EXP 1: BASIC LINUX COMMANDS

```
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~$ mkdir f
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~$ mkdir f2
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~$ cd f
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/f$ vi s.txt
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/f$ cat s.txt
Hello World
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/f$ ls
s2.txt  s3.txt  s.txt
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/f$ ls -l
total 8
-rw-rw-r-- 1 albatch2-17 albatch2-17  0 Mar  2 15:05 s2.txt
-rw-rw-r-- 1 albatch2-17 albatch2-17  3 Mar  2 15:06 s3.txt
-rw-rw-r-- 1 albatch2-17 albatch2-17 12 Mar  2 15:04 s.txt
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/f$ ls -a
.  ..  s2.txt  s3.txt  s.txt
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/f$ cat s2.txt
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/f$ cp s.txt s2.txt
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/f$ cat s2.txt
Hello World
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/f$ mv s2.txt s3.txt
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/f$ ls
s3.txt  s.txt
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/f$ wc s3.txt
 1  2 12 s3.txt
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/f$ cat -b s3.txt
     1  Hello World
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/f$ rm s3.txt
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/f$ ls
s.txt
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/f$ cd
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~$ rmdir f2
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~$ date
Thursday 02 March 2023 03:09:30 PM IST
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~$ cal
     March 2023
Su Mo Tu We Th Fr Sa
                1  2  3  4
 5  6  7  8  9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29 30 31
```

EXP 2: SHELL PROGRAMMING

```
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~$ whoami
albatch2-17
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~$ users
albatch2-17
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~$ passwd
Changing password for albatch2-17.
Current password:
New password:
Retype new password:
passwd: password updated successfully
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~$
```

PROGRAM

```
#!/bin/sh
echo "Enter a number :"
read num
b=`expr $num % 2`
if [ $b -eq 0 ]
then
echo "Even"
else
echo "Odd"
fi
```

OUTPUT

```
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/csal50$ chmod +x evenodd.sh
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/csal50$ ./evenodd.sh
Enter a number :
5
Odd
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/csal50$ ./evenodd.sh
Enter a number :
6
Even
```

PROGRAM

```
#!/bin/sh
echo "Enter the Number:"
read num
sum=0
while [ $num -ne 0 ]
do
        digit=`expr $num % 10`
        num=`expr $num / 10`
        sum=`expr $sum + $digit`
done
echo "Sum is $sum"
```

OUTPUT

```
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/csal50$ chmod +x digitsum.sh
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/csal50$ ./digitsum.sh
Enter the Number:
315
Sum is 9
```

PROGRAM

```
#!/bin/sh
echo "Enter a number:"
read n
p=$n
q=0
rev=0
while [ $p -gt 0 ]
do
        q=`expr $p % 10`
        p=`expr $p / 10`
rev=`expr $rev \* 10 + $q`
done
if [ $rev -eq $n ]
then
        echo "$n is a Palindrome"
else
        echo "$n is a not Palindrome"
fi
```

OUTPUT

```
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/csal50$ ./palindrome.sh
enter number
1331
1331 is a palindrome
albatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/csal50$ ./palindrome.sh
enter number
456
456 is a not palindrome
```

PROGRAM

```
#!/bin/sh
echo "Enter a Number:"
read n
f=1
i=1
until [ $i -ge `expr $n + 1` ]
do
        f=`expr $f \* $i`
        i=`expr $i + 1`
done
echo "Factorial of $n is $f"
```

OUTPUT

```
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$ chmod +x factorial.sh
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$ ./factorial.sh
enter number
6
factorial of 6 is 720
```

PROGRAM

```
#!/bin/sh
while [ 1 -eq 1 ]
do
        echo "Enter two numbers:"
        read a b
        echo "Select Operation \n1.Addition"
        echo "2.Subtraction\n3.Multiplication"
        echo "4.Division\n5.Modulus\n6.Exit"
        read c
        case $c in
        1)echo "$a+$b = `expr $a + $b`";;
        2)echo "$a-$b = `expr $a - $b`";;
        3)echo "$a*$b = `expr $a \* $b`";;
        4)echo "$a/$b = `expr $a / $b`";;
        5)echo "$a%$b = `expr $a % $b`";;
        6)exit;;
        *)echo "Invalid Input"
        esac
done
```

OUTPUT

```
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$ chmod +x calculator.sh
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$ ./calculator.sh
Enter two numbers:
5 6
Select Operation
1.Addition
2.Subtraction
3.Multiplication
4.Division
5.Modulus
6.Exit
2
5-6 = -1
Enter two numbers:
2 3
Select Operation
1.Addition
2.Subtraction
3.Multiplication
4.Division
5.Modulus
6.Exit
2
2-3 = -1
Enter two numbers:
4 6
Select Operation
1.Addition
2.Subtraction
3.Multiplication
4.Division
5.Modulus
6.Exit
3
4*6 = 24
Enter two numbers:
10 5
Select Operation
1.Addition
2.Subtraction
3.Multiplication
4.Division
5.Modulus
6.Exit
4
10/5 = 2
Enter two numbers:
7 3
Select Operation
1.Addition
2.Subtraction
3.Multiplication
4.Division
5.Modulus
6.Exit
5
7%3 = 1
Enter two numbers:
1 2
Select Operation
1.Addition
2.Subtraction
3.Multiplication
4.Division
5.Modulus
6.Exit
6
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$
```

PROGRAM

```
#!/bin/sh
while [ 1 -eq 1 ]
do
        echo "Choose Option
        echo "1.Current Working Directory"
        echo "2.Today's Date"
        echo "3.List of Users\n4.exit"
        read n
        case $n in
        1)pwd;;
        2)date;;
        3)who;;
        4)exit;;
        *)echo "Invalid Input"
        esac
done
```

OUTPUT

```
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$ chmod +x classified.sh
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$ ./classified.sh
Choose Option
1.Current Working Directory
2.Today's Date
3.List of Users
4.exit
1
/home/aibatch2-17/csai50
Choose Option
1.Current Working Directory
2.Today's Date
3.List of Users
4.exit
2
Wednesday 08 March 2023 01:00:26 PM IST
Choose Option
1.Current Working Directory
2.Today's Date
3.List of Users
4.exit
3
aibatch2-17 :0           2023-03-08 12:42 (:0)
Choose Option
1.Current Working Directory
2.Today's Date
3.List of Users
4.exit
4
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$
```

EXP 3: SYSTEM CALLS OF LINUX OS

ROGRAM

```
#include<sys/types.h>
#include<stdio.h>
#include<unistd.h>
#include<sys/wait.h>
void main() {oo
        int status,pid,child_pid;
        pid=fork();
        if(pid==-1) {
                printf("Child Process Creation
Failed!");
                return;
        }
        else if(pid==0) {
                printf("Inside Child Process with
process ID : %d \n",getpid());
                char *arg[]={"Hello",NULL};
                execvp("./child",arg);
        }
        else {
                child_pid=wait(&status);
                printf("\nInside the parent process
with ID: %d \n",getpid());
                printf("Child Process Created
Successfully\n");
        }
}
```

OUTPUT

```
:~/csal50$ gcc q3.1.c
:~/csal50$ ./a.out
Inside Child Process with process ID : 5759
Hello, I'm the Child Process!
Inside the parent process with ID: 5758
Child Process Created Successfully
:~/csal50$
```

child.c

```
#include<stdio.h>
void main() {
    printf("Hello, I'm the Child Process!");
}
```

C Tab Width: 4 Ln 5, Col 1 INS

PROGRAM

```
#include<stdio.h>
#include<stdlib.h>
#include<sys/stat.h>
#include<time.h>
int main() {
char file[10];
struct stat *node;
node= (struct stat*) malloc (sizeof(struct stat));
printf("\nEnter  Filename: ");
scanf("%s",file);
stat(file,node);
if(node->st_ino==0)
        printf("\nSuch a file does not exist!!");
else {
        printf("\nInode/Serial Number:
%ld",node->st_ino);
        printf("\nBlock Size: %ld",node->st_blksize);
        printf("\nAccess Time: %ld",node->st_atime);
        printf("\nLast modified time:
%ld",node->st_mtime);
        printf("\nGroup ID: %d",node->st_gid);
        printf("\nSize of File: %ld",node->st_size);
        printf("\nPermissions: %d",node->st_mode);
        printf("\nUser ID: %d\n",node->st_uid);
}
}
```

OUTPUT

```
aibatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/csal50$ gcc q3.2.c
aibatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/csal50$ ./a.out

Enter  Filename: dummydir

Inode/Serial Number: 9190561
Block Size: 4096
Access Time: 1679558786
Last modified time: 1679558784
Group ID: 1005
Size of File: 4096
Permissions: 16893
User ID: 1005
aibatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/csal50$
```

PROGRAM

```
#include<sys/types.h>
#include<stdio.h>
#include<dirent.h>
void main() {
        DIR *dir;
        struct dirent *ptr2;
        char dir_name[50];
        printf("\nEnter the Directory: ");
        scanf("%s",dir_name);
        dir=opendir(dir_name);
        while((ptr2=readdir(dir))!=NULL)

printf("%ld\t%s\n",ptr2->d_ino,ptr2->d_name);
        closedir(dir);
}
```

OUTPUT

```
aibatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/csal50$ gcc q3.c
aibatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/csal50$ ./a.out

Enter the Directory: dummydir
9193094 sample1.txt
9193807 sample3.txt
9187978 ..
9190561 .
9193793 sample2.txt
aibatch2-17@nits-HP-280-Pro-G6-Microtower-PC:~/csal50$
```

EXP 4: I/O SYSTEM CALLS

PROGRAM

```
#include<sys/stat.h>
#include<sys/types.h>
#include<stdio.h>
#include<unistd.h>
#include<string.h>
#include<fcntl.h>
```

```
int main() {
        int fd,fd2;
        char wbuf[128];
        char rbuf[128];
        fd=open("file.txt",O_RDWR);
        printf("Enter the text to be written:\n");
        scanf("%s",wbuf);
        write(fd,wbuf,strlen(wbuf));
        close(fd);
        fd2=open("file.txt",O_RDWR);
        printf("\nThe Contents are:\n");
        read(fd2,rbuf,100);
        printf("%s\n",rbuf);
        close(fd2);
        return 0;
}
```

OUTPUT

```
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$ gcc q4.c
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$ ./a.out
Enter the text to be written:
Welcome_To_OS_Lab

The Contents are:
Welcome_To_OS_Lab
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$
```

EXP 5: NON-PREEMPTIVE SCHEDULING ALGORITHMS

PROGRAM

```
//SJF Scheduling
#include<stdio.h>
void main() {
        int p[20],wt[20],bt[20],tt[20],n,temp;
        float wt_avg=0,tt_avg=0;
        printf("Enter the number of Processes : ");
        scanf("%d",&n);
        printf("\nEnter the Burst Time of Each Process
:\n");
        for(int i=0;i<n;i++) {
                p[i]=i+1;
                printf("P%d : ",i+1);
                scanf("%d",&bt[i]);
        }
        for(int i=1;i<n;i++)
                for(int j=0;j<n-i;j++)
                        if(bt[j]>bt[j+1]) {
                                temp=p[j];
                                p[j]=p[j+1];
                                p[j+1]=temp;
                                temp=bt[j];
                                bt[j]=bt[j+1];
                                bt[j+1]=temp;

                        }
wt[0]=0;
for(int i=1;i<n;i++) {
        wt[i]=bt[i-1]+wt[i-1];
        wt_avg+=wt[i];
}
wt_avg/=n;
        for(int i=0;i<n;i++) {
                tt[i]=wt[i]+bt[i];
                tt_avg+=tt[i];
        }
        tt_avg/=n;
        printf("\nProcess\t\t Burst Time\t\t Waiting
Time\t\t Turnaround Time\n");
        for(int i=0;i<n;i++)

printf("P%d\t\t\t%d\t\t\t\t%d\t\t\t\t%d\n",p[i],bt[i],wt[i]
,tt[i]);
        printf("\nAverage Waiting Time :
%.2f",wt_avg);
        printf("\nAverage Turnaround Time :
%.2f\n",tt_avg);
        printf("\nGantt Chart\n\n");

printf("----------------------------------------------------------
----------------------

        ----------\n");
        for(int i=0;i<n;i++)
        printf("|\tP%d\t|",p[i]);

printf("\n----------------------------------------------------------
----------------------

        ----------\n");
        for(int i=0;i<n;i++)
        printf("%d\t \t",wt[i]);
        printf("%d",tt[n-1]);

printf("\n----------------------------------------------------------
-----

        ----------\n");
}
```

OUTPUT

```
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$ gcc q7.2.c
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$ ./a.out
Enter the number of Processes : 4

Enter the Burst Time of Each Process :
P1 : 4
P2 : 8
P3 : 3
P4 : 7

Process         Burst Time      Waiting Time    Turnaround Time
P1                      4                       4
P2                      8                       12
P3                      3                       15
P4                      7                       22

Average Waiting Time : 7.75
Average Turnaround Time : 13.25

Gantt Chart

--------------------------------------------------------------------
|      P3      |      P1      |      P4      |      P2      |
--------------------------------------------------------------------
0              3              7              14             22
--------------------------------------------------------------------
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$
```

PROGRAM

```
//FCFS Scheduling
#include<stdio.h>
void main() {
        int p[20],wt[20],bt[20],tt[20],n,temp;
```

```
        float wt_avg=0,tt_avg=0;
        printf("Enter the number of Processes : ");
        scanf("%d",&n);
        printf("\nEnter the Burst Time of Each Process
:\n");

        for(int i=0;i<n;i++) {
                p[i]=i+1;
                printf("P%d : ",i+1);
                scanf("%d",&bt[i]);
        }
        wt[0]=0;
        for(int i=1;i<n;i++) {
                wt[i]=bt[i-1]+wt[i-1];
                wt_avg+=wt[i];
        }
        wt_avg/=n;
        for(int i=0;i<n;i++) {
                tt[i]=wt[i]+bt[i];
                tt_avg+=tt[i];
        }
        tt_avg/=n;
        printf("\nProcess\t\t Burst Time\t\t Waiting
Time\t\t Turnaround Time\n");
        for(int i=0;i<n;i++)

printf("P%d\t\t\t%d\t\t\t%d\t\t\t%d\n",p[i],bt[i],wt[i]
,tt[i]);
        printf("\nAverage Waiting Time :
%.2f",wt_avg);
        printf("\nAverage Turnaround Time :
%.2f\n",tt_avg);
        printf("\nGantt Chart\n\n");

printf("\n----------------------------------------------------
--------------------
        ----------------------\n");
        for(int i=0;i<n;i++)
        printf("|\tP%d\t|",p[i]);

printf("\n----------------------------------------------------
--------------------
        ----------------------\n");
        for(int i=0;i<n;i++)
        printf("%d\t \t",wt[i]);
        printf("%d",tt[n-1]);

printf("\n----------------------------------------------------
--------------------
        ----------------------\n");
}
```

OUTPUT

```
albatch2-17@ritis-HP-280-Pro-G6-Microtower-PC:~/csal505$ gcc q7.1.c
albatch2-17@ritis-HP-280-Pro-G6-Microtower-PC:~/csal505$ ./a.out
Enter the number of Processes : 4

Enter the Burst Time of Each Process :
P1 : 4
P2 : 8
P3 : 3
P4 : 7

Process         Burst Time              Waiting Time            Turnaround Time
P3                      3                       0                       3
P1                      4                       3                       7
P4                      7                       7                       14
P2                      8                       14                      22

Average Waiting Time : 6.00
Average Turnaround Time : 11.50

Gantt Chart

-----------------------------------------------------
|       P3      ||      P1      ||      P4      ||      P2      |
-----------------------------------------------------
0               3               7               14              22
-----------------------------------------------------
albatch2-17@ritis-HP-280-Pro-G6-Microtower-PC:~/csal505$
```

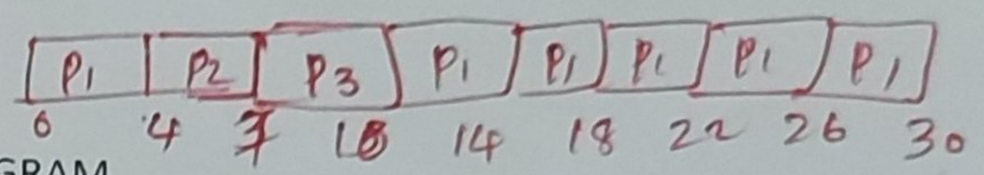
PROGRAM

```
//Round Robin Scheduling
#include<stdio.h>
void main() {
        int
i,sum,gt[20],time[20],qt,n,bt[20],tt[20],wt[20],bt_cp[2
0],p[20],temp,count=0,l=0,k=1;
        float wt_avg=0,tt_avg=0;
        printf("Enter the number of Processes(Max
20) : ");
        scanf("%d",&n);
        printf("\nEnter the Burst Time of Each Process
:\n");
        for(int j=0;j<n;j++) {
                p[j]=j+1;
                printf("P%d : ",j+1);
                scanf("%d",&bt[j]);
                bt_cp[j]=bt[j];
        }
}
time[0]=0;
printf("\nEnter the Time Slice : ");
scanf("%d",&qt);
while(count!=n) {
        for(i=0,count=0;i<n;i++) {
                if(bt_cp[i]==0) {
                        count++;
                        continue;
                }
                if(bt_cp[i]>qt) {
                        bt_cp[i]-=qt;
                        temp=qt;
                }
                else
                        if(bt_cp[i]<=qt &&
bt_cp[i]>0) {

temp=bt_cp[i];
                                bt_cp[i]=0;
                }
sum+=temp;
tt[i]=sum;
gt[l++]=p[i];
time[k]=time[k-1]+temp;
k++;
```

```
                }
        }
    for(i=0;i<n;i++) {
        wt[i]=tt[i]-bt[i];
        wt_avg+=wt[i];
        tt_avg+=tt[i];
    }
    wt_avg/=n;
    tt_avg/=n;
    printf("\nProcess\t\t Burst Time\t\t Waiting
Time\t\t Turnaround Time\n");
    for(i=0;i<n;i++)

printf("P%d\t\t\t%d\t\t\t%d\t\t\t%d\n",p[i],bt[i],wt[i]
,tt[i]);
    printf("\nAverage Waiting Time : %.2f
ms",wt_avg);
    printf("\nAverage Turnaround Time : %.2f
ms\n",tt_avg);
    printf("\nGantt Chart\n\n");

printf("\n----------------------------------------------------------
------

----------------------------------------------------------------------
--\n");
    for(i=0;i<l;i++)
        printf("|\tP%d\t|",gt[i]);

printf("\n----------------------------------------------------------
------

----------------------------------------------------------------------
---\n");
    for(i=0;i<k;i++)
        printf("%d\t \t",time[i]);

printf("\n----------------------------------------------------------
------

----------------------------------------------------------------------
---\n");
}
```

OUTPUT

```
albatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csa1505$ gcc q7_3.c
albatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csa1505$ ./a.out
Enter the number of Processes(Max 20) : 3

Enter the Burst Time of Each Process :
P1 : 6
P2 : 3
P3 : 3

Enter the Time Slice : 3

Process         Burst Time              Waiting Time            Turnaround Time
P1                      6                       6                       12
P2                      3                       3                       6
P3                      3                       6                       9

Average Waiting Time : 5.00 ms
Average Turnaround Time : 9.00 ms

Gantt Chart

-------------------------------------------------
|       P1      ||      P2      ||      P3      ||      P1      |
-------------------------------------------------
0               3               6               9               12
albatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csa1505$
```



PROGRAM

```
//Priority Scheduling
#include<stdio.h>
void main() {
        int p[20],wt[20],bt[20],tt[20],prty[20],n,temp;
        float wt_avg=0,tt_avg=0;
        printf("Enter the number of Processes : ");
        scanf("%d",&n);
        printf("\nEnter the Priority & Burst Time of
Each Process :\n");
        for(int i=0;i<n;i++) {
                p[i]=i+1;
                printf("P%d : ",i+1);
                scanf("%d%d",&prty[i],&bt[i]);
        }
        for(int i=1;i<n;i++)
                for(int j=0;j<n-i;j++)
                        if(prty[j]>prty[j+1]) {
                                temp=p[j];
                                p[j]=p[j+1];
                                p[j+1]=temp;
                                temp=bt[j];
                                bt[j]=bt[j+1];
                                bt[j+1]=temp;
                                temp=prty[j];
                                prty[j]=prty[j+1];
                                prty[j+1]=temp;

                        }
wt[0]=0;
for(int i=1;i<n;i++) {
        wt[i]=bt[i-1]+wt[i-1];
        wt_avg+=wt[i];
}
wt_avg/=n;
for(int i=0;i<n;i++) {
        tt[i]=wt[i]+bt[i];
        tt_avg+=tt[i];
```

```
        }
        tt_avg/=n;
        printf("\nProcess\t\t Burst Time\t\t Waiting
Time\t\t
                                        Turnaround
Time\n");
        for(int i=0;i<n;i++)

printf("P%d\t\t\t%d\t\t\t%d\t\t\t%d\n",p[i],bt[i],wt[i]
,tt[i]);
        printf("\nAverage Waiting Time :
%.2f",wt_avg);
        printf("\nAverage Turnaround Time :
%.2f\n",tt_avg);
        printf("\nGantt Chart\n\n");

printf("----------------------------------------------------------
-----

----------\n");
        for(int i=0;i<n;i++)
        printf("|\tP%d\t|",p[i]);

printf("----------------------------------------------------------
-----

----------\n");
        for(int i=0;i<n;i++)
                printf("%d\t \t",wt[i]);
        printf("%d",tt[n-1]);

printf("----------------------------------------------------------
-----

----------\n");
}
```

OUTPUT

```
albatch2-1@pts-49-Z80-Pro-G6-Microtower-PC:~/csal505$ gcc q7.4.c
albatch2-1@pts-49-Z80-Pro-G6-Microtower-PC:~/csal505$ ./a.out
Enter the number of Processes : 5

Enter the Priority & Burst time of Each Process :
P1 : 3 10
P2 : 5 4
P3 : 2 5
P4 : 1 1
P5 : 5 1

Process         Burst Time      Waiting Time    Turnaround Time
P4              1               0               1
P3              5               1               6
P1              10              6               16
P2              4               16              20
P5              1               20              21

Average Waiting Time : 8.60
Average Turnaround Time : 12.80

Gantt Chart

|      P4      |      P5      |      P2      |      P3      |      P1      |
0              1              6             11             13             16
albatch2-1@pts-49-Z80-Pro-G6-Microtower-PC:~/csal505$
```

EXP 6: PREEMPTIVE SCHEDULING ALGORITHMS

PROGRAM

```
//SRTF Scheduling
#include <stdio.h>
int main() {
   int
n,at[10],bt[10],ct[10],wt[10],temp[10],tat[10],p[10],s
mallest,count=0,time;
   double avg_wt=0,avg_tt=0,end=0;
   printf("Enter the no. of Process: ");
   scanf("%d",&n);
   printf("Enter the AT and BT of processes\n");
   for(int i=0;i<n;i++) {
      printf("P%d:", i + 1);
      scanf("%d %d",&at[i],&bt[i]);
      temp[i]=bt[i];
   }
   bt[9] = 9876;
   for(time=0;count!=n;time++) {
      smallest=9;
      for(int i=0;i<n;i++)
         if(at[i]<=time&&bt[i]<bt[smallest]&&bt[i]>0)
            smallest=i;
      bt[smallest]--;
      if(bt[smallest]==0) {
         count++;
         end=time+1;
         ct[smallest]=end;
         wt[smallest]=end-at[smallest]-temp[smallest];
         tat[smallest]=end-at[smallest];
      }
   }
   printf("\n----------------------------------------");
   printf("\n Prcs\tAT\tBT\tCT\tTAT\tWT\n");
   printf("----------------------------------------\n");
   for(int i=0;i<n;i++) {
      printf("
P%d\t%d\t%d\t%d\t%d\t%d\n",i+1,at[i],temp[i],ct[i],t
at[i],wt[i]);
      avg_tt+=tat[i];
      avg_wt+=wt[i];
   }
   printf("----------------------------------------");
   printf("\nAverage Turn Around Time is :
%lf\n",avg_tt/n);
   printf("Average Waiting Time is : %lf\n",avg_wt/n);
}
```

OUTPUT

	WT	TT
P1	13	16
P2	6	11
P3	11	13
P4	0	1
P5	1	6

4 — 31/5 — 47

Avg WT = 6.2

TT = 47/5 = 9.4

```
Command Prompt

D:\OS LAB\Pardhiv50>gcc -o SRTF.exe SRTF.c

D:\OS LAB\Pardhiv50>SRTF
Enter the no. of Process: 4
Enter the AT and BT of processes
P1:0 5
P2:2 4
P3:3 1
P4:5 2

-------------------------------------------
 Prcs   AT     BT     CT     TAT    WT
-------------------------------------------
 P1     0      5      6      6      1
 P2     2      4      12     10     6
 P3     3      1      4      1      0
 P4     5      2      8      3      1
-------------------------------------------
Average Turn Around Time is : 5.000000
Average Waiting Time is : 2.000000

D:\OS LAB\Pardhiv50>
```

PROGRAM

```
//Pre-Emptive Priority Scheduling
#include <stdio.h>
int main() {
   int
n,at[10],bt[10],ct[10],wt[10],temp[10],tat[10],p[10],pr
[10],smallest,count=0,time;
   double avg_wt=0,avg_tt=0,end=0;
   printf("Enter the no. of Process: ");
   scanf("%d",&n);
   printf("Enter the AT, BT and Priority of
processes\n");
   for(int i=0;i<n;i++) {
      printf("P%d:", i + 1);
      scanf("%d%d%d",&at[i],&bt[i],&pr[i]);
      temp[i]=bt[i];
   }
   bt[9] = 9876;
   for(time=0;count!=n;time++) {
      smallest=9;
      for(int i=0;i<n;i++)
         if(at[i]<=time&&pr[i]<pr[smallest]&&bt[i]>0)
            smallest=i;
      bt[smallest]--;
      if(bt[smallest]==0) {
         count++;
         end=time+1;
         ct[smallest]=end;
         wt[smallest]=end-at[smallest]-temp[smallest];
         tat[smallest]=end-at[smallest];
      }
   }
   printf("\n---------------------------------------------");
   printf("\n Prcs\tPrty\tAT\tBT\tCT\tTAT\tWT\n");
   printf("---------------------------------------------\n");
   for(int i=0;i<n;i++) {
      printf("
P%d\t%d\t%d\t%d\t%d\t%d\t%d\n",i+1,pr[i],at[i],tem
p[i],ct[i],tat[i],wt[i]);
      avg_tt+=tat[i];
      avg_wt+=wt[i];
   }
   printf("---------------------------------------------");
   printf("\nAverage Turn Around Time is :
%lf\n",avg_tt/n);
   printf("Average Waiting Time is : %lf\n",avg_wt/n);
}
```

OUTPUT

```
Command Prompt

D:\OS LAB\Pardhiv50>gcc -o PEPriority.exe PEPriority.c

D:\OS LAB\Pardhiv50>PEPriority
Enter the no. of Process: 4
Enter the AT, BT and Priority of processes
P1:0 5 3
P2:2 4 1
P3:3 1 2
P4:5 2 4

-------------------------------------------------------
 Prcs   Prty   AT     BT     CT     TAT    WT
-------------------------------------------------------
 P1     3      0      5      10     10     5
 P2     1      2      4      6      4      0
 P3     2      3      1      7      4      3
 P4     4      5      2      12     7      5
-------------------------------------------------------
Average Turn Around Time is : 6.250000
Average Waiting Time is : 3.250000

D:\OS LAB\Pardhiv50>
```

EXP 7: IPC USING SHARED MEMORY

PROGRAM: WRITER PROCESS

```
#include<sys/ipc.h>
#include<sys/shm.h>
#include<unistd.h>
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
void main() {
        int id;
        void *sm;
        char buf[100];

id=shmget((key_t)1222,1024,0666|IPC_CREAT);
        printf("Key of Shared Memory is %d\n",id);
        sm=shmat(id,NULL,0);
        printf("Process attached at %p\n",sm);
        printf("Enter the data to be written:\n");
        read(0,buf,100);
        strcpy(sm,buf);
        printf("Written Data is:\n%s\n",(char *)sm);
}
```

OUTPUT

```
[illegible]$ gcc ipcw.c
[illegible]$ ./a.out
Key of Shared Memory is 98333
Process attached at 0x7f5cb2c58000
Enter the data to be written:
7 4
Written Data is:
7 4
```

PROGRAM: READER PROCESS

```c
#include<sys/ipc.h>
#include<sys/shm.h>
#include<unistd.h>
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
void main() {
        int id;
        void *sm;
        char buf[100];
        id=shmget((key_t)1222,1024,0666);
        printf("Key of Shared Memory is %d\n",id);
        sm=shmat(id,NULL,0);
printf("Process attached at %p\n",sm);
printf("Data read from Memory:\n%s\n",(char *)sm);
strcpy(buf,sm);
int a=buf[0]-'0';
int b=buf[2]-'0';
printf("The Sum is %d\n",a+b);
}
```

OUTPUT

```
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$ gcc ipcr.c
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$ ./a.out
Key of Shared Memory is 98333
Process attached at 0x7f262b0d3000
Data read from Memory:
7 4

The Sum is 11
```

EXP 8: PRODUCER-CONSUMER PROBLEM PROGRAM

```
#include<stdio.h>
#include<semaphore.h>
#include<pthread.h>
#include<unistd.h>
#include<stdlib.h>
sem_t mutex;
sem_t empty;
sem_t full;
int buffer[8];
pthread_t p[5];
pthread_t c[5];
void producer(int *p) {
        int a[10],i=0,n=*(int*)p;
        while(i<=5) {
                sem_wait(&empty);
                sem_wait(&mutex);
                a[i]=3;
                printf("Producer %d Produced Item
%d\n",n,i);
                sleep(1);
                i++;
                buffer[i]=a[i];
                sem_post(&mutex);
                sem_post(&full);
        }
}
void consumer(void *p) {
        int b[10],i=0,n=*(int*)p;
        while(i<=5) {
                sem_wait(&full);
                sem_wait(&mutex);
                printf("Consumer %d Consumes Item
%d\n",n,i);
                sleep(1);
                b[i]=buffer[i];
                i++;
                sem_post(&mutex);
                sem_post(&empty);
        }
}
void main()
{
        int n;
        sem_init(&mutex,0,1);
        sem_init(&empty,0,5);
        sem_init(&full,0,0);
        for(n=0;n<5;n++) {
                pthread_create(&p[n],0,(void
*)producer,(void *)&n);
                pthread_create(&c[n],0,(void
*)consumer,(void *)&n);
        }while(1);
}
```

OUTPUT

```
a1batch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csa150$ gcc prodcon.c -lpthread
a1batch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csa150$ ./a.out
Producer 0 Produced Item 0
Producer 1 Produced Item 0
Producer 2 Produced Item 0
Producer 4 Produced Item 0
Producer 5 Produced Item 0
Consumer 1 Consumes Item 0
Consumer 2 Consumes Item 0
Consumer 3 Consumes Item 0
Consumer 4 Consumes Item 0
Consumer 5 Consumes Item 0
Producer 0 Produced Item 1
Producer 1 Produced Item 1
Consumer 2 Consumes Item 1
Producer 4 Produced Item 1
Producer 5 Produced Item 1
Consumer 1 Consumes Item 1
Producer 2 Produced Item 1
Producer 0 Produced Item 2
Consumer 3 Consumes Item 1
Consumer 4 Consumes Item 1
Producer 1 Produced Item 2
Consumer 5 Consumes Item 1
Consumer 2 Consumes Item 2
^C
a1batch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csa150$
```

EXP 9: DINING PHILOSOPHER PROBLEM PROGRAM

```
#include<stdio.h>
#include<semaphore.h>
#include<pthread.h>
#include<unistd.h>
#include<stdlib.h>
#define N 5
#define LEFT (i+4)%5
#define RIGHT (i)%5
#define THINKING 0
#define HUNGRY 1
#define EATING 2
int state[N];
pthread_t t[N];
sem_t s[N];
sem_t mutex;
void think(int n) {
        printf("The Philosopher %d is thinking \n",n);
        sleep(1);
}
void eat(int n) {
        printf("Philosopher %d is eating\n",n);
        sleep(1);
        printf("Philosopher %d finished eating\n",n);
}
void take_fork(int i) {
        sem_wait(&mutex);
        state[i]=HUNGRY;
        if(state[i]==HUNGRY && state[LEFT]!=EATING
&& state[RIGHT]!=EATING) {
                state[i]=EATING;
                sem_wait(&s[LEFT]);
                sem_wait(&s[RIGHT]);
        }
        sem_post(&mutex);
}
void putforks(int i) {
        state[i]=THINKING;
        sem_post(&s[LEFT]);
```

```
        sem_post(&s[RIGHT]);
}
void *philo(int n) {
        while(1) {
                think(n);
                take_fork(n);
                if(state[n]==EATING) {
                        eat(n);
                        putforks(n);
                }
        }
}
void main() {
        int i;
        for(i=0;i<N;i++)
                sem_init(&s[i],0,1);
        for(i=0;i<N;i++)
sem_init(&mutex,0,1);
        for(i=0;i<N;i++) {
                pthread_create(&t[i],0,(void
*)philo,(void *)i);
        }while(1);
}
```

OUTPUT

```
atbatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csal50$ gcc philo.c -lpthread
philo.c: In function 'main':
philo.c:57:40: warning: cast to pointer from integer of different size [-Wint-to-pointer-cast]
   57 |   pthread_create(&t[i],0,(void *)philo,(void *)i);
      |                                        ^
atbatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csal50$ ./a.out
The Philosopher 0 is thinking
The Philosopher 3 is thinking
The Philosopher 2 is thinking
The Philosopher 4 is thinking
The Philosopher 1 is thinking
Philosopher 0 is eating
Philosopher 3 is eating
Philosopher 0 finished eating
The Philosopher 0 is thinking
Philosopher 3 finished eating
The Philosopher 3 is thinking
Philosopher 2 is eating
Philosopher 2 finished eating
The Philosopher 2 is thinking
Philosopher 1 is eating
Philosopher 4 is eating
The Philosopher 0 is thinking
Philosopher 1 finished eating
The Philosopher 1 is thinking
Philosopher 4 finished eating
The Philosopher 4 is thinking
Philosopher 3 is eating
^C
atbatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csal50$
```

EXP 10: BANKER'S ALGORITHM FOR DEADLOCK AVOIDANCE

PROGRAM

```
#include<stdio.h>
#include<stdbool.h>
#include<stdlib.h>
void exit(int status);
int i,j,no,res;
int safety(int[][10],int[],int[][10],int[]);
void output(int [][10]);
void request(int[][10],int[][10],int[],int[],int);
void main(){
        int
ans,id,seq[10],R[10],A[10][10],C[10][10],N[10][10],W[
10]={0},req[10],AV[10];
        printf("\nEnter the number of Processes (less
than 10): ");
        scanf("%d",&no);
        printf("Enter the number of resources (less
than 10): ");
        scanf("%d",&res);
        printf("Enter the max available instances of
each resource : \n");
        for(i=0;i<res;i++){
                printf("R%d:",i);
                scanf("%d",&R[i]);
        }
        printf("\nEnter the Allocated Resource Table :
\n");
        for(i=0;i<no;i++)
                for(j=0;j<res;j++)
                        scanf("%d",&A[i][j]);
        printf("\nEnter the Maximum Claim Table :
\n");
        for(i=0;i<no;i++)
                for(j=0;j<res;j++)
                        scanf("%d",&C[i][j]);
        for(i=0;i<no;i++)
                for(j=0;j<res;j++)
                        N[i][j]=C[i][j]-A[i][j];
        printf("\nAllocated Resource Table:\n\t");
        output(A);
        printf("\n\nMaximum Claim Table:\n\t");
        output(C);
        printf("\n\nNeed Matrix:\n\t");
        output(N);
        for(j=0;j<res;j++){
                for(i=0;i<no;i++)
                        W[j]+=A[i][j];
                W[j]=R[j]-W[j];
        }
        for(j=0;j<res;j++)
                AV[j]=W[j];
        int ch=safety(A,W,N,seq);
        if(ch==1){
                printf("\n\nThe System is in SAFE
STATE :)\n");
                printf("\nThe Safe Sequence : ");
                for(i=0;i<no;i++)
                        printf("|P%d|",seq[i]);
                printf("\nResource Request for a
Process Needed? (1=YES,0=NO): ");
                scanf("%d",&ans);
                if(ans==1){
                        printf("Enter the Process ID
for initiating request: ");
                        scanf("%d",&id);
                        printf("Enter the Request
Vector for P%d : ",id);
                        for(j=0;j<res;j++)
                                scanf("%d",&req[j]);
                        request(A,N,req,AV,id);
                }
        }
```

```
}
void output(int arr[10][10]){
        for(i=0;i<res;i++)
        printf("R%d\t",i);
        for(i=0;i<no;i++){
                printf("\nP%d\t",i);
                for(j=0;j<res;j++)
                        printf("%d\t",arr[i][j]);
        }
}
int safety(int A[10][10],int W[10],int N[10][10],int
seq[]){
        int x=0,flg=0,target=0;
        int finish[10];
        for(i=0;i<no;i++)
                finish[i]=0;
for(int w=0;w<no;w++){
        label:
        for(int i=0;i<no;i++){
                flg=0;
                for(int j=0;j<res;j++)
                        if(N[i][j] > W[j])
                                flg++;
                if(flg==0 && finish[i]==0){
                        for(j=0;j<res;j++)
                                W[j]+=A[i][j];
                        finish[i]=1;
                        target++;
                        seq[x++]=i;
                        goto label;
                }
                continue;
        }
}
        if(target==no)
                return 1;
        else
                printf("\nThe System is in UNSAFE
STATE :(\n");
}
void request(int A[10][10],int N[10][10],int req[10],int
AV[10],int ID){
        int seq[10];
        for(i=0;i<res;i++)
        if(req[i] > N[ID][i]){
                printf("\nNOT POSSIBLE as the
process P%d has exceeded its max claim !!",ID);
                exit(0);
        }
        for(i=0;i<res;i++)
                if(req[i] > AV[i]){
                        printf("\nThe process P%d has
to wait since resouces are not

                available yet!!",ID);
                        exit(0);
                }
        for(j=0;j<res;j++){
                AV[j] -= req[j];
                A[ID][j] += req[j];
                N[ID][j] -= req[j];
        }
        int ch=safety(A,AV,N,seq);
        if(ch==1){
                printf("\nThe System is in SAFE state.
Hence, the resources can be allocated\n");
                printf("\nThe Safe Sequence : ");
                for(i=0;i<no;i++)
                        printf("|P%d|",seq[i]);
        }
        printf("\n");
}
```

OUTPUT

```
Command Prompt

D:\OS LAB\Pardhiv50>gcc Bankers.c

D:\OS LAB\Pardhiv50>a.exe

Enter the number of Processes (less than 10): 5
Enter the number of resources (less than 10): 3
Enter the max available instances of each resource :
R0:10
R1:5
R2:7

Enter the Allocated Resource Table :
0 1 0
2 0 0
3 0 2
2 1 1
0 0 2

Enter the Maximum Claim Table :
7 5 3
3 2 2
9 0 2
2 2 2
4 3 3

Allocated Resource Table:
        R0      R1      R2
P0      0       1       0
P1      2       0       0
P2      3       0       2
P3      2       1       1
P4      0       0       2

Maximum Claim Table:
        R0      R1      R2
P0      7       5       3
P1      3       2       2
P2      9       0       2
P3      2       2       2
P4      4       3       3

Need Matrix:
        R0      R1      R2
P0      7       4       3
P1      1       2       2
P2      6       0       0
P3      0       1       1
P4      4       3       1

The System is in SAFE STATE :)

The Safe Sequence : |P1||P3||P0||P2||P4|
Resource Request for a Process Needed? (1=YES,0=NO): 1
Enter the Process ID for initiating request: 0
Enter the Request Vector for P0 : 0 2 0

The System is in SAFE state. Hence, the resources can be allocated

The Safe Sequence : |P3||P1||P0||P2||P4|

D:\OS LAB\Pardhiv50>
```

EXP 11: DEADLOCK DETECTION ALGORITHM

PROGRAM

```
#include<stdio.h>
#include<conio.h>
int max[100][100];
int alloc[100][100];
int need[100][100];
int avail[100];
int n,r;
void input();
void show();
void cal();
int main(){
        int i,j;
        input();
        show();
        cal();
        getch();
        return 0;
}
void input(){
        int i,j;
        printf("Enter the no of Processes:");
        scanf("%d",&n);
        printf("Enter the no of resource instances:");
        scanf("%d",&r);
        printf("Enter the Max Matrix:\n");
        for(i=0;i<n;i++)
                for(j=0;j<r;j++)
                        scanf("%d",&max[i][j]);
        printf("\nEnter the Allocation Matrix:\n");
        for(i=0;i<n;i++)
                for(j=0;j<r;j++)
                        scanf("%d",&alloc[i][j]);
        printf("\nEnter the available Resources:\n");
        for(j=0;j<r;j++)
                scanf("%d",&avail[j]);
}
void show(){
        int i,j;
        printf("\nProcess\t  Allocation\t  Max\t\t
Available");
        for(i=0;i<n;i++){
                printf("\nP%d\t  ",i+1);
                for(j=0;j<r;j++)
                        printf("%d ",alloc[i][j]);
                printf("\t  ");
                for(j=0;j<r;j++)
                        printf("%d ",max[i][j]);
                printf("\t  ");
                if(i==0)
                        for(j=0;j<r;j++)
                                printf("%d ",avail[j]);
        }
}
void cal(){
        int
finish[100],temp,need[100][100],flag=1,k,c1=0;
        int dead[100],safe[100],i,j;
        for(i=0;i<n;i++)
                finish[i]=0;
        for(i=0;i<n;i++)
                for(j=0;j<r;j++)
                        need[i][j]=max[i][j]-alloc[i][j];
        while(flag){
                flag=0;
                for(i=0;i<n;i++){
                        int c=0;
                        for(j=0;j<r;j++){

if((finish[i]==0)&&(need[i][j]<=avail[j])){
                                        c++;
                                        if(c==r){
                                for(k=0;k<r;k++){
                                        avail[k]+=alloc[i][j];
                                        finish[i]=1;
                                        flag=1;
                                }
                        if(finish[i]==1){
                                        i=n;
                                }
                        }
                }
        }
        j=0;
        flag=0;
        for(i=0;i<n;i++){
                if(finish[i]==0){
                        dead[j]=i;
                        j++;
                        flag=1;
                }
        }
        if(flag==1){
                printf("\n\nSystem is in Deadlock and
the Deadlock process are\n");
                for(i=0;i<n;i++){
                        printf("P%d\t",dead[i]);
                }
        }
        else
                printf("\nNo Deadlock Occur");
}
```

OUTPUT

Command Prompt

```
D:\OS LAB\Pardhiv50>gcc deadlock.c

D:\OS LAB\Pardhiv50>a.exe
Enter the no of Processes:3
Enter the no of resource instances:3
Enter the Max Matrix:
3 6 8
4 3 3
3 4 4

Enter the Allocation Matrix:
3 3 3
2 0 3
1 2 4

Enter the available Resources:
1 2 0

Process     Allocation      Max             Available
P1          3 3 3           3 6 8           1 2 0
P2          2 0 3           4 3 3
P3          1 2 4           3 4 4

System is in Deadlock and the Deadlock process are
P0      P1      P2
D:\OS LAB\Pardhiv50>
```

EXP 12: MEMORY ALLOCATION

PROGRAM

```
#include<stdio.h>
int p,b,block[10],process[10],b_copy[10],remain[10];
void first_fit();
void best_fit();
void worst_fit();
void main() {
        int i;
        printf("How many process & blocks ? :");
        scanf("%d%d",&p,&b);
        printf("\nProcess sizes ?\n");
        for(i=0;i<p;i++) {
                printf("P%d: ",i+1);
                scanf("%d",&process[i]);
        }
        printf("\nBlock sizes ?\n");
        for(i=0;i<b;i++) {
                printf("B%d: ",i+1);
                scanf("%d",&block[i]);
        }
        printf("\nFIRST FIT:\n");
        first_fit();
        printf("\nBEST FIT:\n");
        best_fit();
        printf("\nWORST FIT:\n");
        worst_fit();
}

void first_fit() {
        int i,j,p_flg[10]={0},b_flg[10]={0};
        for(i=0;i<b;i++)
                remain[i]=b_copy[i]=block[i];
        printf("Process Name\tProcess Size\tBlock
name\tTotal
                                        Space\tWastage
Space\n");
        for(i=0;i<p;i++) {
                for(j=0;j<b;j++)
                        if(process[i]<=block[j] &&
b_flg[j]==0) {

remain[j]=block[j]-process[i];

printf("P%d\t\t%d\t\tB%d\t\t%d\t\t%d\n"
,i+1,process[i],j+1,block[j],remain[j]);

p_flg[i]=b_flg[j]=1;break;
                        }
                if(p_flg[i]==0)

printf("P%d\t\t%d\t\t-\t\t-\t\t-\n",i+1,process[i]);
        }
}

void best_fit() {
        int i,j,p_flg[10]={0},b_flg[10]={0},b_index[10];
        for(i=0;i<b;i++) {
                b_index[i]=i+1;
                remain[i]=b_copy[i]=block[i];
        }
        for(i=1;i<b;i++)
                for(j=0;j<b-i;j++)
                        if(block[j]>block[j+1]) {
                                int temp=block[j];
                                block[j]=block[j+1];
                                block[j+1]=temp;
                                temp=b_index[j];

b_index[j]=b_index[j+1];
                                b_index[j+1]=temp;
                        }
        printf("Process Name\tProcess Size\tBlock
name
                                \tTotal Space\tWastage
Space\n");
        for(i=0;i<p;i++) {
                for(j=0;j<b;j++)
                        if(process[i]<=block[j] &&
b_flg[j]==0) {

remain[j]=block[j]-process[i];

printf("P%d\t\t%d\t\tB%d\t\t%d\t\t%d\n",

i+1,process[i],b_index[j],block[j],remain[j]);

p_flg[i]=b_flg[j]=1;break;
                        }
                if(p_flg[i]==0)

printf("P%d\t\t%d\t\t-\t\t-\t\t-\n",i+1,process[i]);
        }
}
void worst_fit() {
        int i,j,p_flg[10]={0},b_flg[10]={0},b_index[10];
        for(i=0;i<b;i++) {
                b_index[i]=i+1;
                remain[i]=b_copy[i];
        }
        for(i=1;i<b;i++)
                for(j=0;j<b-i;j++)
                        if(b_copy[j]<b_copy[j+1]) {
                                int temp=b_copy[j];

b_copy[j]=b_copy[j+1];
                                b_copy[j+1]=temp;
                                temp=b_index[j];

b_index[j]=b_index[j+1];
                                b_index[j+1]=temp;
                        }
```

```
		printf("Process Name\tProcess Size\tBlock
name
					\tTotal Space\tWastage
Space\n");
		for(i=0;i<p;i++) {
			for(j=0;j<b;j++)
				if(process[i]<=b_copy[j] &&
b_flg[j]==0) {

remain[j]=b_copy[j]-process[i];

printf("P%d\t\t%d\t\tB%d\t\t%d\t\t%d\n",

i+1,process[i],b_index[j],b_copy[j],remain[j]);

p_flg[i]=b_flg[j]=1;break;
				}
			if(p_flg[i]==0)

printf("P%d\t\t%d\t\t-\t\t-\t\t-\n",i+1,process[i]);
		}
}
```

OUTPUT

```
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$ gcc fixedtable.c
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$ ./a.out
How many process & blocks ? :4 6

Process sizes ?
P1: 357
P2: 210
P3: 468
P4: 491

Block sizes ?
B1: 200
B2: 400
B3: 600
B4: 500
B5: 300
B6: 250

FIRST FIT:
Process Name    Process Size    Block name      Total Space     Wastage Space
P1              357             B2              400             43
P2              210             B3              600             390
P3              468             B4              500             32
P4              491             -               -               -

BEST FIT:
Process Name    Process Size    Block name      Total Space     Wastage Space
P1              357             B2              400             43
P2              210             B6              250             40
P3              468             B4              500             32
P4              491             B3              600             109

WORST FIT:
Process Name    Process Size    Block name      Total Space     Wastage Space
P1              357             B3              600             243
P2              210             B4              500             290
P3              468             -               -               -
P4              491             -               -               -
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$
```

EXP 13: PAGE REPLACEMENT ALGORITHMS

PROGRAM: FIRST IN FIRST OUT (FIFO)

```
#include<stdio.h>
int main() {
		int frames,l,i,j,k,exist,m[10],str[100],count=0;
		printf("Enter the length of the reference
string: ");
		scanf("%d",&l);
		printf("Enter the reference string: ");
		for(i=0;i<l;i++)
			scanf("%d",&str[i]);
		printf("Enter the no. of partitions: ");
		scanf("%d",&frames);
		for(i=0;i<frames;i++)
			m[i]=-1;
		printf("\nThe Page Replacement Process
is....\n");
		for(i=0;i<l;i++,exist=0) {
			for(j=0;j<frames;j++)
				if(m[j]==str[i])
					exist=1;
			if(exist==0) {
				m[count%frames]=str[i];
				count++;
			}
			for(k=0;k<frames;k++)
				printf("%d\t",m[k]);
			if(exist==0)
				printf("Page Fault:
%d\n",count);
			else
				printf("HIT!\n");
		}
		printf("\nTotal Page Fault = %d\n",count);
		printf("Total Hits = %d\n",l-count);
		printf("Miss Ratio = %d%%\n",(count*100)/l);
		printf("Hit Ratio =
%d%%\n\n",((l-count)*100)/l);

}
```

OUTPUT

```
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$ gcc fifoallo.c
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$ ./a.out
Enter the length of the reference string: 20
Enter the reference string: 7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1
Enter the no. of partitions: 3

The Page Replacement Process is....
7       -1      -1      Page Fault: 1
7       0       -1      Page Fault: 2
7       0       1       Page Fault: 3
2       0       1       Page Fault: 4
2       0       1       HIT!
2       3       1       Page Fault: 5
2       3       0       Page Fault: 6
4       3       0       Page Fault: 7
4       2       0       Page Fault: 8
4       2       3       Page Fault: 9
0       2       3       Page Fault: 10
0       2       3       HIT!
0       2       3       HIT!
0       1       3       Page Fault: 11
0       1       2       Page Fault: 12
0       1       2       HIT!
0       1       2       HIT!
7       1       2       Page Fault: 13
7       0       2       Page Fault: 14
7       0       1       Page Fault: 15

Total Page Fault = 15
Total Hits = 5
Miss Ratio = 75%
Hit Ratio = 25%

aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$
```

PROGRAM: LEAST RECENTLY USED (LRU)

```
#include <stdio.h>
void main() {
		int
str[20],flg[20],i,j,f,len,count[20],m[10],next=0,min,pf=
0;
		printf("Enter the length of Reference String:");
```

```
scanf("%d",&len);
printf("Enter the Reference String");
for(i=0;i<len;i++) {
        scanf("%d",&str[i]);
        flg[i]=0;
}
printf("Enter the no. of Frames:");
scanf("%d",&f);
for(i=0;i<f;i++) {
        count[i]=0;
        m[i]=-1;
}
printf("\nTHE PAGE REPLACEMENT PROCESS
IS...\n");
        for(i=0;i<len;i++) {
                for(j=0;j<f;j++)
                        if(m[j]==str[i]) {
                                flg[i]=1;
                                count[j]=next++;
                        }
                if(flg[i]==0) {
                        if(i<f) {
                                m[i]=str[i];
                                count[i]=next++;
                        }
                        else {
                                min=0;
                                for(j=1;j<f;j++)

if(count[min]>count[j])

min=j;
                                m[min]=str[i];
                                count[min]=next++;
                        }
                        pf++;
                }
                for(j=0;j<f;j++)
                        printf("%d\t",m[j]);
                if(flg[i]!=1)
                        printf("Page Fault:%d\n",pf);
                else
                        printf("HIT!\n");
        }
        printf("\nTotal Page Fault = %d\n",pf);
        printf("Total Hits = %d\n",len-pf);
        printf("Miss Ratio = %d%%\n",(pf*100)/len);
        printf("Hit Ratio =
%d%%\n\n",((len-pf)*100)/len);
}
```

OUTPUT

```
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$ gcc lru.c
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$ ./a.out
Enter the length of Reference String:20
Enter the Reference String 7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1
Enter the no. of Frames:3

THE PAGE REPLACEMENT PROCESS IS...
7       -1      -1      Page Fault:1
7       0       -1      Page Fault:2
7       0       1       Page Fault:3
2       0       1       Page Fault:4
2       0       1       HIT!
2       0       3       Page Fault:5
2       0       3       HIT!
4       0       3       Page Fault:6
4       0       2       Page Fault:7
4       3       2       Page Fault:8
0       3       2       Page Fault:9
0       3       2       HIT!
0       3       2       HIT!
1       3       2       Page Fault:10
1       3       2       HIT!
1       0       2       Page Fault:11
1       0       2       HIT!
1       0       7       Page Fault:12
1       0       7       HIT!
1       0       7       HIT!

Total Page Fault = 12
Total Hits = 8
Miss Ratio = 60%
Hit Ratio = 40%

aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$
```

PROGRAM: LEAST FREQUENTLY USED (LFU)

```
#include <stdio.h>
void main() {
        int str[20],flg[20]={0},i,j,f,len,count[20]={0};
        int m[10],next=0,min,pf=0,freq[20]={0};
        printf("Enter the length of Reference String:");
        scanf("%d",&len);
        printf("Enter the Reference String");
        for(i=0;i<len;i++)
                scanf("%d",&str[i]);
        printf("Enter the no. of Frames:");
        scanf("%d",&f);
        for(i=0;i<f;i++)
                m[i]=-1;
        printf("\nTHE PAGE REPLACEMENT PROCESS
IS...\n");
        for(i=0;i<len;i++) {
                for(j=0;j<f;j++)
                        if(m[j]==str[i]) {
                                flg[i]=1;
                                count[j]=next++;
                                freq[j]++;
                        }
                if(flg[i]==0) {
                        if(i<f) {
                                m[i]=str[i];
                                count[i]=next++;
                                freq[i]=1;
                        }
                        else {
                                min=0;
                                for(j=1;j<f;j++)
                                if(freq[min]>=freq[j])

if(count[min]>count[j])

                                        min=j;
                                m[min]=str[i];
                                count[min]=next++;
                                freq[min]=1;
```

```
            }
            pf++;
        }
        for(j=0;j<f;j++)
            printf("%d\t",m[j]);
        if(flg[i]!=1)
            printf("Page Fault:%d\n",pf);
        else
            printf("HIT!\n");
}
printf("\nTotal Page Fault = %d\n",pf);
printf("Total Hits = %d\n",len-pf);
printf("Miss Ratio = %d%%\n",(pf*100)/len);

    printf("Hit Ratio =
%d%%\n\n",((len-pf)*100)/len);
}
```

OUTPUT

```
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$ gcc lfu.c
aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$ ./a.out
Enter the length of Reference String:10
Enter the Reference String 4 5 2 1 2 3 3 5 5 1
Enter the no. of Frames:3

THE PAGE REPLACEMENT PROCESS IS...
4       -1      -1      Page Fault:1
4       5       -1      Page Fault:2
4       5       2       Page Fault:3
1       5       2       Page Fault:4
1       5       2       HIT!
1       3       2       Page Fault:5
1       3       2       HIT!
5       3       2       Page Fault:6
5       3       2       HIT!
5       3       1       Page Fault:7

Total Page Fault = 7
Total Hits = 3
Miss Ratio = 70%
Hit Ratio = 30%

aibatch2-17@mits-HP-280-Pro-G6-Microtower-PC:~/csai50$
```

PROGRAM: FCFS DISC SCHEDULING

```
#include<stdio.h> #include<stdlib.h> void main() {
int n,q[100],i,diff,seek=0;    printf("Enter the size of
Queue: ");    scanf("%d",&n);    printf("Enter the
Queue: ");    for(i=1;i<=n;i++) {
    scanf("%d",&q[i]);
  }
  printf("Enter the intial head position: ");
scanf("%d",&q[0]);    for(i=0;i<n;i++) {
diff=abs(q[i]-q[i+1]);
    seek+=diff;
    printf("\nMove from %d to %d and the seek is
%d",q[i],q[i+1],diff);
  }
  printf("\n\nTotal Seek Distance is : %d",seek);
float avg=seek/n;
  printf("\nAverage Seek Distance is : %.3f",avg);
}
```

OUTPUT

Command Prompt

```
D:\OS LAB\Pardhiv50>gcc fcfsdisk.c

D:\OS LAB\Pardhiv50>a
Enter the size of Queue: 8
Enter the Queue: 98 183 37 122 14 124 65 67
Enter the intial head position: 53

Move from 53 to 98 and the seek is 45
Move from 98 to 183 and the seek is 85
Move from 183 to 37 and the seek is 146
Move from 37 to 122 and the seek is 85
Move from 122 to 14 and the seek is 108
Move from 14 to 124 and the seek is 110
Move from 124 to 65 and the seek is 59
Move from 65 to 67 and the seek is 2

Total Seek Distance is : 640
Average Seek Distance is : 80.000
D:\OS LAB\Pardhiv50>
```

PROGRAM: SCAN DISC SCHEDULING

```
#include<stdio.h> #include<stdlib.h> void main() {
  int q[100],n,seek=0,i,cur,prev,j,m,cyl,loc;    float avg;
  printf("Enter the no. of Cylinders: ");
scanf("%d",&cyl);    printf("Cylinders: 0 to
%d\n",cyl-1);    printf("Enter the Queue Size: ");
scanf("%d",&m);    n=m+1;    printf("Enter the Queue:
");    for(i=1;i<n;i++)      scanf("%d",&q[i]);
printf("Enter Current Head Position: ");
scanf("%d",&cur);    q[0]=cur;
  printf("Enter Previous Head Position: ");
scanf("%d",&prev);      for(i=1;i<n;i++)
for(j=0;j<n-i;j++)          if(q[j]>q[j+1]) {          int
temp=q[j];          q[j]=q[j+1];
          q[j+1]=temp;
        }
  printf("Displaying Requests in Order...\n");
  for(i=0;i<n;i++)      printf("%d\t",q[i]);
for(i=0;i<n;i++)      if(q[i]==cur) {          loc=i;
        break;
    }
  if(cur<prev) {
    printf("\n\nScanning towards left...then right\n");
    for(i=loc;i>=0;i--)
            printf("%d --> ",q[i]);          printf("0 -->
");      for(i=loc+1;i<n;i++)
            printf("%d --> ",q[i]);
seek=cur+q[n-1];
  }  else {
    printf("\n\nScanning towards right...then left\n");
    for(i=loc;i<n;i++)
            printf("%d --> ",q[i]);
          printf("%d --> ",cyl-1);
    for(i=loc-1;i>=0;i--)
            printf("%d --> ",q[i]);
seek=2*(cyl-1)-cur-q[0];
  }
  printf("\n\nTotal Seek Distance: %d\t",seek);
avg=(float)seek/n;
  printf("\nAverage Seek Distance: %.3f\t",avg);
```

OUTPUT

```
D:\OS LAB\Pardhiv50>gcc scandisk.c

D:\OS LAB\Pardhiv50>a
Enter the no. of Cylinders: 200
Cylinders: 0 to 199
Enter the Queue Size: 8
Enter the Queue: 98 183 37 122 14 124 65 67
Enter Current Head Position: 53
Enter Previous Head Position: 45
Displaying Requests in Order...
14      37      53      65      67      98      122     124     183

Scanning towards right...then left
53 --> 65 --> 67 --> 98 --> 122 --> 124 --> 183 --> 199 --> 37 --> 14 -->

Total Seek Distance: 331
Average Seek Distance: 36.778
D:\OS LAB\Pardhiv50>a
Enter the no. of Cylinders: 200
Cylinders: 0 to 199
Enter the Queue Size: 8
Enter the Queue: 98 183 37 122 14 124 65 67
Enter Current Head Position: 53
Enter Previous Head Position: 65
Displaying Requests in Order...
14      37      53      65      67      98      122     124     183

Scanning towards left...then right
53 --> 37 --> 14 --> 0 --> 65 --> 67 --> 98 --> 122 --> 124 --> 183 -->

Total Seek Distance: 236
Average Seek Distance: 26.222
D:\OS LAB\Pardhiv50>
```

PROGRAM: CSCAN DISC SCHEDULING

```
#include<stdio.h> #include<stdlib.h> void main() {
   int q[100],n,seek=0,i,cur,prev,j,m,cyl,loc;    float avg;
   printf("Enter the no. of Cylinders: ");
scanf("%d",&cyl);    printf("Cylinders: 0 to
%d\n",cyl-1);    printf("Enter the Queue Size: ");
scanf("%d",&m);    n=m+1;
   printf("Enter the Queue: ");
   for(i=1;i<n;i++)       scanf("%d",&q[i]);
printf("Enter Current Head Position: ");
scanf("%d",&cur);    q[0]=cur;
   printf("Enter Previous Head Position: ");
scanf("%d",&prev);       for(i=1;i<n;i++)
for(j=0;j<n-i;j++)          if(q[j]>q[j+1]) {              int
temp=q[j];              q[j]=q[j+1];
          q[j+1]=temp;
       }
   printf("Displaying Requests in Order...\n");
   for(i=0;i<n;i++)       printf("%d\t",q[i]);
for(i=0;i<n;i++)       if(q[i]==cur) {          loc=i;
       break;
    }
  if(cur<prev) {
     printf("\n\nScanning towards left...then restart at
right end\n");       for(i=loc;i>=0;i--)
                printf("%d --> ",q[i]);              printf("0 -->
");       printf("%d --> ",cyl-1);
     for(i=n-1;i>loc;i--)
                printf("%d --> ",q[i]);
     seek=cur+2*(cyl-1)-q[loc+1];
  }   else {
     printf("\n\nScanning towards right...then restart
at left end\n");       for(i=loc;i<n;i++)
                printf("%d --> ",q[i]);
             printf("%d --> ",cyl-1);
     printf("0 --> ");
      for(i=0;i<loc;i++)
                printf("%d --> ",q[i]);
seek=2*(cyl-1)-cur+q[loc-1];
  }
  printf("\n\nTotal Seek Distance: %d\t",seek);
avg=(float)seek/(n+1);
  printf("\nAverage Seek Distance: %.3f\t",avg);
}
```

OUTPUT

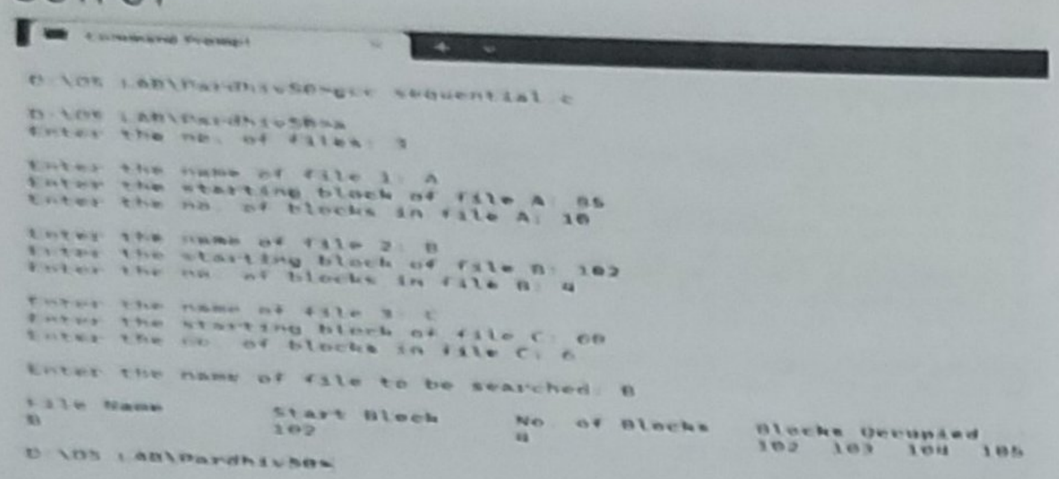
```
D:\OS LAB\Pardhiv50>gcc cscandisk.c

D:\OS LAB\Pardhiv50>a
Enter the no. of Cylinders: 200
Cylinders: 0 to 199
Enter the Queue Size: 8
Enter the Queue: 98 183 37 122 14 124 65 67
Enter Current Head Position: 53
Enter Previous Head Position: 50
Displaying Requests in Order...
14      37      53      65      67      98      122     124     183

Scanning towards right...then restart at left end
53 --> 65 --> 67 --> 98 --> 122 --> 124 --> 183 --> 199 --> 0 --> 14 --> 37 -->

Total Seek Distance: 382
Average Seek Distance: 38.200
D:\OS LAB\Pardhiv50>a
Enter the no. of Cylinders: 200
Cylinders: 0 to 199
Enter the Queue Size: 8
Enter the Queue: 98 183 37 122 14 124 65 67
Enter Current Head Position: 53
Enter Previous Head Position: 55
Displaying Requests in Order...
14      37      53      65      67      98      122     124     183

Scanning towards left...then restart at right end
53 --> 37 --> 14 --> 0 --> 199 --> 183 --> 124 --> 122 --> 98 --> 67 --> 65 -->

Total Seek Distance: 386
Average Seek Distance: 38.600
D:\OS LAB\Pardhiv50>
```

PROGRAM: SEQUENTIAL FILE ALLOCATION

```
#include<stdio.h>
#include<string.h>
struct file {
   char name[10];
   int num,start;
} a[20];
void main() {
   int n,i,j;
   char nam[10];
   printf("Enter the no. of files: ");
   scanf("%d",&n);
   for(i=0;i<n;i++) {
      printf("\nEnter the name of file %d: ",i+1);
      scanf("%s",a[i].name);
      printf("Enter the starting block of file %s:
",a[i].name);
      scanf("%d",&a[i].start);
      printf("Enter the no. of blocks in file %s:
",a[i].name);
      scanf("%d",&a[i].num);
   }
   printf("\nEnter the name of file to be searched: ");
   scanf("%s",nam);
   for(i=0;i<n;i++)
      if(strcmp(nam,a[i].name)==0)
         break;
   printf("\nFile Name\tStart Block\tNo. of
Blocks\tBlocks Occupied\n");
```

```
printf("%s\t\t%d\t\t%d\t\t",a[i].name,a[i].start,a[i].nu
m);
   for(j=0;j<a[i].num;j++)
      printf("%d  ",a[i].start++);
   printf("\n");
}
```

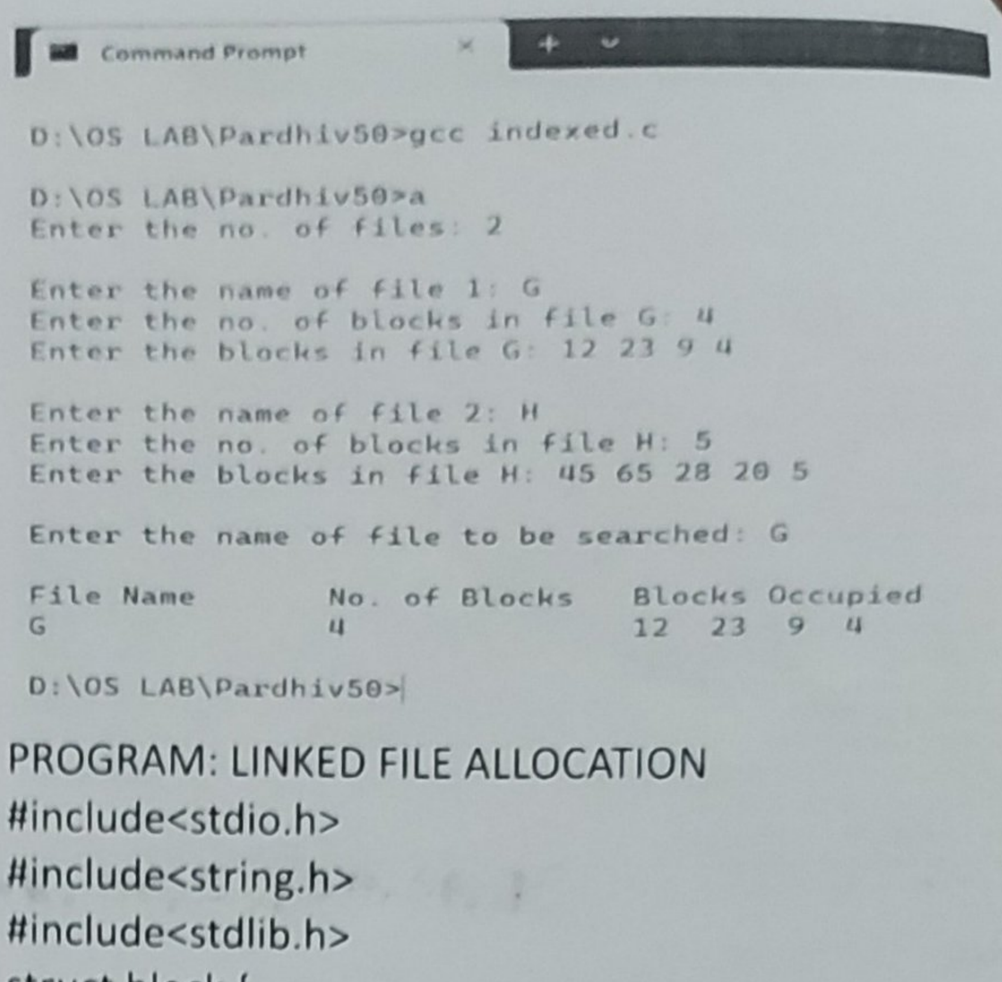
OUTPUT



PROGRAM: INDEXED FILE ALLOCATION

```
#include<stdio.h>
#include<string.h>
struct file {
   char name[10];
   int num,block[20];
} a[20];
void main() {
   int n,i,j;
   char nam[10];
   printf("Enter the no. of files: ");
   scanf("%d",&n);
   for(i=0;i<n;i++) {
      printf("\nEnter the name of file %d: ",i+1);
      scanf("%s",a[i].name);
      printf("Enter the no. of blocks in file %s:
",a[i].name);
      scanf("%d",&a[i].num);
      printf("Enter the blocks in file %s: ",a[i].name);
      for(j=0;j<a[i].num;j++)
         scanf("%d",&a[i].block[j]);
   }
   printf("\nEnter the name of file to be searched: ");
   scanf("%s",nam);
   for(i=0;i<n;i++)
      if(strcmp(nam,a[i].name)==0)
         break;
   printf("\nFile Name\tNo. of Blocks\tBlocks
Occupied\n");
   printf("%s\t\t%d\t\t",a[i].name,a[i].num);
   for(j=0;j<a[i].num;j++)
      printf("%d  ",a[i].block[j]);
   printf("\n");
}
```

OUTPUT



PROGRAM: LINKED FILE ALLOCATION

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
struct block {
   int n;
   struct block *next;
};
struct file {
   char name[10];
   int num;
   struct block *temp,*node;
} a[20];
void main() {

   struct block *head[20];
   int n,i,j;
   char nam[10];
   printf("Enter the no. of files: ");
   scanf("%d",&n);
   for(i=0;i<n;i++) {
      head[i]=NULL;
      printf("\nEnter the name of file %d: ",i+1);
      scanf("%s",a[i].name);
      printf("Enter the no. of blocks in file %s:
",a[i].name);
      scanf("%d",&a[i].num);
      printf("Enter the blocks in file %s: ",a[i].name);
      for(j=0;j<a[i].num;j++) {
         a[i].node=(struct block*)malloc(sizeof(struct
block));
         scanf("%d",&a[i].node->n);
         if(head[i]==NULL) {
            head[i]=a[i].temp=a[i].node;
            a[i].temp->next=NULL;
         }
         else {
            a[i].temp->next=a[i].node;
            a[i].temp=a[i].temp->next;
            a[i].temp->next=NULL;
         }
      }
   }
   printf("\nEnter the name of file to be searched: ");
   scanf("%s",nam);
```

```
    for(i=0;i<n;i++)
      if(strcmp(nam,a[i].name)==0)
        break;
    printf("\nFile Name\tNo. of Blocks\tBlocks
Occupied\n");
    printf("%s\t\t%d\t\t",a[i].name,a[i].num);
    a[i].temp=head[i];
    while(a[i].temp!=NULL) {
      printf("%d -> ",a[i].temp->n);
      a[i].temp=a[i].temp->next;
    }
    printf("\n");
}
```

OUTPUT

```
Command Prompt

D:\OS LAB\Pardhiv50>gcc linked.c

D:\OS LAB\Pardhiv50>a
Enter the no. of files: 3

Enter the name of file 1: A
Enter the no. of blocks in file A: 2
Enter the blocks in file A: 10 14

Enter the name of file 2: B
Enter the no. of blocks in file B: 4
Enter the blocks in file B: 66 55 44 33

Enter the name of file 3: C
Enter the no. of blocks in file C: 2
Enter the blocks in file C: 68 90

Enter the name of file to be searched: B

File Name       No. of Blocks   Blocks Occupied
B               4               66 -> 55 -> 44 -> 33 ->

D:\OS LAB\Pardhiv50>
```