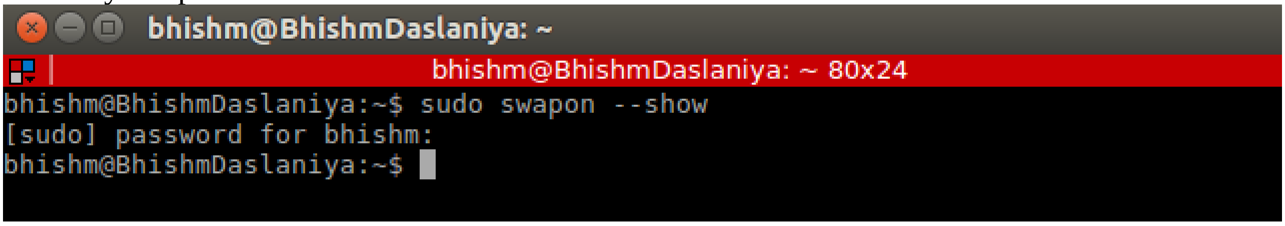
Practical - 8

Aim [A] : Create 1GB swap area in your linux partition and free it. Check the allocation of swap space. Execute following commands to monitor swap space in linux.

a. Swapon
b. use of /proc/swaps
c. free
d. top
e. atop
f. htop
g. glances
h. vmstat

Output :

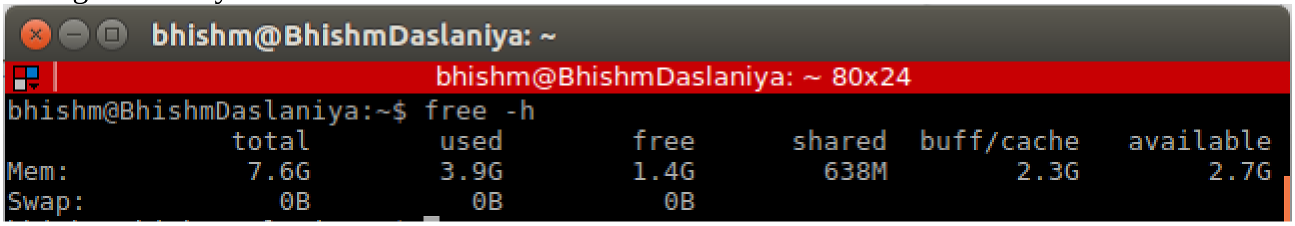
1. Check if any swap area available:



```
bhishm@BhishmDaslaniya:~$ sudo swapon --show
[sudo] password for bhishm:
bhishm@BhishmDaslaniya:~$
```

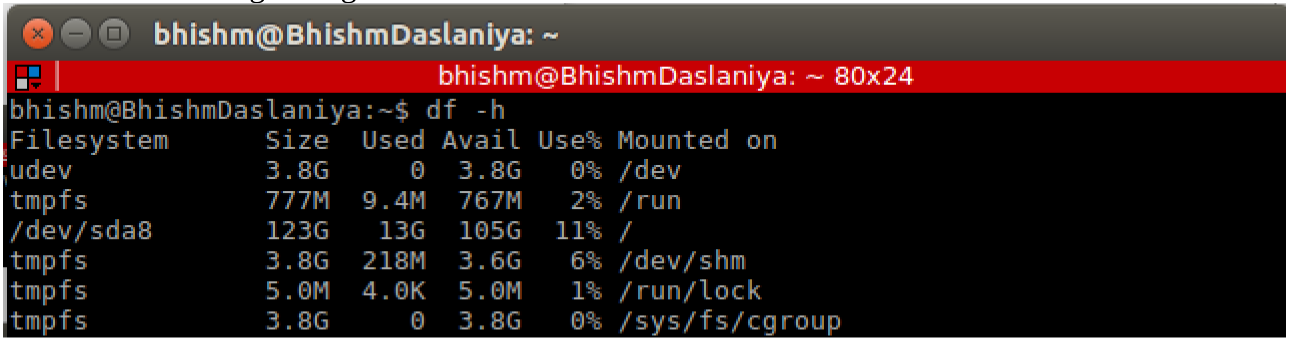
here system does not have any swap space available.

Check using free utility :



```
bhishm@BhishmDaslaniya:~$ free -h
              total        used        free      shared  buff/cache   available
Mem:           7.6G        3.9G        1.4G        638M        2.3G        2.7G
Swap:            0B          0B          0B
```

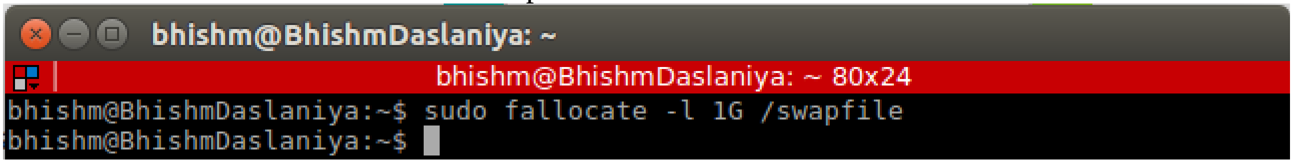
2. Check surrent disk uasge using df -h :



```
bhishm@BhishmDaslaniya:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            3.8G     0  3.8G   0% /dev
tmpfs           777M  9.4M  767M   2% /run
/dev/sda8       123G   13G  105G  11% /
tmpfs           3.8G  218M  3.6G   6% /dev/shm
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
tmpfs           3.8G     0  3.8G   0% /sys/fs/cgroup
```

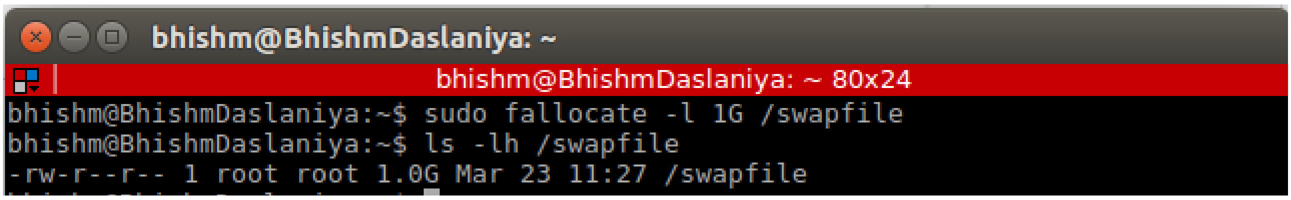
3. Creating Swap area of 1GB:

Commmand : sudo fallocate -l 1G /swapfile



```
bhishm@BhishmDaslaniya:~$ sudo fallocate -l 1G /swapfile
bhishm@BhishmDaslaniya:~$
```

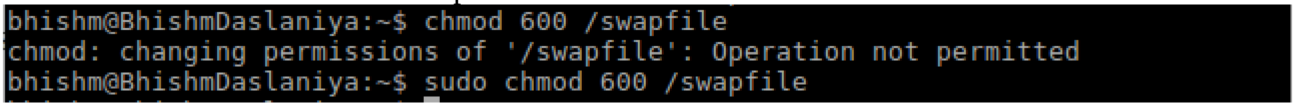
Verify space reserved :
Command : ls -lh /swapfile



```
bhishm@BhishmDaslaniya:~$ sudo fallocate -l 1G /swapfile
bhishm@BhishmDaslaniya:~$ ls -lh /swapfile
-rw-r--r-- 1 root root 1.0G Mar 23 11:27 /swapfile
```

4. Enabling the swap file:

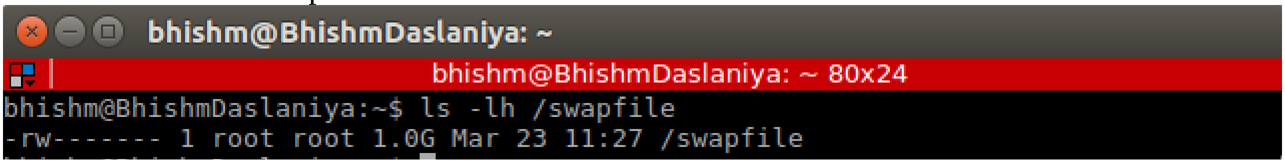
command : sudo chmod 600 /swapfile



```
bhishm@BhishmDaslaniya:~$ chmod 600 /swapfile
chmod: changing permissions of '/swapfile': Operation not permitted
bhishm@BhishmDaslaniya:~$ sudo chmod 600 /swapfile
```

Verify permission:

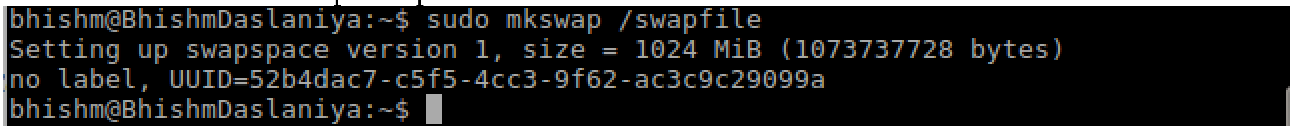
Command : ls -lh /swapfile



```
bhishm@BhishmDaslaniya:~$ ls -lh /swapfile
-rw------- 1 root root 1.0G Mar 23 11:27 /swapfile
```

Now mark the file as swap file:

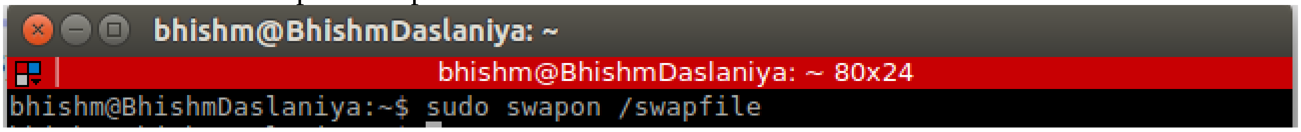
command : sudo mkswap /swapfile



```
bhishm@BhishmDaslaniya:~$ sudo mkswap /swapfile
Setting up swapspace version 1, size = 1024 MiB (1073737728 bytes)
no label, UUID=52b4dac7-c5f5-4cc3-9f62-ac3c9c29099a
bhishm@BhishmDaslaniya:~$
```

After marking the file, we can enable the swap file, allowing our system to start utilizing it :

command : sudo swapon /swapfile



```
bhishm@BhishmDaslaniya:~$ sudo swapon /swapfile
```

To make change permanent open the /etc/fstab file and append the following line:

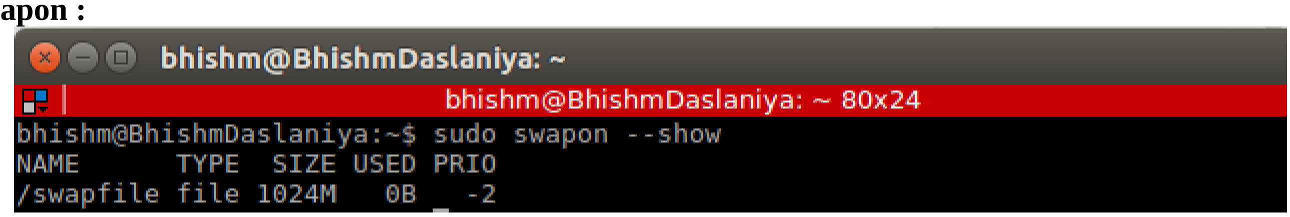
/swapfile swap swap defaults 0 0

[Make sure that you take backup of /etc/fstab if anything goes wrong by typing:

sudo cp /etc/fstab /etc/fstab.bak]

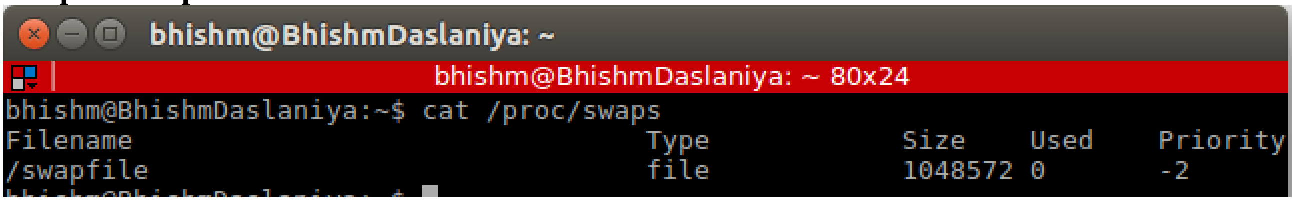
Now execute commands to monitor swap space :

a. swapon :



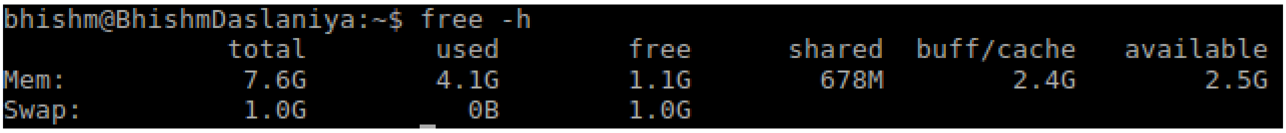
```
bhishm@BhishmDaslaniya:~$ sudo swapon --show
NAME      TYPE  SIZE USED PRIO
/swapfile file 1024M   0B   -2
```

b. use of /proc/swaps :



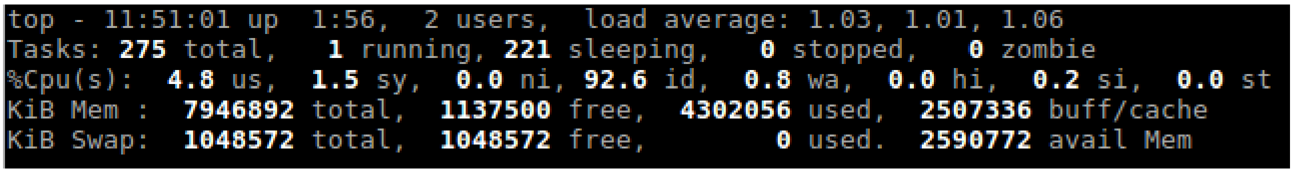
```
bhishm@BhishmDaslaniya:~$ cat /proc/swaps
Filename                                Type            Size    Used    Priority
/swapfile                               file            1048572 0       -2
```

c. free :



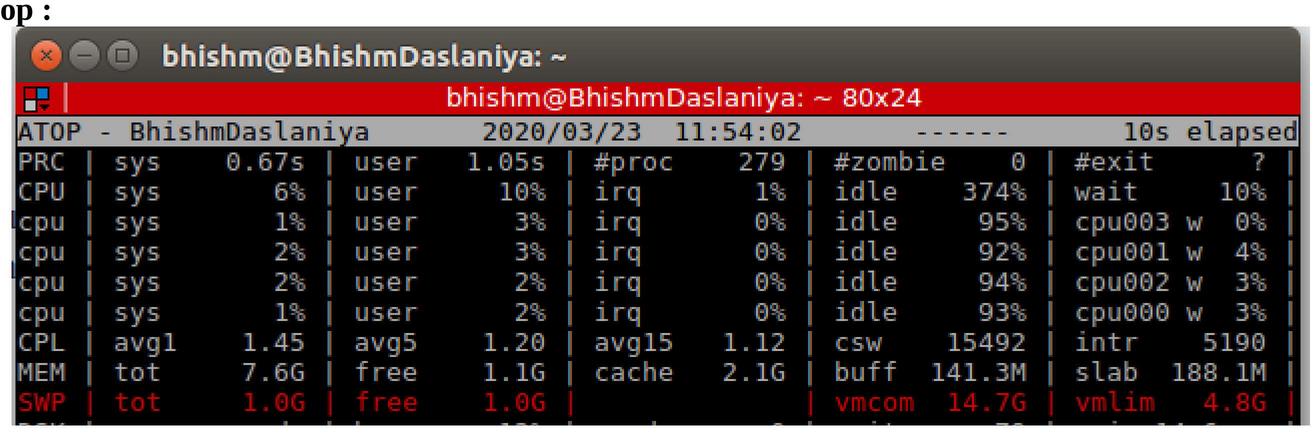
```
bhishm@BhishmDaslaniya:~$ free -h
              total        used        free      shared  buff/cache   available
Mem:           7.6G        4.1G        1.1G        678M        2.4G        2.5G
Swap:          1.0G          0B        1.0G
```

d. top :



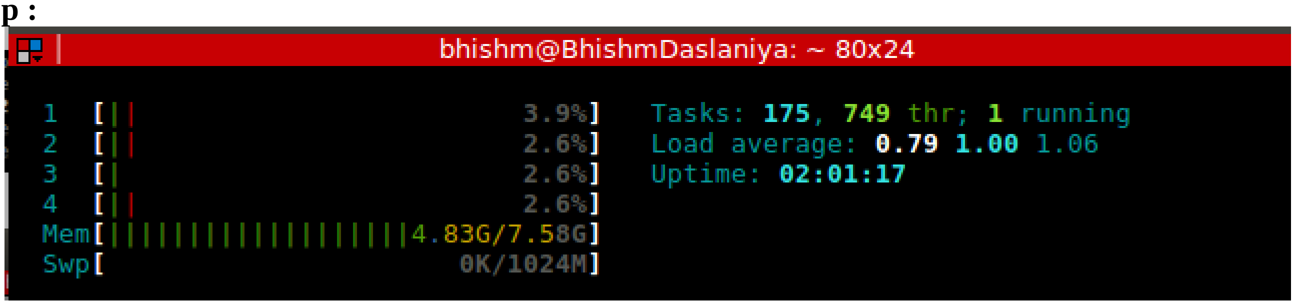
```
top - 11:51:01 up  1:56,  2 users,  load average: 1.03, 1.01, 1.06
Tasks: 275 total,   1 running, 221 sleeping,   0 stopped,   0 zombie
%Cpu(s):  4.8 us,  1.5 sy,  0.0 ni, 92.6 id,  0.8 wa,  0.0 hi,  0.2 si,  0.0 st
KiB Mem :  7946892 total,  1137500 free,  4302056 used,  2507336 buff/cache
KiB Swap:  1048572 total,  1048572 free,        0 used.  2590772 avail Mem
```

e. atop :



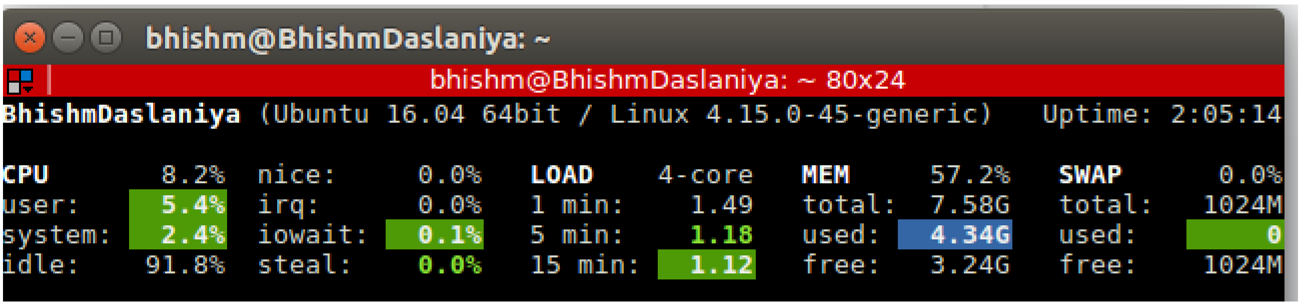
```
ATOP - BhishmDaslaniya         2020/03/23  11:54:02         ------        10s elapsed
PRC | sys    0.67s | user   1.05s | #proc    279 | #zombie    0 | #exit      ? |
CPU | sys       6% | user     10% | irq       1% | idle    374% | wait     10% |
cpu | sys       1% | user      3% | irq       0% | idle     95% | cpu003 w  0% |
cpu | sys       2% | user      3% | irq       0% | idle     92% | cpu001 w  4% |
cpu | sys       2% | user      2% | irq       0% | idle     94% | cpu002 w  3% |
cpu | sys       1% | user      2% | irq       0% | idle     93% | cpu000 w  3% |
CPL | avg1    1.45 | avg5    1.20 | avg15   1.12 | csw    15492 | intr    5190 |
MEM | tot     7.6G | free    1.1G | cache   2.1G | buff  141.3M | slab  188.1M |
SWP | tot     1.0G | free    1.0G |              | vmcom  14.7G | vmlim   4.8G |
```

f. htop :



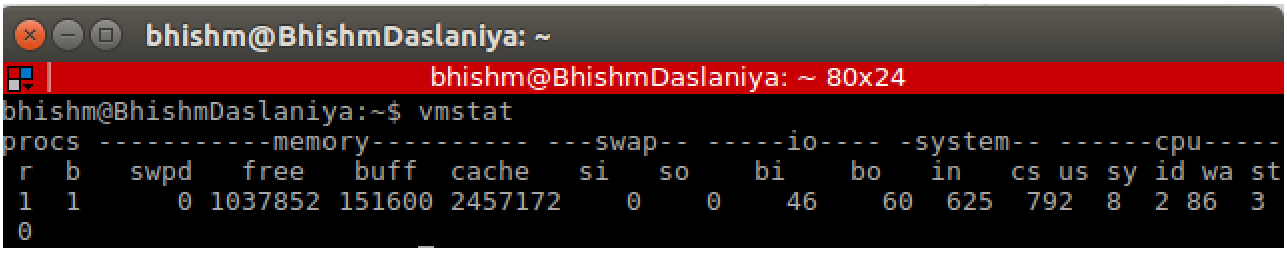
```
  1  [||                        3.9%]   Tasks: 175, 749 thr; 1 running
  2  [||                        2.6%]   Load average: 0.79 1.00 1.06
  3  [|                         2.6%]   Uptime: 02:01:17
  4  [||                        2.6%]
  Mem[||||||||||||||||||4.83G/7.58G]
  Swp[                     0K/1024M]
```

g. glances :



```
BhishmDaslaniya (Ubuntu 16.04 64bit / Linux 4.15.0-45-generic)   Uptime: 2:05:14

CPU       8.2%  nice:     0.0%  LOAD    4-core  MEM     57.2%  SWAP     0.0%
user:     5.4%  irq:      0.0%  1 min:    1.49  total:  7.58G  total:  1024M
system:   2.4%  iowait:   0.1%  5 min:    1.18  used:   4.34G  used:       0
idle:    91.8%  steal:    0.0%  15 min:   1.12  free:   3.24G  free:   1024M
```

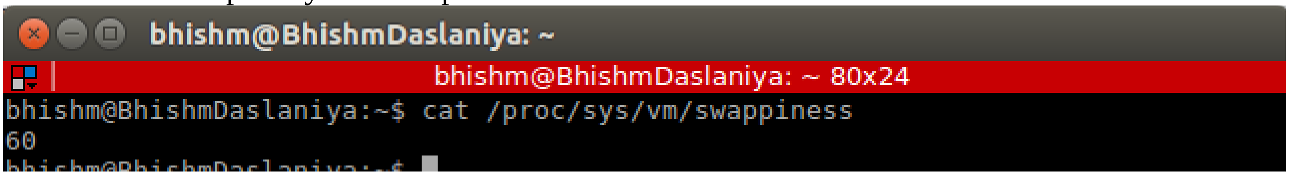
h. vmstat :



```
bhishm@BhishmDaslaniya:~$ vmstat
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 1  1      0 1037852 151600 2457172    0    0    46    60  625  792  8  2 86  3
 0
```

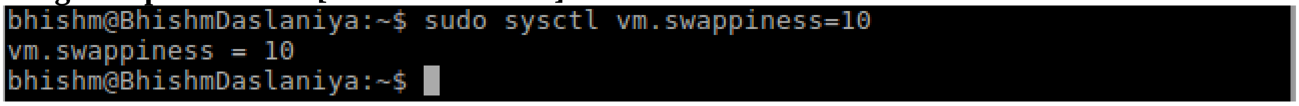
For checking current swapiness value:

command : cat /proc/sys/vm/swapiness



```
bhishm@BhishmDaslaniya:~$ cat /proc/sys/vm/swappiness
60
bhishm@BhishmDaslaniya:~$
```

For change swapiness value [between 0 to 100] :

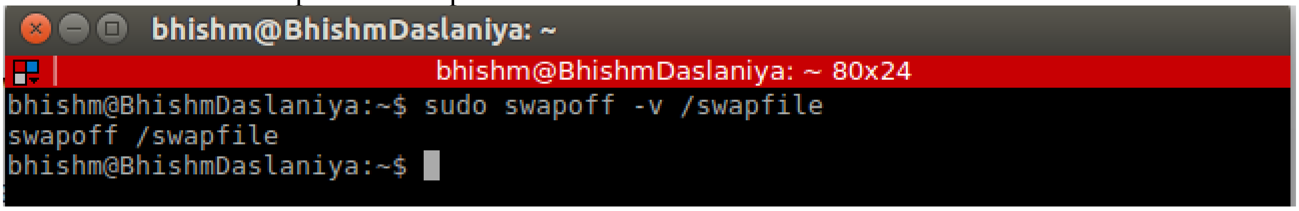


```
bhishm@BhishmDaslaniya:~$ sudo sysctl vm.swappiness=10
vm.swappiness = 10
bhishm@BhishmDaslaniya:~$
```

Remove the swap space :

First deactive the swap:

command : sudo swapoff -v /swapfile

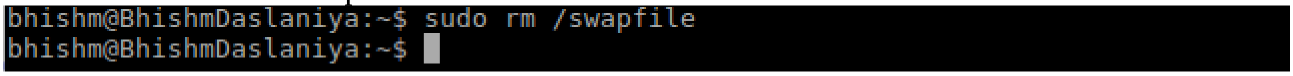


```
bhishm@BhishmDaslaniya:~$ sudo swapoff -v /swapfile
swapoff /swapfile
bhishm@BhishmDaslaniya:~$
```

If you have added swap file as permanent in /etc/fstab then remve the swap file entry **/swapfile swap swap defaults 0 0** from /stc/fstab.

Finally remove the swap file:

command : sudo rm /swapfile



```
bhishm@BhishmDaslaniya:~$ sudo rm /swapfile
bhishm@BhishmDaslaniya:~$
```

Aim [B] : Write the simulation Paging Algorithms program for demand paging and show the page scheduling and total number of page faults according to FIFO, LRU, and optimal page replacement algorithm. Assume the memory of ‘n’ frames.

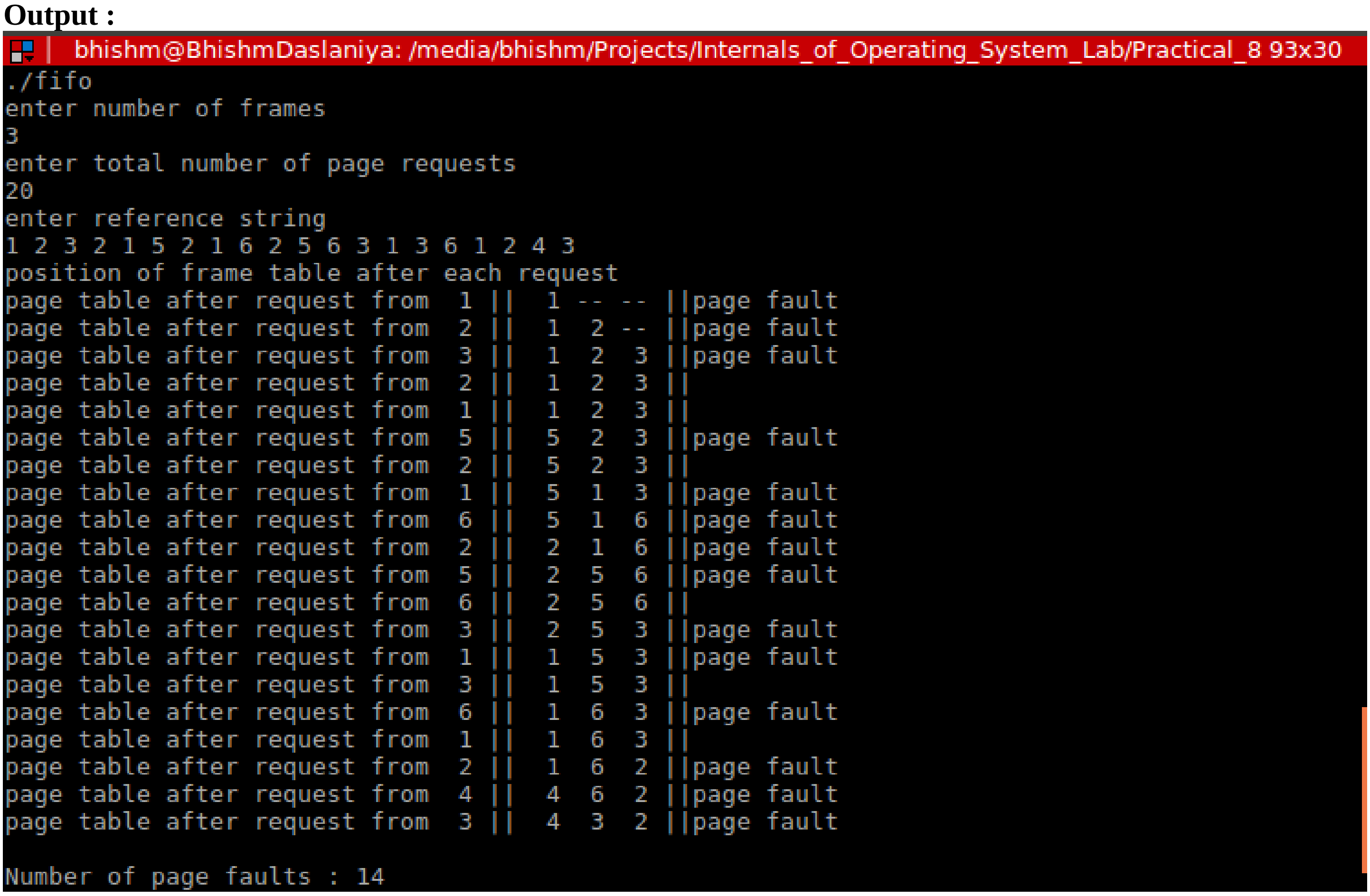
FIFO Page Replacement algorithm:

Code :

```
#include<bits/stdc++.h>
using namespace std;

int present(int table_frame[], int nf, int page){
	for(int i=0; i<nf; i++)
		if(page == table_frame[i])
			return 1;
	return 0;
}

void printtable(int table_frame[], int nf){
	for(int i=0; i<nf; i++){
		if(table_frame[i] == -1){
			printf("-- ");
		}else{
```

```
                                    printf("%2d ", table_frame[i]);
                        }
            }
            printf("||");
}

int main(){

    //nf-number of frames
    int n,nf,i,pos=0;
    printf("enter number of frames\n");
    scanf("%d",&nf);
    int table_frame[nf];

    for(i=0;i<nf;i++) {
        table_frame[i]=-1;
    }
    printf("enter total number of page requests\n");
    scanf("%d",&n);
    int pages[n];
    printf("enter reference string\n");

    for(i=0;i<n;i++){
        scanf("%d",&pages[i]);
    }

    int count1=0;
    printf("position of frame table after each request\n");

    for(i=0;i<n;i++){
        printf("page table after request from %2d || ",pages[i]);
        if(!present(table_frame,nf,pages[i])){
           table_frame[pos] = pages[i];
           pos = (pos+1)%nf ;//considering it as a queue
           printtable(table_frame,nf);
           printf("page fault\n");
           count1++;
           continue;
        }
        printtable(table_frame,nf);
                        printf("\n");
    }
    printf("\nNumber of page faults : %d\n\n", count1);
}
```

Output :


bhishm@BhishmDaslaniya: /media/bhishm/Projects/Internals_of_Operating_System_Lab/Practical_8 93x30

```
./fifo
enter number of frames
3
enter total number of page requests
20
enter reference string
1 2 3 2 1 5 2 1 6 2 5 6 3 1 3 6 1 2 4 3
position of frame table after each request
page table after request from  1 ||  1 -- -- ||page fault
page table after request from  2 ||  1  2 -- ||page fault
page table after request from  3 ||  1  2  3 ||page fault
page table after request from  2 ||  1  2  3 ||
page table after request from  1 ||  1  2  3 ||
page table after request from  5 ||  5  2  3 ||page fault
page table after request from  2 ||  5  2  3 ||
page table after request from  1 ||  5  1  3 ||page fault
page table after request from  6 ||  5  1  6 ||page fault
page table after request from  2 ||  2  1  6 ||page fault
page table after request from  5 ||  2  5  6 ||page fault
page table after request from  6 ||  2  5  6 ||
page table after request from  3 ||  2  5  3 ||page fault
page table after request from  1 ||  1  5  3 ||page fault
page table after request from  3 ||  1  5  3 ||
page table after request from  6 ||  1  6  3 ||page fault
page table after request from  1 ||  1  6  3 ||
page table after request from  2 ||  1  6  2 ||page fault
page table after request from  4 ||  4  6  2 ||page fault
page table after request from  3 ||  4  3  2 ||page fault

Number of page faults : 14
```

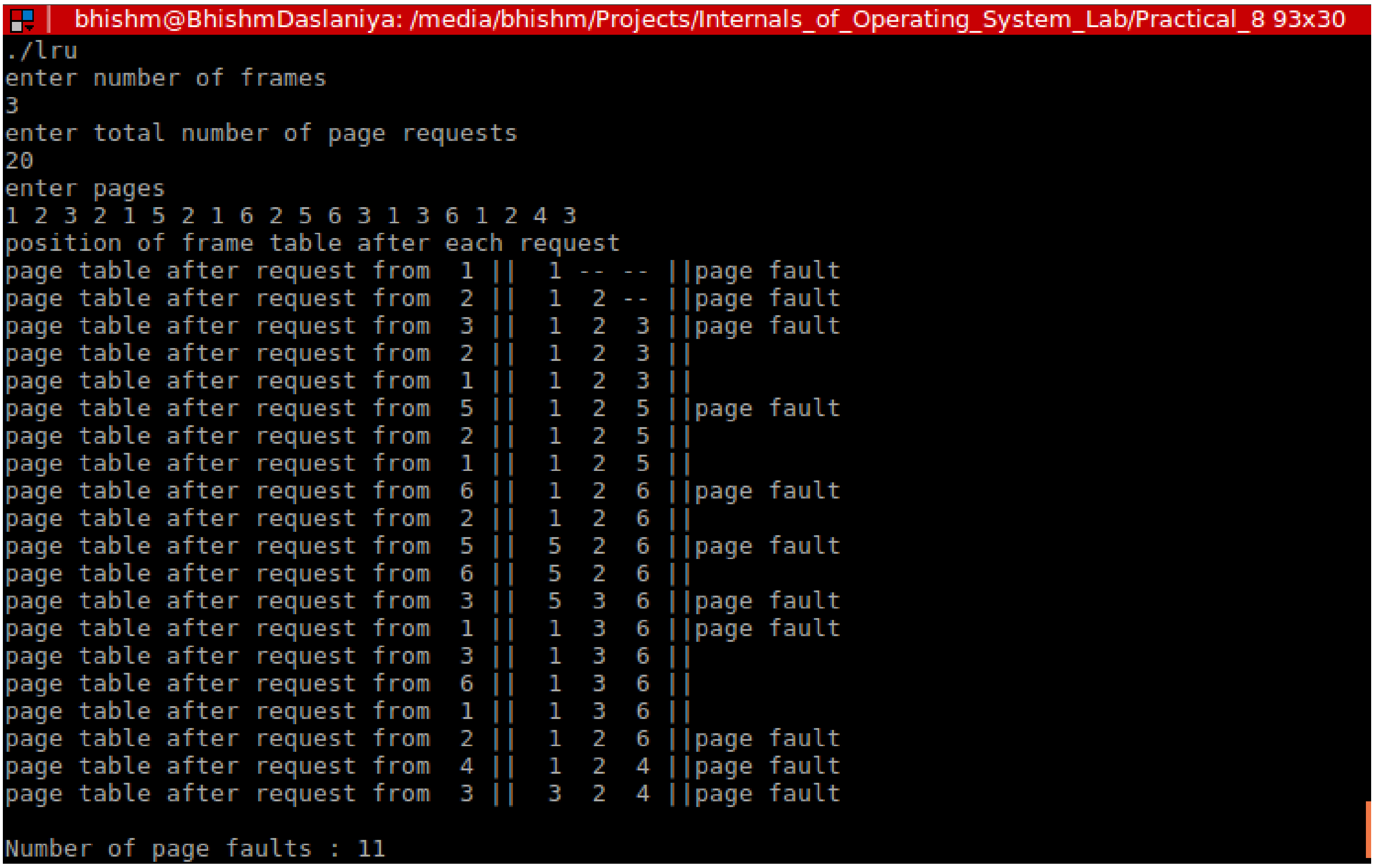
LRU Page Replacement algorithm:
Code :

```
#include<bits/stdc++.h>

int present(int table_frame[], int nf, int page)
{
        for(int i=0; i<nf; i++)
                if(page == table_frame[i])
                        return 1;
        return 0;
}

void printtable(int table_frame[], int nf)
{
        for(int i=0; i<nf; i++)
        {
                if(table_frame[i] == -1)
                        printf("-- ");
                else
                        printf("%2d ", table_frame[i]);
        }
        printf("||");
}


int findpos(int table_frame[], int nf, int pages[], int curr, int np)
```

```
{
        for(int i=0; i<nf; i++)
                if(table_frame[i] == -1)
                        return i;

        int pos[nf] = {0};
        for(int i=0; i<nf; i++)
        {
                pos[i] = -1e9;
                for(int j=curr-1; j>=0; j--)
                        if(pages[j] == table_frame[i])
                        {
                                pos[i] = j;
                                break;
                        }
        }

        int min1 = 1000000, retPos = -1;
        for(int i=0; i<nf; i++)
                if(min1 > pos[i])
                {
                        min1 = pos[i];
                        retPos = i;
                }

        return retPos;
}

int main()
{
        // #ifndef ONLINE_JUDGE
 //      // for getting input from input.txt
 //      freopen("input.txt", "r", stdin);
 //      // for writing output to output.txt
 //      freopen("lru.txt", "w", stdout);
 //   #endif

   //nf-number of frames
   int n,nf,i,pos=0;

   printf("enter number of frames\n");
   scanf("%d",&nf);
   int table_frame[nf];
   for(i=0;i<nf;i++)
   {
      table_frame[i]=-1;
   }
   printf("enter total number of page requests\n");
   scanf("%d",&n);
   int pages[n];
   printf("enter pages\n");
   for(i=0;i<n;i++)
```

```
  {
    scanf("%d",&pages[i]);
  }
  int count1=0;
  printf("position of frame table after each request\n");
  for(i=0;i<n;i++)
  {
    printf("page table after request from %2d || ",pages[i]);
    if(!present(table_frame,nf,pages[i]))
    {
      int pos = findpos(table_frame,nf,pages,i,n);
      table_frame[pos]=pages[i];

      printtable(table_frame,nf);
      printf("page fault\n");
      count1++;
      continue;
    }
    printtable(table_frame,nf);
            printf("\n");
  }
  printf("\nNumber of page faults : %d\n\n", count1);
}
```

Output :


bhishm@BhishmDaslaniya: /media/bhishm/Projects/Internals_of_Operating_System_Lab/Practical_8 93x30

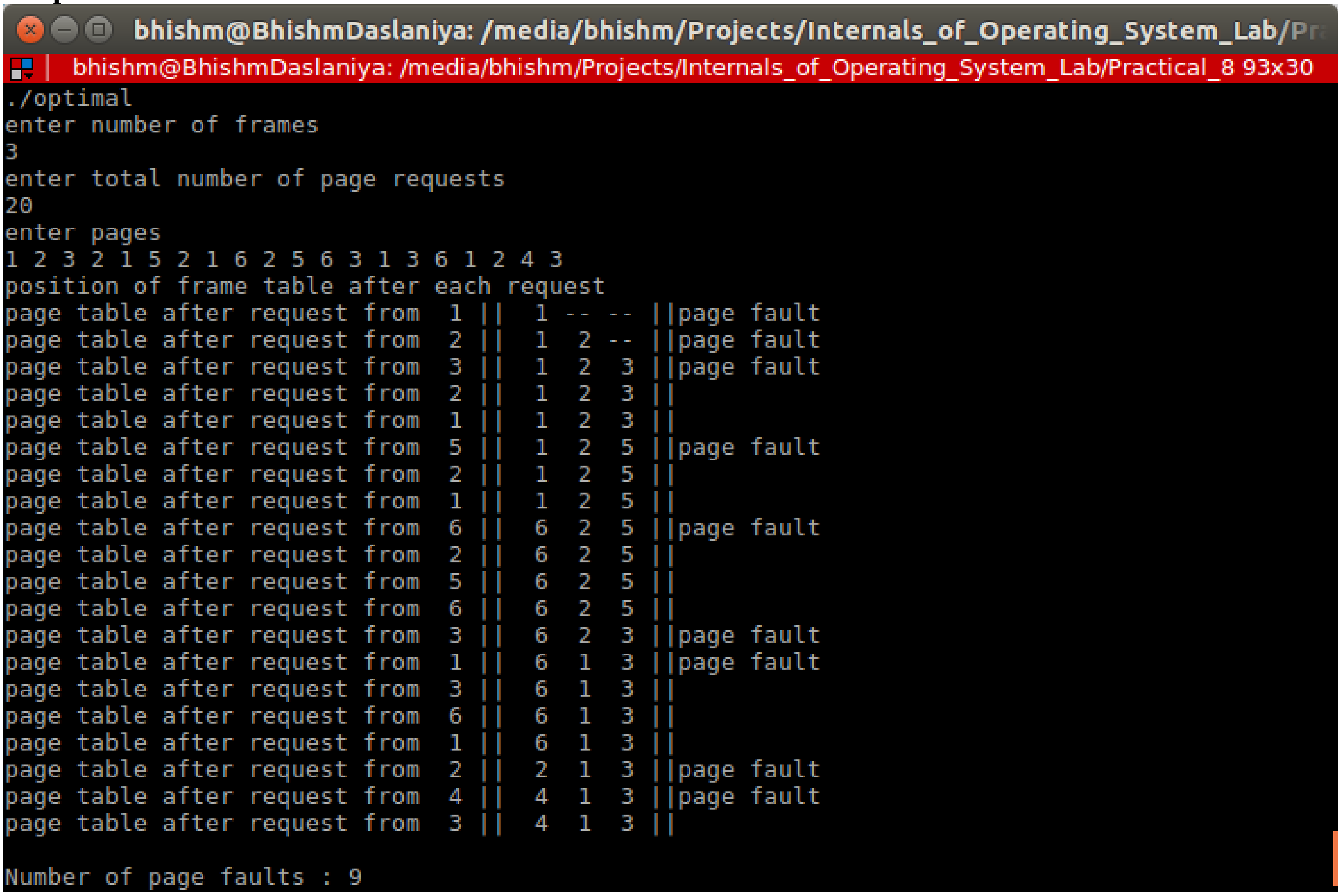
```
./lru
enter number of frames
3
enter total number of page requests
20
enter pages
1 2 3 2 1 5 2 1 6 2 5 6 3 1 3 6 1 2 4 3
position of frame table after each request
page table after request from  1 ||  1 -- -- ||page fault
page table after request from  2 ||  1  2 -- ||page fault
page table after request from  3 ||  1  2  3 ||page fault
page table after request from  2 ||  1  2  3 ||
page table after request from  1 ||  1  2  3 ||
page table after request from  5 ||  1  2  5 ||page fault
page table after request from  2 ||  1  2  5 ||
page table after request from  1 ||  1  2  5 ||
page table after request from  6 ||  1  2  6 ||page fault
page table after request from  2 ||  1  2  6 ||
page table after request from  5 ||  5  2  6 ||page fault
page table after request from  6 ||  5  2  6 ||
page table after request from  3 ||  5  3  6 ||page fault
page table after request from  1 ||  1  3  6 ||page fault
page table after request from  3 ||  1  3  6 ||
page table after request from  6 ||  1  3  6 ||
page table after request from  1 ||  1  3  6 ||
page table after request from  2 ||  1  2  6 ||page fault
page table after request from  4 ||  1  2  4 ||page fault
page table after request from  3 ||  3  2  4 ||page fault

Number of page faults : 11
```

Optimal Page Replacement algorithm :

Code :

```
#include<bits/stdc++.h>

int present(int table_frame[], int nf, int page)
{
	for(int i=0; i<nf; i++)
		if(page == table_frame[i])
			return 1;
	return 0;
}

void printtable(int table_frame[], int nf)
{
	for(int i=0; i<nf; i++)
	{
		if(table_frame[i] == -1)
			printf("-- ");
		else
			printf("%2d ", table_frame[i]);
	}
	printf("||");
}


int findpos(int table_frame[],int nf,int pages[],int curr,int np)
{
    int i,j;
    for(i=0;i<nf;i++)
    {
        if(table_frame[i] == -1)
            return i;
    }

    int pos[nf]={0};
    for(i=0;i<nf;i++)
    {
        pos[i]=1e9;
        for(j=curr+1;j<np;j++)
        {
            if(pages[j]==table_frame[i])
            {
                pos[i]=j;
                break;
            }
        }
    }

    int max1=-1;
    int returnpos=-1;
    for(i=0;i<nf;i++)
    {
        if(pos[i]>max1)
        {
            max1=pos[i];
            returnpos=i;
        }
    }

    return returnpos;
}

int main()
{
    //nf-number of frames
    int n,nf,i,pos=0;

    printf("enter number of frames\n");
    scanf("%d",&nf);
    int table_frame[nf];
    for(i=0;i<nf;i++)
    {
        table_frame[i]=-1;
    }

    printf("enter total number of page requests\n");
    scanf("%d",&n);
    int pages[n];
    printf("enter pages\n");
    for(i=0;i<n;i++)
    {
        scanf("%d",&pages[i]);
    }

    int count1=0;
    printf("position of frame table after each request\n");
    for(i=0;i<n;i++)
    {
        printf("page table after request from %2d || ",pages[i]);
        if(!present(table_frame,nf,pages[i]))
        {
             int pos = findpos(table_frame,nf,pages,i,n);
             table_frame[pos]=pages[i];

             printtable(table_frame,nf);
             printf("page fault\n");
             count1++;
             continue;
        }
        printtable(table_frame,nf);
                    printf("\n");
    }
    printf("\nNumber of page faults : %d\n\n", count1);
}
```

Output:



bhishm@BhishmDaslaniya: /media/bhishm/Projects/Internals_of_Operating_System_Lab/Pr
bhishm@BhishmDaslaniya: /media/bhishm/Projects/Internals_of_Operating_System_Lab/Practical_8 93x30

```
./optimal
enter number of frames
3
enter total number of page requests
20
enter pages
1 2 3 2 1 5 2 1 6 2 5 6 3 1 3 6 1 2 4 3
position of frame table after each request
page table after request from  1 ||  1 -- -- ||page fault
page table after request from  2 ||  1  2 -- ||page fault
page table after request from  3 ||  1  2  3 ||page fault
page table after request from  2 ||  1  2  3 ||
page table after request from  1 ||  1  2  3 ||
page table after request from  5 ||  1  2  5 ||page fault
page table after request from  2 ||  1  2  5 ||
page table after request from  1 ||  1  2  5 ||
page table after request from  6 ||  6  2  5 ||page fault
page table after request from  2 ||  6  2  5 ||
page table after request from  5 ||  6  2  5 ||
page table after request from  6 ||  6  2  5 ||
page table after request from  3 ||  6  2  3 ||page fault
page table after request from  1 ||  6  1  3 ||page fault
page table after request from  3 ||  6  1  3 ||
page table after request from  6 ||  6  1  3 ||
page table after request from  1 ||  6  1  3 ||
page table after request from  2 ||  2  1  3 ||page fault
page table after request from  4 ||  4  1  3 ||page fault
page table after request from  3 ||  4  1  3 ||

Number of page faults : 9
```

Conclusion: From this practical I have learnt about swap area and how to create it and some useful commands to monitor swap area and also learnt about how to implement page replacement algorithms in C/C++.