code, such as Z (zombie), S (sleeping), and R (running).
- ***START*** - The time when the command started.

In the VSZ column, we can find out the V.M taken by memory1.c which is 8300KB. In the RSS column we can find out actual physical memory taken by process which is 796KB. Since we are just creating array , the physical memory consumed is less.

Code in memory2.c:

```
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>

#define ARRAY_SIZE 1000000

int main()
{
	int array[ARRAY_SIZE];
	int i;

	printf("\n\nProgram : 'memory_2'\n");
	printf("_____\n");
	printf ("\n\nPID : %d \n",getpid());
	printf( "Size of int : %ld \n",sizeof(int));

	for(i=0;i<ARRAY_SIZE/2;i++)
	{
		array[i] = 10;
	}

	for(i=1;i<ARRAY_SIZE/2;i++)
	{
		array[i] = array[i-1]+25;
	}


	printf("\nPress Enter Key to exit.\n");

	getchar();
	return 0;
}
```

In this program, we are creating array of size 1000000. We are also assigning values to array also and we are also printing PID and size of int.

Program compiling and execution:

```
shubham@shubham-Vostro-15-3568:~/Desktop/lab_2_related_files$ gcc memory2.c -o m
emory2
shubham@shubham-Vostro-15-3568:~/Desktop/lab_2_related_files$ ./memory2


Program : 'memory_2'
--------------------


PID : 20467
Size of int : 4

Press Enter Key to exit.
```

Checking memory usage of memory2.c:

```
                                shubham@shubham-Vostro-15-3568: ~
 File  Edit  View  Search  Terminal  Help
root      18040  0.0  0.0      0     0 ?        I    12:48   0:00 [kworker/3:0-cg
root      18773  0.0  0.0      0     0 ?        I    13:07   0:00 [kworker/0:1]
root      18972  0.0  0.0      0     0 ?        I    13:17   0:01 [kworker/u8:1-e
root      18978  0.0  0.0      0     0 ?        I    13:20   0:00 [kworker/2:0-rc
gdm       19015  0.0  0.1 187776  5124 ?        Sl   13:21   0:00 /usr/lib/dconf/
root      19114  0.0  0.0      0     0 ?        D    13:30   0:01 [kworker/u8:0+e
root      19171  0.0  0.0      0     0 ?        I    13:37   0:01 [kworker/u8:2-e
root      19172  0.0  0.0      0     0 ?        I    13:37   0:00 [kworker/2:1-rc
root      19580  0.0  0.0      0     0 ?        I    13:46   0:00 [kworker/1:0-cg
shubham   19613  1.9  6.8 2929944 267932 tty2    Sl+  13:46   0:23 /usr/lib/firefo
shubham   19692  3.4  7.1 2865428 277724 tty2    Sl+  13:46   0:40 /usr/lib/firefo
shubham   19769  0.1  2.6 2588100 102240 tty2    Sl+  13:46   0:01 /usr/lib/firefo
shubham   19834  0.0  2.0 2561264 78740 tty2     Sl+  13:46   0:00 /usr/lib/firefo
root      19894  0.0  0.1  25996  6108 ?        S    13:46   0:00 /sbin/dhclient
shubham   20158  2.7  3.4 1380560 133824 tty2    Sl+  13:47   0:30 evince /home/sh
shubham   20266  0.3  0.9 802276 36488 ?         Ssl  13:54   0:02 /usr/lib/gnome-
shubham   20276  0.0  0.1  29552  4728 pts/0     Ss   13:54   0:00 bash
shubham   20319  0.0  0.1  29552  4700 pts/1     Ss   13:56   0:00 bash
root      20407  0.0  0.0      0     0 ?        I    14:01   0:00 [kworker/2:2-pm
root      20453  0.0  0.0      0     0 ?        I    14:04   0:00 [kworker/u8:3]
shubham   20467  0.0  0.0   8300  3212 pts/0     S+   14:05   0:00 ./memory2
shubham   20481  0.0  0.0  44476  3140 pts/1     R+   14:06   0:00 ps aux
root      31856  0.0  0.0      0     0 ?        I<   12:23   0:00 [kworker/0:2H-e
shubham@shubham-Vostro-15-3568:~$
```

In the VSZ column, we can find out the virtual memory taken by memory1.c which is 8300KB. In the RSS column we can find out actual physical memory taken by process which is 3212KB.

From this we can say that, memory1.c and memory2.c are using almost same virtual memory but memory2.c is using more physical memory than memory1.c. The reason for this is in memory1.c we are just creating array but in memory2.c we are creating array as well as we are assigning the values .

Part B

Code in disk.c and disk1.c and make copies shell script:

make copies shell script:

```
i=-1
while (( i++ < 5000 )); do
  cp ./disk-files/foo.pdf "./disk-files/foo$i.pdf"
done
```

disk.c

```
#include <unistd.h>
#include <stdio.h>
#include <sys/types.h>
#include <string.h>
#include <errno.h>
#include <stdlib.h>

#define FNAME_SIZE 100
#define MAX_FILE_NO 5000
#define BLOCK_SIZE 1024

int main(int argc, char *argv[])
{

  int n, file_no;
  FILE *fp;
  char dest_file_name[FNAME_SIZE];
  char buf[BLOCK_SIZE];

  while(1)
    {
      file_no = rand() % MAX_FILE_NO;

      bzero(dest_file_name, FNAME_SIZE);
      sprintf(dest_file_name, "disk-files/foo%d.pdf", file_no);

      fp = fopen(dest_file_name, "rb");
      if (fp == NULL) {
           perror("Can't open dest file");
           exit(1);
      }

      bzero(buf,BLOCK_SIZE);
      while  ( (n = (int)fread( buf, 1, BLOCK_SIZE, fp )) > 0)
           {
             //do nothing with the read data;
             bzero(buf,BLOCK_SIZE);
```

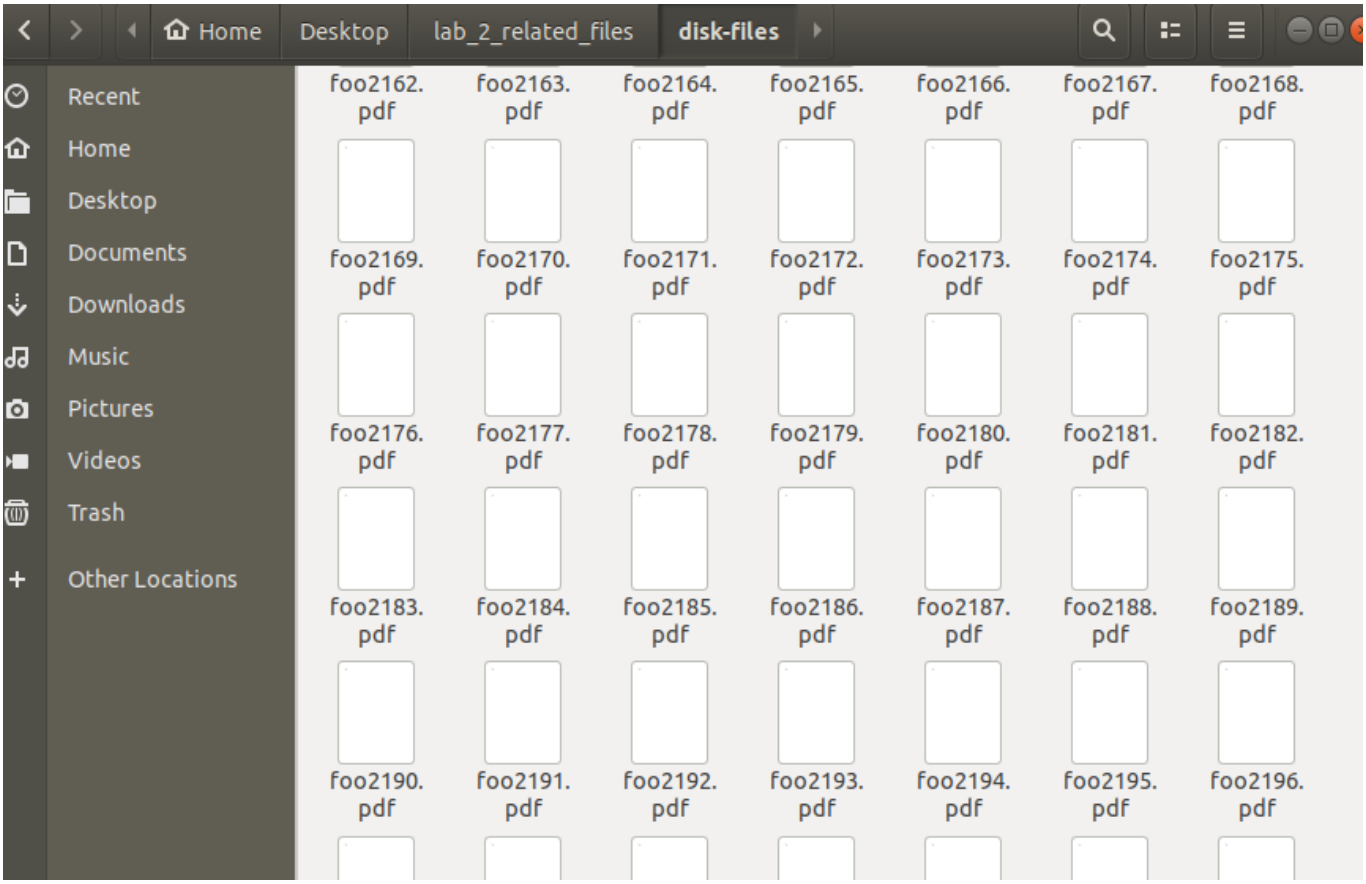
```
        }

      fclose(fp);
    }
}
```

disk1.c

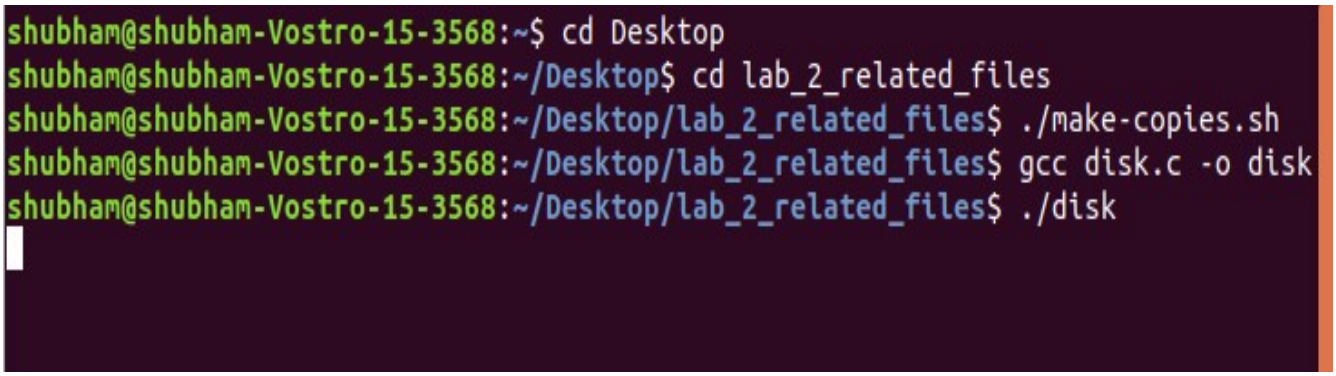
```
#include <unistd.h>
#include <stdio.h>
#include <sys/types.h>
#include <string.h>
#include <errno.h>
#include <stdlib.h>

#define FNAME_SIZE 100
#define MAX_FILE_NO 10000
#define BLOCK_SIZE 1024

int main(int argc, char *argv[])
{

  int n, file_no;
  FILE *fp;
  char dest_file_name[FNAME_SIZE];
  char buf[BLOCK_SIZE];

  while(1)
    {
      //file_no = rand() % MAX_FILE_NO;
      file_no = 0;

      bzero(dest_file_name, FNAME_SIZE);
      sprintf(dest_file_name, "disk-files/foo%d.pdf", file_no);

      fp = fopen(dest_file_name, "rb");
      if (fp == NULL) {
          perror("Can't open dest file");
          exit(1);
      }

      bzero(buf,BLOCK_SIZE);
      while  ( (n = (int)fread( buf, 1, BLOCK_SIZE, fp )) > 0)
          {
            //do nothing with the read data;
            bzero(buf,BLOCK_SIZE);
          }

      fclose(fp);
    }
```

}

After executing the make-copies.sh command:



Executing disk.c program:



Using ps aux command:

Using the “ps” command to view the observed Process ID’ state, it gives the R and S states of the process where R means the running state and S means Sleeping state. The plus sign will indicate if the process is ran by the user in some terminal or in some GUI application i.e. in foreground.

```
root         433  0.0  0.0      0     0 ?        I<   10:41   0:00 [ath10k_aux_wq]
root         471  0.0  0.0      0     0 ?        S    10:41   0:03 [i915/signal:0]
root         472  0.0  0.0      0     0 ?        S    10:41   0:00 [i915/signal:1]
root         473  0.0  0.0      0     0 ?        S    10:41   0:00 [i915/signal:2]
root         474  0.0  0.0      0     0 ?        S    10:41   0:00 [i915/signal:6]
root         479  0.0  0.0      0     0 ?        I<   10:41   0:00 [kmemstick]
root         485  0.0  0.0      0     0 ?        I<   10:41   0:02 [kworker/u9:2-u
shubham      538  101  0.0   4512   772 pts/0    R+   14:17   0:10 ./disk
root         574  0.0  0.0      0     0 ?        I<   10:41   0:00 [rmi4-poller]
root         576  0.1  0.0      0     0 ?        S    10:41   0:19 [irq/131-i2c_hi
shubham      788  1.0  0.1  29552  4460 pts/1    Ss   14:17   0:00 bash
systemd+     812  0.0  0.0  70892  3184 ?        Ss   10:42   0:01 /lib/systemd/sy
root         851  0.0  0.0 434328  3568 ?        Ssl  10:42   0:00 /usr/sbin/Modem
```

Execution of disk1.c:

```
shubham@shubham-Vostro-15-3568:~/Desktop/lab_2_related_files$ gcc disk1.c -o dis
k1
shubham@shubham-Vostro-15-3568:~/Desktop/lab_2_related_files$ ./disk1
```

Again using ps aux command:

```
shubham   2508  0.1  1.1 458004 45212 ?        Ssl  10:48   0:14 /usr/bin/zeitge
shubham   2516  0.0  0.1 330992  6700 ?        Ssl  10:48   0:10 /usr/lib/zeitge
root      5673  0.0  0.0      0     0 ?        I    12:26   0:01 [kworker/0:2-ev
root      6281  0.0  0.0      0     0 ?        I    14:19   0:00 [kworker/u8:1]
shubham   6396 98.3  0.0   4512   716 pts/0    R+   14:19   0:28 ./disk1
shubham   7373  0.0  0.0  44476  3148 pts/1    R+   14:19   0:00 ps aux
shubham  11068  0.0  0.0 152548  3704 tty2     Sl+  11:36   0:00 /usr/lib/libreo
shubham  11088  0.9  4.7 1547388 186844 tty2   Sl+  11:36   1:33 /usr/lib/libreo
shubham  11589  0.0  0.0  24572   916 tty2     Sl+  11:40   0:00 /usr/bin/zoom
shubham  11592  0.0  0.0   4632   664 tty2     S+   11:40   0:00 sh -c export SS
shubham  11593 29.0 11.4 5512696 445528 tty2   SLl+ 11:40  46:17 /opt/zoom/zoom
```

Using iostat command: The iostat command is used for monitoring system input/output device loading by observing the time the devices are active in relation to their average transfer rates. The iostat command generates reports that can be used to change system configuration to better balance the input/output load between physical disks.

```
avg-cpu:  %user   %nice %system %iowait  %steal   %idle
          15.40    0.04   10.60    2.42    0.00   71.54

Device             tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
loop0             0.00         0.01         0.00        121          0
loop1             0.02         0.09         0.00       1254          0
loop2             0.00         0.02         0.00        330          0
loop3             0.02         0.04         0.00        507          0
loop4             0.00         0.01         0.00        128          0
loop5             0.00         0.00         0.00         50          0
loop6             0.00         0.01         0.00        116          0
loop7             0.04         0.06         0.00        787          0
sda              20.77       323.06       283.95    4267580    3750953
loop8             0.01         0.08         0.00       1097          0
loop9             0.00         0.01         0.00        116          0
loop10            0.02         0.04         0.00        497          0
loop11            1.54         1.61         0.00      21315          0
loop12            0.01         0.08         0.00       1114          0
loop13            0.04         0.12         0.00       1536          0
loop14            0.00         0.00         0.00         46          0
loop15            0.00         0.00         0.00         49          0
loop16            0.00         0.00         0.00          4          0
```

Editing make-copies.sh from 5000 files to 100 files:

make copies shell script:

```
i=-1
while (( i++ < 100 )); do
  cp ./disk-files/foo.pdf "./disk-files/foo$i.pdf"
done
```

disk.c

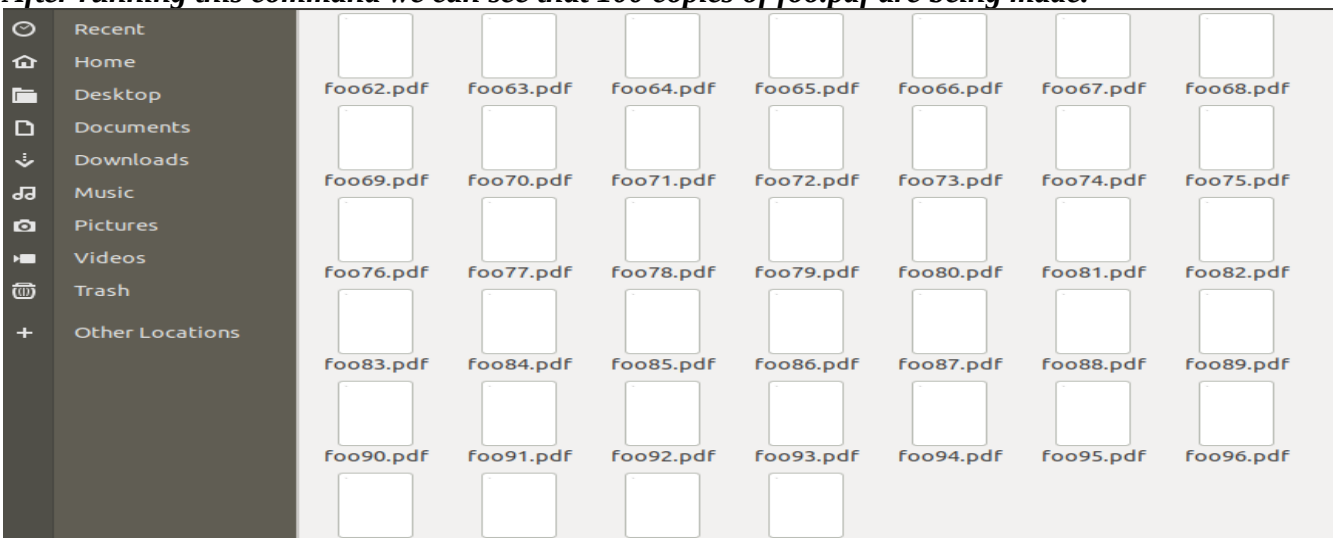
```
#include <unistd.h>
#include <stdio.h>
#include <sys/types.h>
#include <string.h>
#include <errno.h>
#include <stdlib.h>

#define FNAME_SIZE 100
#define MAX_FILE_NO 100
#define BLOCK_SIZE 1024

int main(int argc, char *argv[])
{

  int n, file_no;
  FILE *fp;
```

```
  char dest_file_name[FNAME_SIZE];
  char buf[BLOCK_SIZE];

  while(1)
   {
     file_no = rand() % MAX_FILE_NO;

     bzero(dest_file_name, FNAME_SIZE);
     sprintf(dest_file_name, "disk-files/foo%d.pdf", file_no);

     fp = fopen(dest_file_name, "rb");
     if (fp == NULL) {
          perror("Can't open dest file");
          exit(1);
     }

     bzero(buf,BLOCK_SIZE);
     while  ( (n = (int)fread( buf, 1, BLOCK_SIZE, fp )) > 0)
          {
            //do nothing with the read data;
            bzero(buf,BLOCK_SIZE);
          }

      fclose(fp);
    }
}
```

After executing the make-copies.sh command:

```
shubham@shubham-Vostro-15-3568:~/Desktop/lab_2_related_files$ ./make-copies.sh
shubham@shubham-Vostro-15-3568:~/Desktop/lab_2_related_files$
```

After running this command we can see that 100 copies of foo.pdf are being made.



Executing disk.c program:

```
shubham@shubham-Vostro-15-3568:~/Desktop/lab_2_related_files$ ./make-copies.sh
shubham@shubham-Vostro-15-3568:~/Desktop/lab_2_related_files$ gcc disk.c -o disk
shubham@shubham-Vostro-15-3568:~/Desktop/lab_2_related_files$ ./disk
```

Using ps aux command:

```
shubham  19769  0.0  1.4 2588100 55728 tty2    Sl+  13:46   0:01 /usr/lib/firefo
shubham  19834  0.0  1.1 2561264 44964 tty2    Sl+  13:46   0:00 /usr/lib/firefo
root     19894  0.0  0.0  25996  3196 ?        S    13:46   0:00 /sbin/dhclient
shubham  19927  106  0.0   4512   752 pts/0    R+   14:31   0:06 ./disk
shubham  19929  0.0  0.0  44476  3128 pts/1    R+   14:31   0:00 ps aux
shubham  20158  2.1  3.2 1360028 125824 tty2   Sl+  13:47   0:57 evince /home/sh
root     20531  0.2  0.0      0     0 ?        I    14:09   0:03 [kworker/u8:3-e
root     20641  0.0  0.0      0     0 ?        I    14:12   0:00 [kworker/2:1-ev
shubham  20663  0.3  0.7 802400 31144 ?        Ssl  14:13   0:03 /usr/lib/gnome-
shubham  20673  0.0  0.1  29552  4476 pts/0    Ss   14:13   0:00 bash
root     26283  0.1  0.0      0     0 ?        I    14:14   0:01 [kworker/u8:4-i
root     27556  0.0  0.0      0     0 ?        I    14:14   0:00 [kworker/2:3-rc
root     31856  0.0  0.0      0     0 ?        I<   12:23   0:00 [kworker/0:2H-e
shubham@shubham-Vostro-15-3568:~$
```

Execution of disk1.c:

```
shubham@shubham-Vostro-15-3568:~/Desktop/lab_2_related_files$ gcc disk1.c -o dis
k1
shubham@shubham-Vostro-15-3568:~/Desktop/lab_2_related_files$ ./disk1
```

Again using ps aux command:

```
shubham  19769  0.0  1.4 2588100 55724 tty2    Sl+  13:46   0:01 /usr/lib/firefo
shubham  19834  0.0  1.1 2561264 44960 tty2    Sl+  13:46   0:00 /usr/lib/firefo
root     19894  0.0  0.0  25996  3196 ?        S    13:46   0:00 /sbin/dhclient
shubham  19953 98.8  0.0   4512   772 pts/0    R+   14:32   0:04 ./disk1
shubham  19955  0.0  0.0  44476  3156 pts/1    R+   14:32   0:00 ps aux
shubham  20158  2.1  3.2 1360028 125804 tty2   Sl+  13:47   0:58 evince /home/sh
root     20531  0.2  0.0      0     0 ?        I    14:09   0:03 [kworker/u8:3-i
root     20641  0.0  0.0      0     0 ?        I    14:12   0:00 [kworker/2:1-mm
shubham  20663  0.3  0.7 802584 31136 ?        Dsl  14:13   0:03 /usr/lib/gnome-
shubham  20673  0.0  0.1  29552  4480 pts/0    Ss   14:13   0:00 bash
root     26283  0.1  0.0      0     0 ?        I    14:14   0:01 [kworker/u8:4-i
root     27556  0.0  0.0      0     0 ?        I    14:14   0:00 [kworker/2:3-rc
root     31856  0.0  0.0      0     0 ?        I<   12:23   0:00 [kworker/0:2H-e
shubham@shubham-Vostro-15-3568:~$
```

Using iostat command:

```
avg-cpu:  %user   %nice %system %iowait  %steal   %idle
          15.67    0.04   10.98    2.34    0.00   70.97

Device             tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
loop0             0.00         0.01         0.00        121          0
loop1             0.02         0.09         0.00       1254          0
loop2             0.00         0.02         0.00        330          0
loop3             0.02         0.04         0.00        507          0
loop4             0.00         0.01         0.00        128          0
loop5             0.00         0.00         0.00         50          0
loop6             0.00         0.01         0.00        116          0
loop7             0.04         0.06         0.00        787          0
sda              20.53       318.20       277.60    4450472    3882593
loop8             0.01         0.08         0.00       1097          0
loop9             0.00         0.01         0.00        116          0
loop10            0.01         0.04         0.00        497          0
loop11            1.45         1.52         0.00      21315          0
loop12            0.01         0.08         0.00       1114          0
loop13            0.04         0.11         0.00       1536          0
loop14            0.00         0.00         0.00         46          0
loop15            0.00         0.00         0.00         49          0
loop16            0.00         0.00         0.00          4          0

shubham@shubham-Vostro-15-3568:~$
```

Thus we can see that in both the cases of 5000 copies and 100 copies V.M remains same and physical memory is slightly changing,

Video link :

https://drive.google.com/folderview?id=1kpPBOvesONUwyKDZdPysqd6Lvl0TGMhd

Conclusion:

- ✔ Thus in this experiment we have studied the analysis of process current execution state and analysis of busy IO operations that consume high CPU time.

- ✔ The processes are found to beinn running state if they are continuously running without pausing and if they pause at the end then they enter the sleeping state.