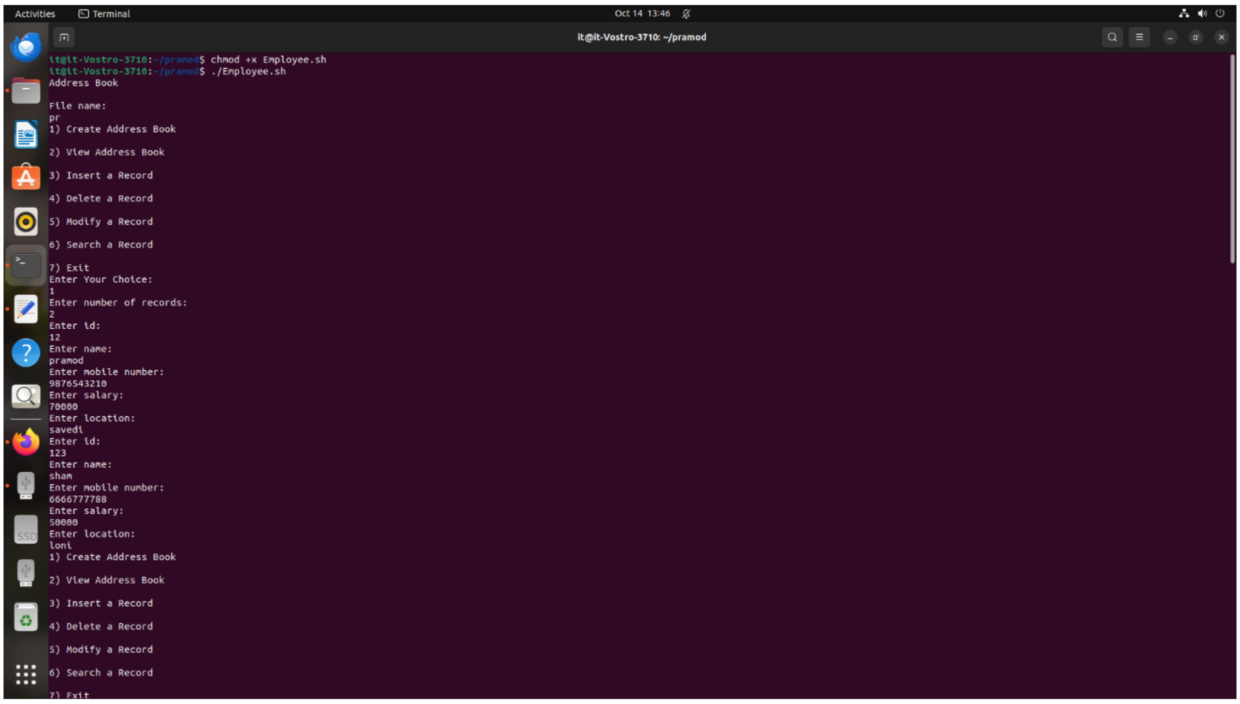
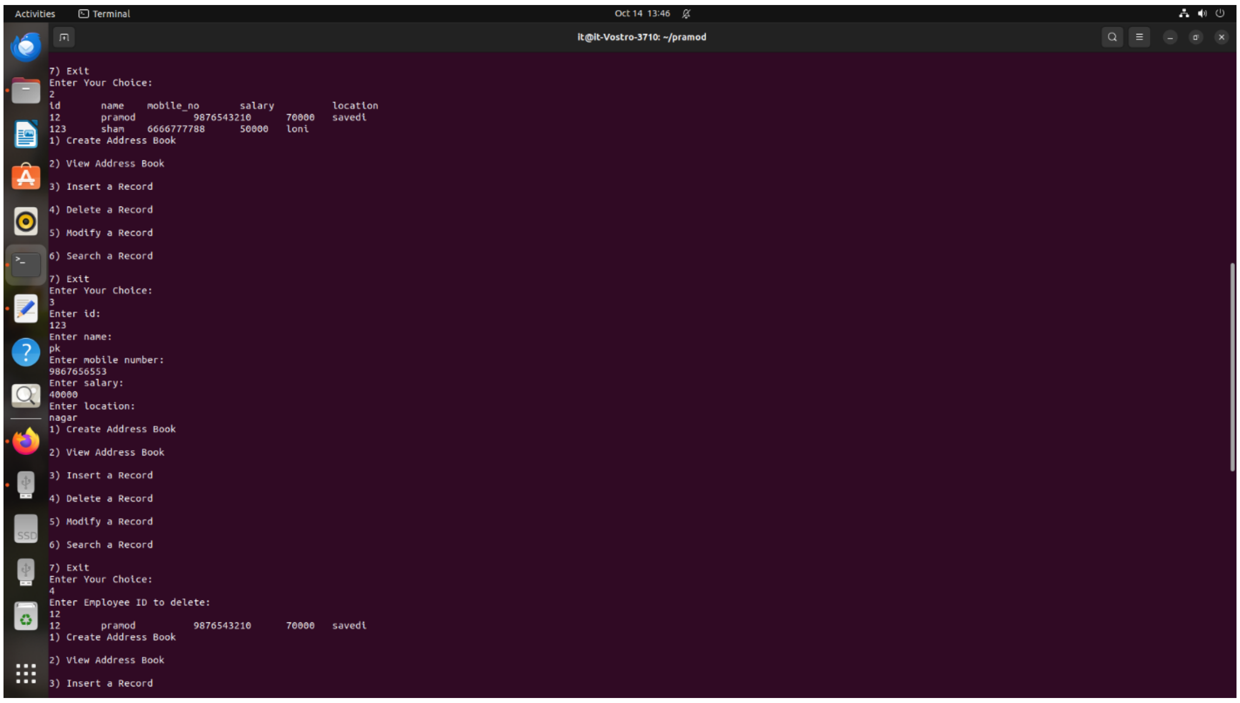
```
echo -e "Address Book\n"

echo -e "File name:"
read fname

touch $fname

echo -e "id \t name \t mobile_no \t salary \t location \t" >> $fname

ch=0
while [ $ch -lt 7 ]
do

echo -e "1) Create Address Book\n"
echo -e "2) View Address Book\n"
echo -e "3) Insert a Record\n"
echo -e "4) Delete a Record\n"
echo -e "5) Modify a Record\n"
echo -e "6) Search a Record\n"
echo -e "7) Exit"

echo "Enter Your Choice:"
read ch

case $ch in

        1)

        echo "Enter number of records:"
        read n


        for((i=0;i<$n;i++))
        do

        echo "Enter id:"
        read id

        echo "Enter name:"
        read name

        echo "Enter mobile number:"
        read mno

        echo "Enter salary:"
        read sal

        echo "Enter location:"
        read loc

        echo -e "$id \t $name \t $mno \t $sal \t $loc" >> $fname

        done
;;
```

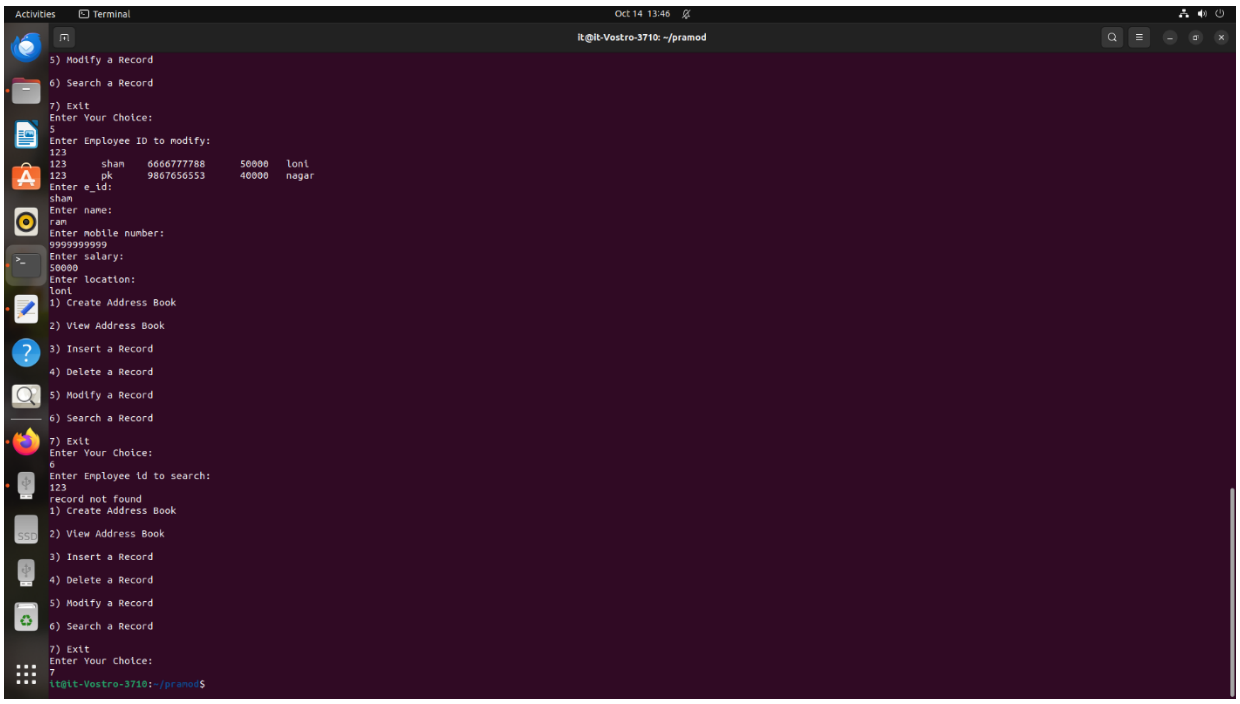
```
        2)

        cat  $fname
;;

        3)

        echo "Enter id:"
        read id

        echo "Enter name:"
        read name

        echo "Enter mobile number:"
        read mno

        echo "Enter salary:"
        read sal

        echo "Enter location:"
        read loc

        echo -e "$id \t $name \t $mno \t $sal \t $loc " >> $fname

;;
        4)

        echo "Enter Employee ID to delete:"
        read id

        if grep -w $id $fname
        then
                grep -wv $id $fname >>temp
                rm $fname
                mv temp $fname
        else
                echo "record not found"
        fi
;;
        5)
        echo "Enter Employee ID to modify:"
        read id

        if grep -w $id $fname
        then
                grep -wv $id $fname >>temp
                rm $fname
                mv temp $fname

                echo "Enter e_id:"
                read id

                echo "Enter name:"
                read name
        echo "Enter mobile number:"
        read mno

        echo "Enter salary:"
        read sal

        echo "Enter location:"
        read loc

        echo -e "$id \t $name \t $mno \t $sal \t $loc " >> $fname
        else
            echo "record not found"
        fi
;;

    6)
        echo "Enter Employee id to search:"
        read id
        if grep -w $id $fname
        then
            echo "Record found"
        else
            echo "record not found"
        fi
;;

    *)

esac

done
```



```
it@it-Vostro-3710:~/pramod$ chmod +x Employee.sh
it@it-Vostro-3710:~/pramod$ ./Employee.sh
Address Book

File name:
pr
1) Create Address Book

2) View Address Book

3) Insert a Record

4) Delete a Record

5) Modify a Record

6) Search a Record

7) Exit
Enter Your Choice:
1
Enter number of records:
2
Enter id:
12
Enter name:
pramod
Enter mobile number:
9876543210
Enter salary:
70000
Enter location:
savedi
Enter id:
123
Enter name:
sham
Enter mobile number:
6666777788
Enter salary:
50000
Enter location:
loni
1) Create Address Book

2) View Address Book

3) Insert a Record

4) Delete a Record

5) Modify a Record

6) Search a Record

7) Exit
```



```
7) Exit
Enter Your Choice:
2
id       name    mobile_no       salary          location
12       pramod          9876543210      70000   savedi
123      sham    6666777788      50000   loni
1) Create Address Book

2) View Address Book

3) Insert a Record

4) Delete a Record

5) Modify a Record

6) Search a Record

7) Exit
Enter Your Choice:
3
Enter id:
123
Enter name:
pk
Enter mobile number:
9867656553
Enter salary:
40000
Enter location:
nagar
1) Create Address Book

2) View Address Book

3) Insert a Record

4) Delete a Record

5) Modify a Record

6) Search a Record

7) Exit
Enter Your Choice:
4
Enter Employee ID to delete:
12
12       pramod          9876543210      70000   savedi
1) Create Address Book

2) View Address Book

3) Insert a Record
```

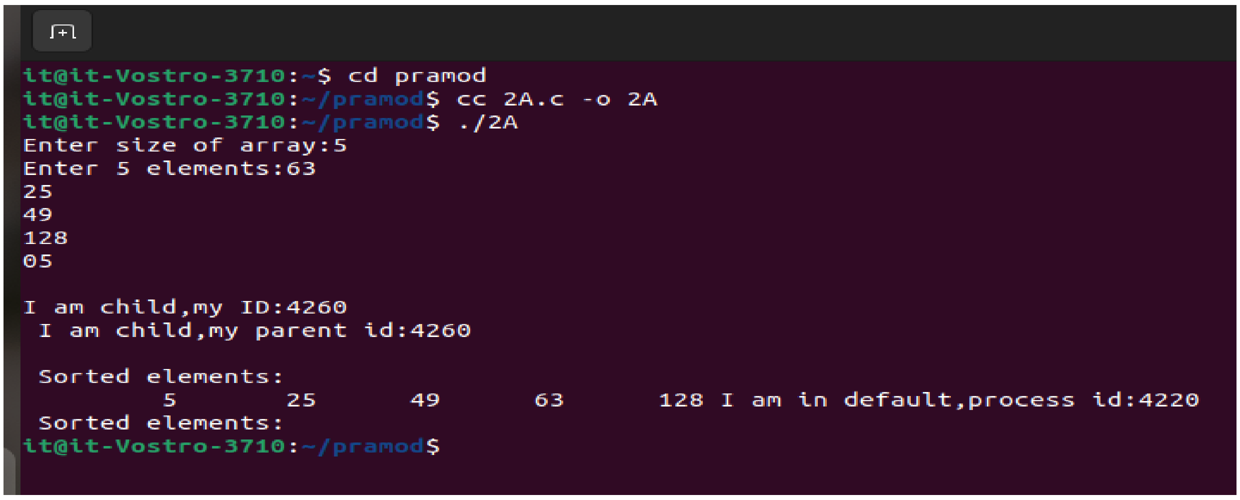


```
5) Modify a Record

6) Search a Record

7) Exit
Enter Your Choice:
5
Enter Employee ID to modify:
123
123      sham    6666777788      50000   loni
123      pk      9867656553      40000   nagar
Enter e_id:
sham
Enter name:
ram
Enter mobile number:
9999999999
Enter salary:
50000
Enter location:
loni
1) Create Address Book

2) View Address Book

3) Insert a Record

4) Delete a Record

5) Modify a Record

6) Search a Record

7) Exit
Enter Your Choice:
6
Enter Employee id to search:
123
record not found
1) Create Address Book

2) View Address Book

3) Insert a Record

4) Delete a Record

5) Modify a Record

6) Search a Record

7) Exit
Enter Your Choice:
7
it@it-Vostro-3710:~/pramod$
```

```
#include<stdio.h>
#include<unistd.h>
#include<sys/types.h>
void quicksort(int a[],int,int);
void merge(int a[],int low,int mid,int high);
void divide(int a[], int low ,int high);
int main()
{
int a[20],n,i;
pid_t pid;
printf("Enter size of array:");
scanf("%d",&n);
printf("Enter %d elements:",n);
for(i=0;i<n;i++)
scanf("%d",&a[i]);
pid=fork();
switch(pid)
{
case 0:
printf("I am child,my ID:%d",getpid());
printf("\n I am child,my parent id:%d\n",getpid());
quicksort(a,0,n-1);
break;
case 1:
printf("The child process has not created");
break;
default:
printf("\n I am in default,process id: %d",getpid());
divide(a,0,n-1);
sleep(3);
break;
```

```
}
printf("\n Sorted elements:\n");
for(i=0;i<n;i++)
printf("\t %d",a[i]);
return 0;
}
void divide(int a[],int low,int high)
{
if(low<high)
{
int mid=(low+high)/2;
divide(a,low,mid);
divide(a,mid+1,high);
merge(a,low,mid,high);
}
}

void merge(int a[],int low,int mid,int high)
{
int i,j,k,m=mid-low+1,n=high-mid;
int first_half[m],second_half[n];
for(i=0;i<m;i++)
first_half[i]=a[low+i];
for(i=0;i<n;i++)
second_half[i]=a[mid+i+1];
i=j=0;
k=low;
while(i<m || j<n)
{
if(i>=m)
{
```

```
a[k++]=second_half[j++];
continue;
}
if(j>=n)
{
a[k++]=first_half[i++];
continue;
}
if(first_half[i]<second_half[j])
a[k++]=first_half[i++];
else
a[k++]=second_half[j++];
}
}
void quicksort(int a[],int first,int last)
{
int pivot,j,temp,i;
if(first<last)
{
pivot=first;
i=first;j=last;while(i<j)
{
while(a[i]<=a[pivot]&&i<last)
i++;
while(a[j]>a[pivot])
j--;
if(i<j)
{
temp=a[i];
a[i]=a[j];
a[j]=temp;
}
}
temp=a[pivot];
a[pivot]=a[j];
a[j]=temp;
quicksort(a,first,j-1);
quicksort(a,j+1,last);
}
```

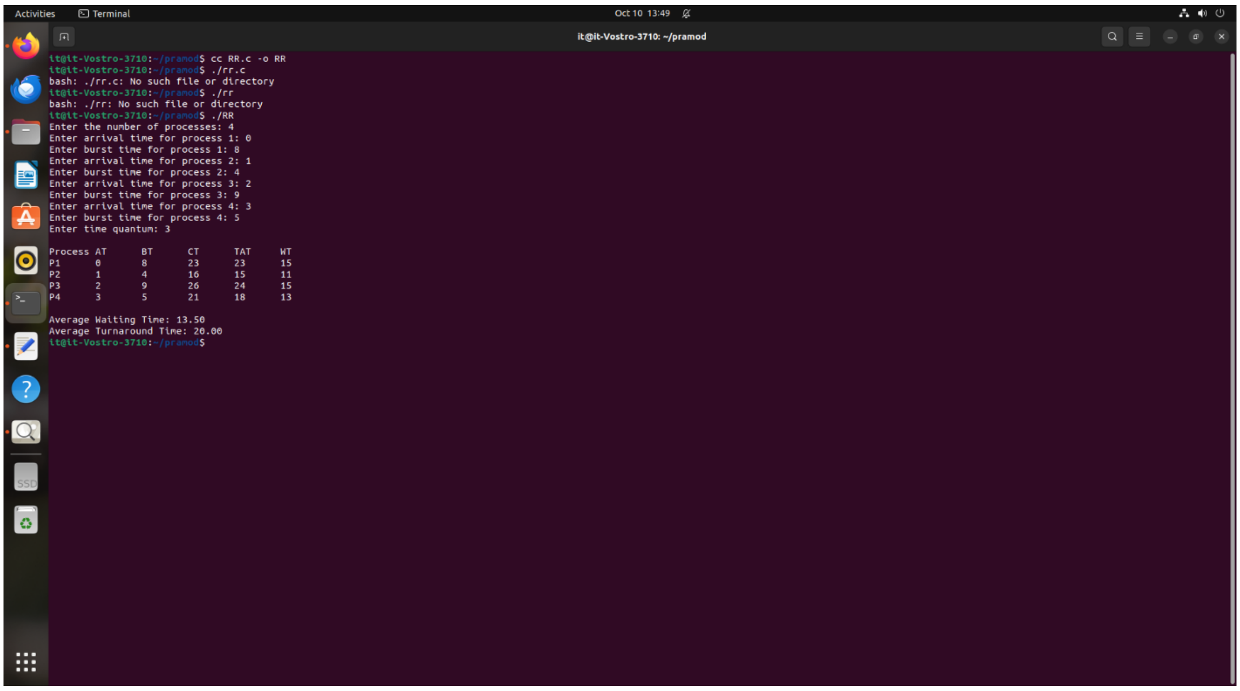


```
it@it-Vostro-3710:~$ cd pramod
it@it-Vostro-3710:~/pramod$ cc 2A.c -o 2A
it@it-Vostro-3710:~/pramod$ ./2A
Enter size of array:5
Enter 5 elements:63
25
49
128
05

I am child,my ID:4260
 I am child,my parent id:4260

 Sorted elements:
         5       25      49      63      128 I am in default,process id:4220
 Sorted elements:
it@it-Vostro-3710:~/pramod$
```

```
#include <stdio.h>
#include <stdlib.h>
#define N 100
struct process {
    int process_id;
    int arrival_time;
    int brust_time;
    int completion_time;
    int waiting_time;
    int turn_around_time;
    int remaining_time;
    int started; // flag to mark if process has started execution
};
struct process proc[N];
int queue[N];
int front = 0, rear = 0;
void push(int process_id) {
    queue[rear] = process_id;
    rear = (rear + 1) % N;
}int pop() {
    if (front == rear)
        return -1;
    int ret = queue[front];
    front = (front + 1) % N;
    return ret;
}int is_in_queue(int process_id) {
    for (int i = front; i != rear; i = (i + 1) % N) {
        if (queue[i] == process_id)
            return 1;
    }  return 0;
}
```

```
int main() {
    int n, time_quantum;
    float total_waiting_time = 0, total_turnaround_time = 0;
    printf("Enter the number of processes: ");
    scanf("%d", &n);
    for (int i = 0; i < n; i++) {
        printf("Enter arrival time for process %d: ", i + 1);
        scanf("%d", &proc[i].arrival_time);
        printf("Enter burst time for process %d: ", i + 1);
        scanf("%d", &proc[i].brust_time);
        proc[i].process_id = i + 1;
        proc[i].remaining_time = proc[i].brust_time;
        proc[i].started = 0;
    }   printf("Enter time quantum: ");
    scanf("%d", &time_quantum);
    int time = 0;
    int completed = 0;
    int current = -1;
    int time_slice = 0;
    // Enqueue all processes that arrive at time 0
    for (int i = 0; i < n; i++) {
        if (proc[i].arrival_time == time) {
            push(i);
            proc[i].started = 1;
        }
    } while (completed < n) {
        if (current == -1) {
            current = pop();
            time_slice = 0;
        }
        if (current != -1) {
```

```
        proc[current].remaining_time--;
        time_slice++;
        time++;
        // Enqueue new arrivals at current time
        for (int i = 0; i < n; i++) {
            if (proc[i].arrival_time == time && !proc[i].started) {
                push(i);
                proc[i].started = 1;
            }
        }
        if (proc[current].remaining_time == 0) {
            proc[current].completion_time = time;
            proc[current].turn_around_time = proc[current].completion_time - proc[current].arrival_time;
            proc[current].waiting_time = proc[current].turn_around_time - proc[current].brust_time;
            total_turnaround_time += proc[current].turn_around_time;
            total_waiting_time += proc[current].waiting_time;
            completed++;
            current = -1;
            time_slice = 0;
        } else if (time_slice == time_quantum) {
            push(current);
            current = -1;
        }
    } else {
        // CPU is idle, move time forward
        time++;
        for (int i = 0; i < n; i++) {
            if (proc[i].arrival_time == time && !proc[i].started) {
                push(i);
                proc[i].started = 1;
            }
        }
      }
  } printf("\nProcess\tAT\tBT\tCT\tTAT\tWT\n");
  for (int i = 0; i < n; i++) {
     printf("P%d\t%d\t%d\t%d\t%d\t%d\n",
          proc[i].process_id,
          proc[i].arrival_time,
          proc[i].brust_time,
          proc[i].completion_time,
          proc[i].turn_around_time,
          proc[i].waiting_time);
  }   printf("\nAverage Waiting Time: %.2f\n", total_waiting_time / n);
  printf("Average Turnaround Time: %.2f\n", total_turnaround_time / n);
  return 0;
}
```

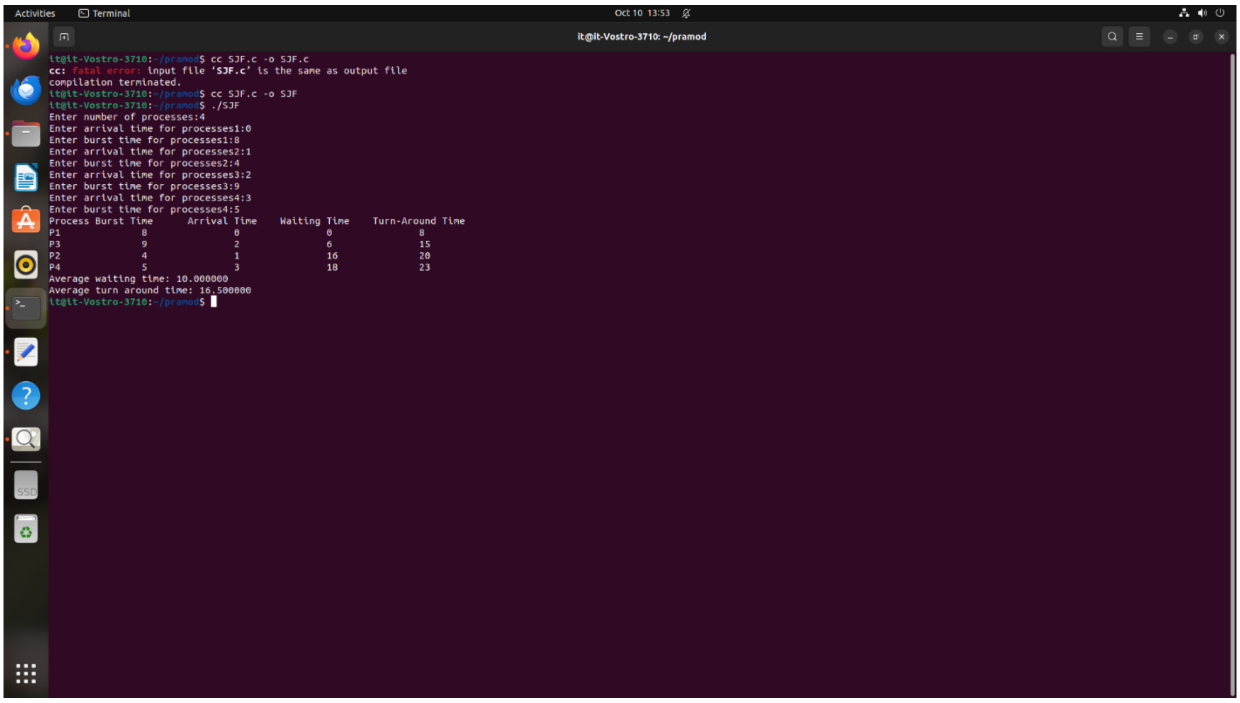


```
it@it-Vostro-3710:~/pramod$ cc RR.c -o RR
it@it-Vostro-3710:~/pramod$ ./rr.c
bash: ./rr.c: No such file or directory
it@it-Vostro-3710:~/pramod$ ./rr
bash: ./rr: No such file or directory
it@it-Vostro-3710:~/pramod$ ./RR
Enter the number of processes: 4
Enter arrival time for process 1: 0
Enter burst time for process 1: 8
Enter arrival time for process 2: 1
Enter burst time for process 2: 4
Enter arrival time for process 3: 2
Enter burst time for process 3: 9
Enter arrival time for process 4: 3
Enter burst time for process 4: 5
Enter time quantum: 3

Process AT      BT      CT      TAT     WT
P1      0       8       23      23      15
P2      1       4       16      15      11
P3      2       9       26      24      15
P4      3       5       21      18      13

Average Waiting Time: 13.50
Average Turnaround Time: 20.00
it@it-Vostro-3710:~/pramod$
```

```
#include<stdio.h>
struct Process
{
int id;
int arrivalTime;
int burstTime;
int waitingTime;
int turnAroundTime;
};
void calculateTimes(struct Process proc[],int n)
{
int totalWaitingTime=0,totalTurnAroundTime=0;
int completionTime[n];
for(int i=0;i<n-1;i++)
{
for(int j=+1;j<n;j++)
{
if(proc[i].arrivalTime>proc[j].arrivalTime||
(proc[i].arrivalTime==proc[j].arrivalTime &&
proc[i].burstTime>proc[j].burstTime))
{
struct Process temp=proc[i];
proc[i]=proc[j];
proc[j]=temp;
}
}
}
completionTime[0]=proc[0].arrivalTime + proc[0].burstTime;
proc[0].turnAroundTime = proc[0].burstTime;
proc[0].waitingTime=0;
for(int i=1;i<n;i++)
```

```
{
completionTime[i]=completionTime[i-1]+proc[i].burstTime;
proc[i].turnAroundTime=completionTime[i]-proc[i].arrivalTime;
proc[i].waitingTime=proc[i].turnAroundTime-proc[i].burstTime;
}
printf("Process\tBurst Time\tArrival Time\tWaiting Time\tTurn-Around Time\n");
for(int i=0;i<n;i++)
{
printf("P%d\t\t%d\t\t%d\t\t%d\t\t%d\n",proc[i].id,proc[i].burstTime,proc[i].arrivalTime,proc[i].waitingT
ime,proc[i].turnAroundTime);
totalWaitingTime+=proc[i].waitingTime;
totalTurnAroundTime+=proc[i].turnAroundTime;
}
printf("Average waiting time: %2f\n",(float)totalWaitingTime/n);
printf("Average turn around time: %2f\n",(float)totalTurnAroundTime/n);
}
int main()
{
int n;
printf("Enter number of processes:");
scanf("%d",&n);
struct Process proc[n];
for(int i=0;i<n;i++)
{
proc[i].id=i+1;
printf("Enter arrival time for processes%d:",proc[i].id);
scanf("%d",&proc[i].arrivalTime);
printf("Enter burst time for processes%d:",proc[i].id);
scanf("%d",&proc[i].burstTime);
}
calculateTimes(proc,n);
```

```
return 0;

}
```

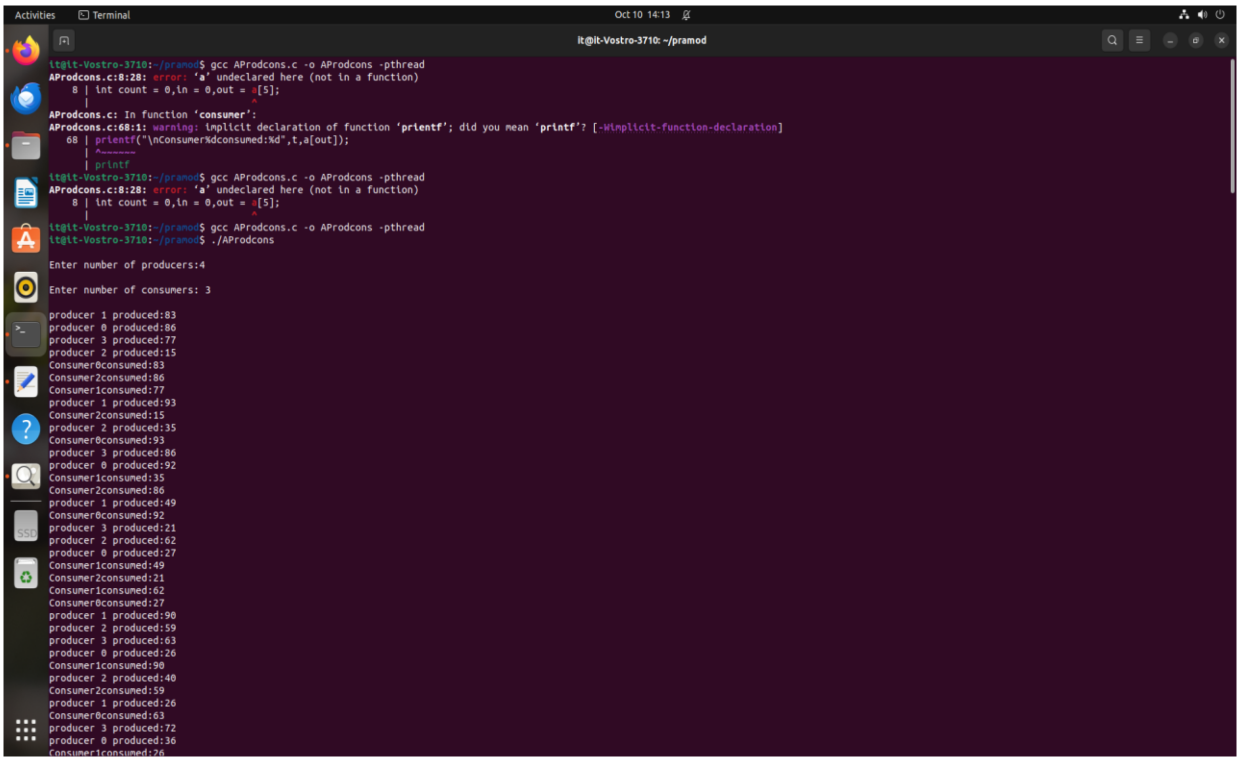


```
it@it-Vostro-3710:~/pramod$ cc SJF.c -o SJF.c
cc: fatal error: input file 'SJF.c' is the same as output file
compilation terminated.
it@it-Vostro-3710:~/pramod$ cc SJF.c -o SJF
it@it-Vostro-3710:~/pramod$ ./SJF
Enter number of processes:4
Enter arrival time for processes1:0
Enter burst time for processes1:8
Enter arrival time for processes2:1
Enter burst time for processes2:4
Enter arrival time for processes3:2
Enter burst time for processes3:9
Enter arrival time for processes4:3
Enter burst time for processes4:5
Process Burst Time      Arrival Time    Waiting Time    Turn-Around Time
P1              8               0               0               8
P3              9               2               6               15
P2              4               1               16              20
P4              5               3               18              23
Average waiting time: 10.000000
Average turn around time: 16.500000
it@it-Vostro-3710:~/pramod$ 
```

```
#include <stdio.h>
#include <pthread.h>
#include <semaphore.h>
#include <unistd.h>
#include <stdlib.h>
void *producer(void *thread);
void *consumer(void *thread);
int count = 0,in = 0,out =0,a[5];
sem_t full;
sem_t empty;
pthread_mutex_t mutex;
int main() {
int  i,p,c;
pthread_t pid[10], cid[10];
pthread_mutex_init(&mutex, NULL);
sem_init(&full,0, 0);
sem_init(&empty,0,5);
printf("\nEnter number of producers:");
scanf("%d", &p);
printf("\nEnter number of consumers: ");
scanf("%d", &c);
int producer_indices[p];
int consumer_indices[c];
for(i=0;i<p;i++) {
producer_indices[i]=i;
pthread_create(&pid[i],NULL,producer,&producer_indices[i]);
}for(i=0;i<c;i++) {
consumer_indices[i]=i;
pthread_create(&cid[i],NULL,consumer,&consumer_indices[i]);
}for(i=0;i<p;i++) {
pthread_join(pid[i],NULL);
```

```
}
for(i=0;i<c;i++){
pthread_join(cid[i],NULL);
}
sem_destroy(&full);
sem_destroy(&empty);
pthread_mutex_destroy(&mutex);
return 0;
}
void*producer(void*thread) {
int t = *(int *)thread;
while(1) {
sem_wait(&empty);
pthread_mutex_lock(&mutex);
if(count>=5) {
printf("\nBuffer is full");
} else {
a[in]=rand()%100;
printf("\nproducer %d produced:%d",t, a[in]);
in=(in+1) % 5;
count++;
}
pthread_mutex_unlock(&mutex);
sem_post(&full);
sleep(1);
}
pthread_exit(0);
}
void*consumer(void*thread) {
int t = *(int *)thread;
while(1) {
```

```
sem_wait(&full);
pthread_mutex_lock(&mutex);
if(count<=0) {
printf("\nBuffer is empty");
}else{
printf("\nConsumer%dconsumed:%d",t,a[out]);
out=(out + 1) %5;
count--;
}pthread_mutex_unlock(&mutex);
sem_post(&empty);
sleep(1);
}
pthread_exit(0);
}
```

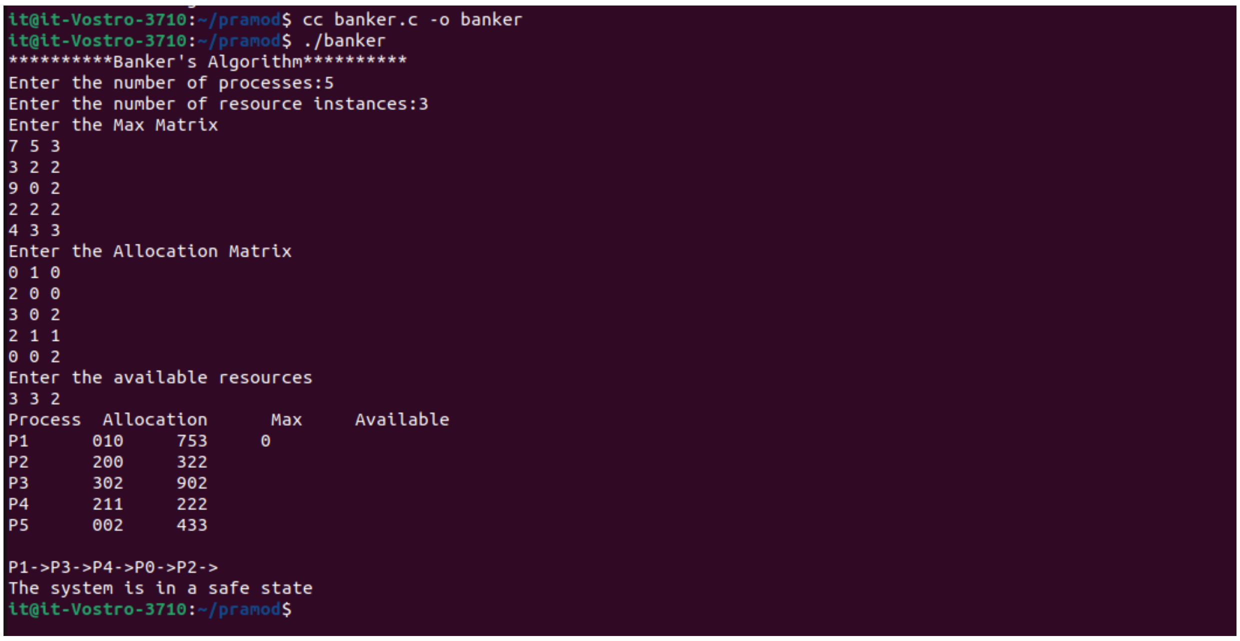


```
it@it-Vostro-3710:~/pramod$ gcc AProdcons.c -o AProdcons -pthread
AProdcons.c:8:28: error: ‘a’ undeclared here (not in a function)
    8 | int count = 0,in = 0,out = a[5];
      |                            ^
AProdcons.c: In function ‘consumer’:
AProdcons.c:68:1: warning: implicit declaration of function ‘prientf’; did you mean ‘printf’? [-Wimplicit-function-declaration]
   68 | prientf("\nConsumer%dconsumed:%d",t,a[out]);
      | ^~~~~~~
      | printf
it@it-Vostro-3710:~/pramod$ gcc AProdcons.c -o AProdcons -pthread
AProdcons.c:8:28: error: ‘a’ undeclared here (not in a function)
    8 | int count = 0,in = 0,out = a[5];
      |                            ^
it@it-Vostro-3710:~/pramod$ gcc AProdcons.c -o AProdcons -pthread
it@it-Vostro-3710:~/pramod$ ./AProdcons

Enter number of producers:4

Enter number of consumers: 3

producer 1 produced:83
producer 0 produced:86
producer 3 produced:77
producer 2 produced:15
Consumer0consumed:83
Consumer2consumed:86
Consumer1consumed:77
producer 1 produced:93
Consumer2consumed:15
producer 2 produced:35
Consumer0consumed:93
producer 3 produced:86
producer 0 produced:92
Consumer1consumed:35
Consumer2consumed:86
producer 1 produced:49
Consumer0consumed:92
producer 3 produced:21
producer 2 produced:62
producer 0 produced:27
Consumer1consumed:49
Consumer2consumed:21
Consumer1consumed:62
Consumer0consumed:27
producer 1 produced:90
producer 2 produced:59
producer 3 produced:63
producer 0 produced:26
Consumer1consumed:90
producer 2 produced:40
Consumer2consumed:59
producer 1 produced:26
Consumer0consumed:63
producer 3 produced:72
producer 0 produced:36
Consumer1consumed:26
```

```
#include <stdio.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(){
printf("**********Banker's Algorithm**********\n");
input();
show();
cal();
getchar();
return 0;
}
void input(){
int i,j;
printf("Enter the number of processes:");
scanf("%d",&n);
printf("Enter the number 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("Enter the Allocation Matrix\n");
for(i=0;i<n;i++){
```

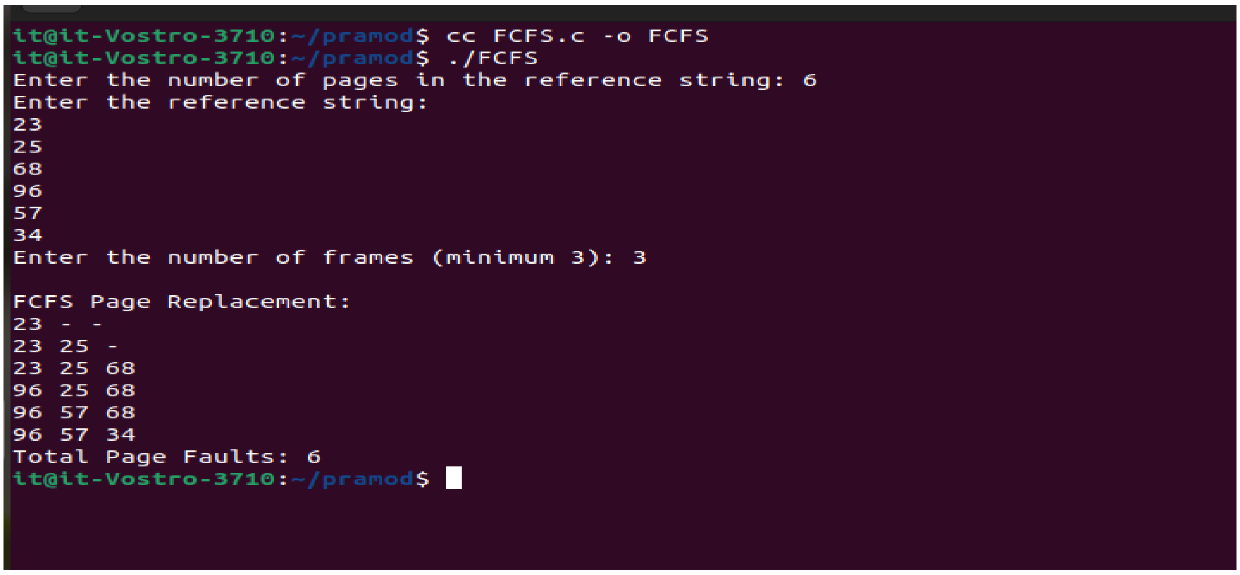
```
for(j=0;j<r;j++){
scanf("%d",&alloc[i][j]);
}
}
printf("Enter the available resources\n");
for(j=0;j<r;j++){
scanf("%d",&avail[j]);
}
}
void show(){
int i,j;
printf("Process\t Allocation\t Max\t Available\n");
for(i=0;i<n;i++){
printf("P%d\t",i+1);
for(j=0;j<r;j++){
printf("%d",alloc[i][j]);
}
printf("\t");
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]);
}
}
printf("\n");
}}
void cal(){
int finish[100],temp,flag=1,k,cl=0;
int safe[100];
int 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];
}
}
printf("\n");
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][k];
}
finish[i]=1;
flag=1;
printf("P%d->",i);
}
}
}
}
for(i=0;i<n;i++){
```

```
if(finish[i]==1){
cl++;
}else{
printf("P%d->",i);
}
}
if(cl==n){
printf("\nThe system is in a safe state\n");
}else{
printf("\nProcesses are in deadlock\n");
printf("\nSystem is in an unsafe state\n");
}
}
```



```
it@it-Vostro-3710:~/pramod$ cc banker.c -o banker
it@it-Vostro-3710:~/pramod$ ./banker
**********Banker's Algorithm**********
Enter the number of processes:5
Enter the number of resource instances:3
Enter the Max Matrix
7 5 3
3 2 2
9 0 2
2 2 2
4 3 3
Enter the Allocation Matrix
0 1 0
2 0 0
3 0 2
2 1 1
0 0 2
Enter the available resources
3 3 2
Process  Allocation      Max     Available
P1      010     753     0
P2      200     322
P3      302     902
P4      211     222
P5      002     433

P1->P3->P4->P0->P2->
The system is in a safe state
it@it-Vostro-3710:~/pramod$
```

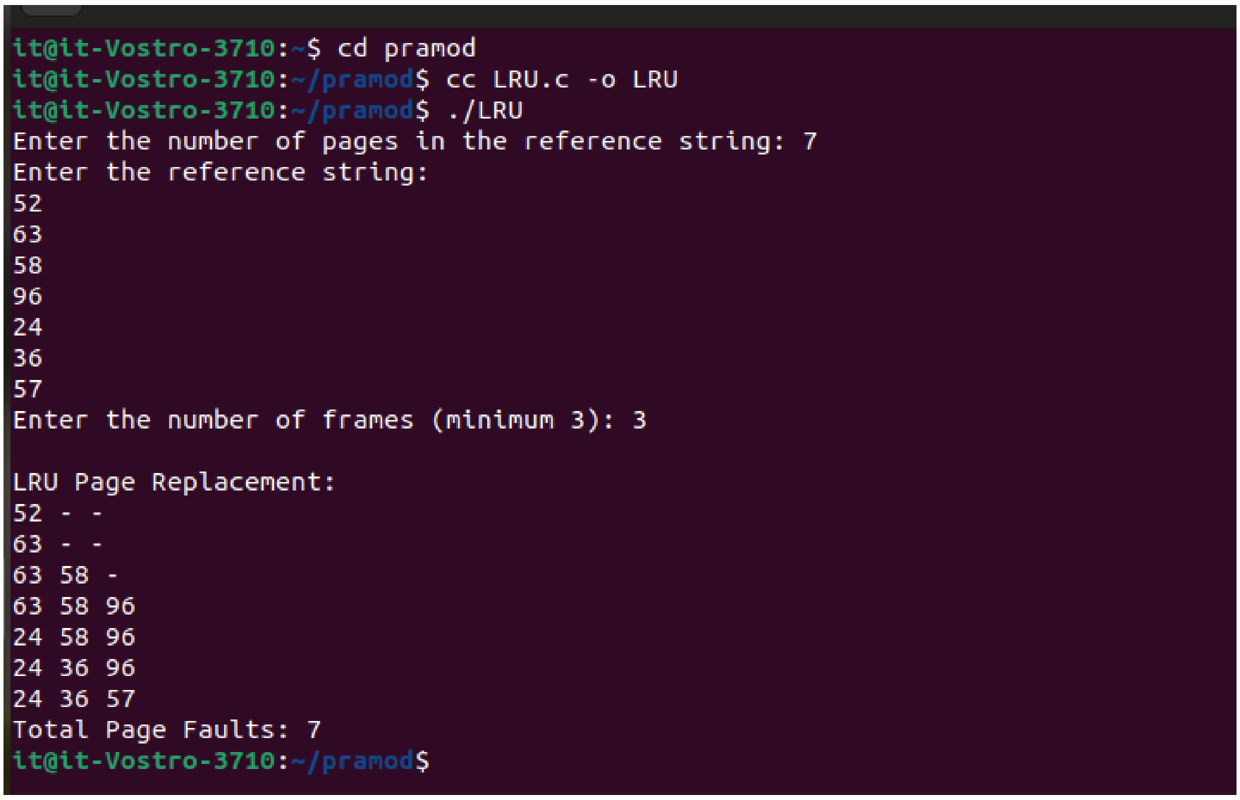
```
#include <stdio.h>
#include <stdlib.h>
void printFrames(int frames[], int frameSize) {
for (int i = 0; i < frameSize; i++) {
if (frames[i] == -1)
printf("- ");
else
printf("%d ", frames[i]);
}printf("\n");
}
void fcfs(int refString[], int refSize, int frameSize) {
int *frames = (int *)malloc(frameSize * sizeof(int));
for (int i = 0; i < frameSize; i++) frames[i] = -1;
int pageFaults = 0, nextReplace = 0;
printf("\nFCFS Page Replacement:\n");
for (int i = 0; i < refSize; i++) {
int found = 0;
for (int j = 0; j < frameSize; j++) {
if (frames[j] == refString[i]) {
found = 1;
break;
}
}
if (!found) {
frames[nextReplace] = refString[i];
nextReplace = (nextReplace + 1) % frameSize;
pageFaults++;
}
printFrames(frames, frameSize);
}
printf("Total Page Faults: %d\n", pageFaults);
free(frames);
}
int main() {
int refSize, frameSize;
printf("Enter the number of pages in the reference string: ");
scanf("%d", &refSize);
int *refString = (int *)malloc(refSize * sizeof(int));
printf("Enter the reference string:\n");
for (int i = 0; i < refSize; i++) {
scanf("%d", &refString[i]);
}
printf("Enter the number of frames (minimum 3): ");
scanf("%d", &frameSize);
if (frameSize < 3) {
printf("Frame size should be at least 3.\n");
free(refString);
return 1;
}
fcfs(refString, refSize, frameSize);
free(refString);return 0;
}
```



```
it@it-Vostro-3710:~/pramod$ cc FCFS.c -o FCFS
it@it-Vostro-3710:~/pramod$ ./FCFS
Enter the number of pages in the reference string: 6
Enter the reference string:
23
25
68
96
57
34
Enter the number of frames (minimum 3): 3

FCFS Page Replacement:
23 - -
23 25 -
23 25 68
96 25 68
96 57 68
96 57 34
Total Page Faults: 6
it@it-Vostro-3710:~/pramod$ 
```

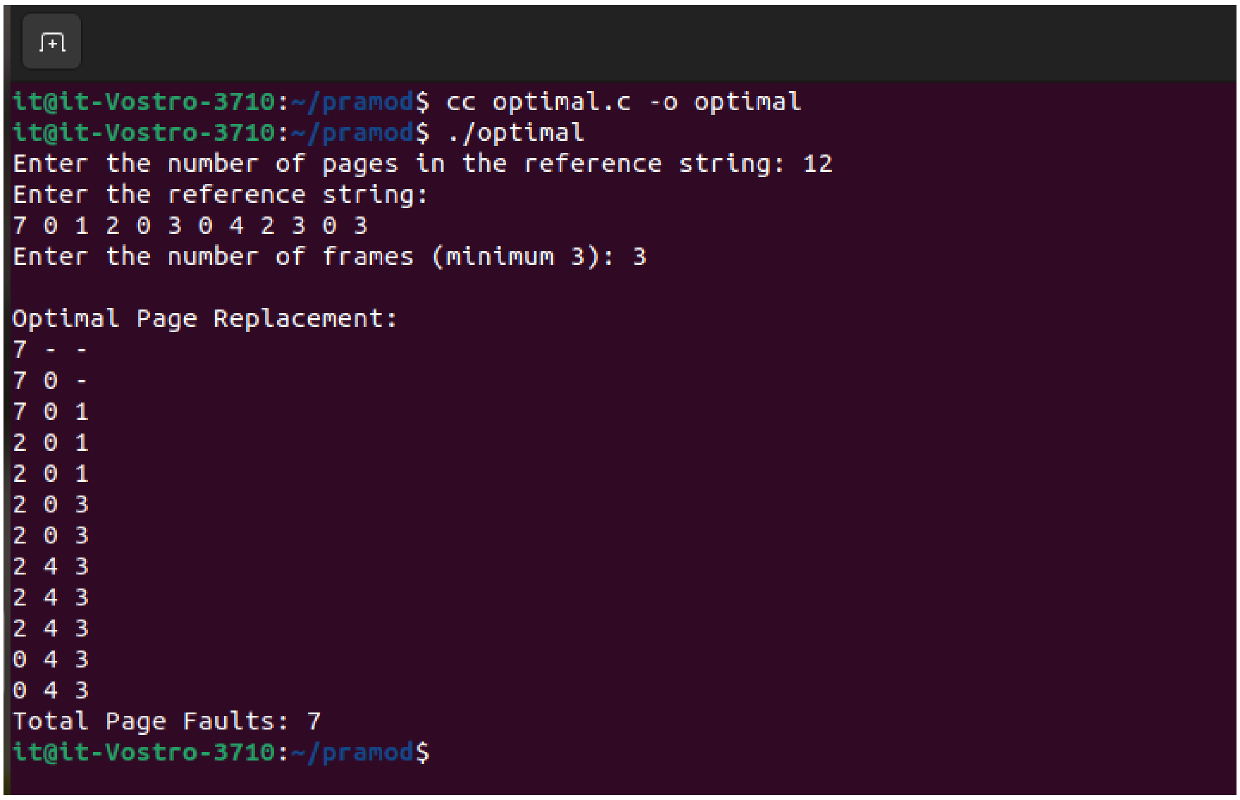
```
#include <stdio.h>
#include <stdlib.h>
void printFrames(int frames[], int frameSize) {
for (int i = 0; i < frameSize; i++) {
if (frames[i] == -1)
printf("- ");
else
printf("%d ", frames[i]);
}
printf("\n");
}
void lru(int refString[], int refSize, int frameSize) {
int *frames = (int *)malloc(frameSize * sizeof(int));
int *time = (int *)malloc(frameSize * sizeof(int));
for (int i = 0; i < frameSize; i++) {
frames[i] = -1;
time[i] = 0;
}
int pageFaults = 0;
printf("\nLRU Page Replacement:\n");
for (int i = 0; i < refSize; i++) {
int found = 0;
for (int j = 0; j < frameSize; j++) {
if (frames[j] == refString[i]) {
found = 1;
time[j] = i;
break;
}
}
if (!found) {
int lruIndex = 0;
for (int j = 1; j < frameSize; j++) {
if (time[j] < time[lruIndex])
lruIndex = j;
}
frames[lruIndex] = refString[i];
time[lruIndex] = i;
pageFaults++;
}
printFrames(frames, frameSize);
}
printf("Total Page Faults: %d\n", pageFaults);
free(frames);
free(time);
}
int main() {
int refSize, frameSize;
printf("Enter the number of pages in the reference string: ");
scanf("%d", &refSize);
int *refString = (int *)malloc(refSize * sizeof(int));
printf("Enter the reference string:\n");
for (int i = 0; i < refSize; i++) {
scanf("%d", &refString[i]);
}
printf("Enter the number of frames (minimum 3): ");
scanf("%d", &frameSize);
if (frameSize < 3) {
printf("Frame size should be at least 3.\n");
free(refString);
return 1;
}lru(refString, refSize, frameSize);
free(refString);
```

```
return 0;

}
```



```
it@it-Vostro-3710:~$ cd pramod
it@it-Vostro-3710:~/pramod$ cc LRU.c -o LRU
it@it-Vostro-3710:~/pramod$ ./LRU
Enter the number of pages in the reference string: 7
Enter the reference string:
52
63
58
96
24
36
57
Enter the number of frames (minimum 3): 3

LRU Page Replacement:
52 - -
63 - -
63 58 -
63 58 96
24 58 96
24 36 96
24 36 57
Total Page Faults: 7
it@it-Vostro-3710:~/pramod$
```

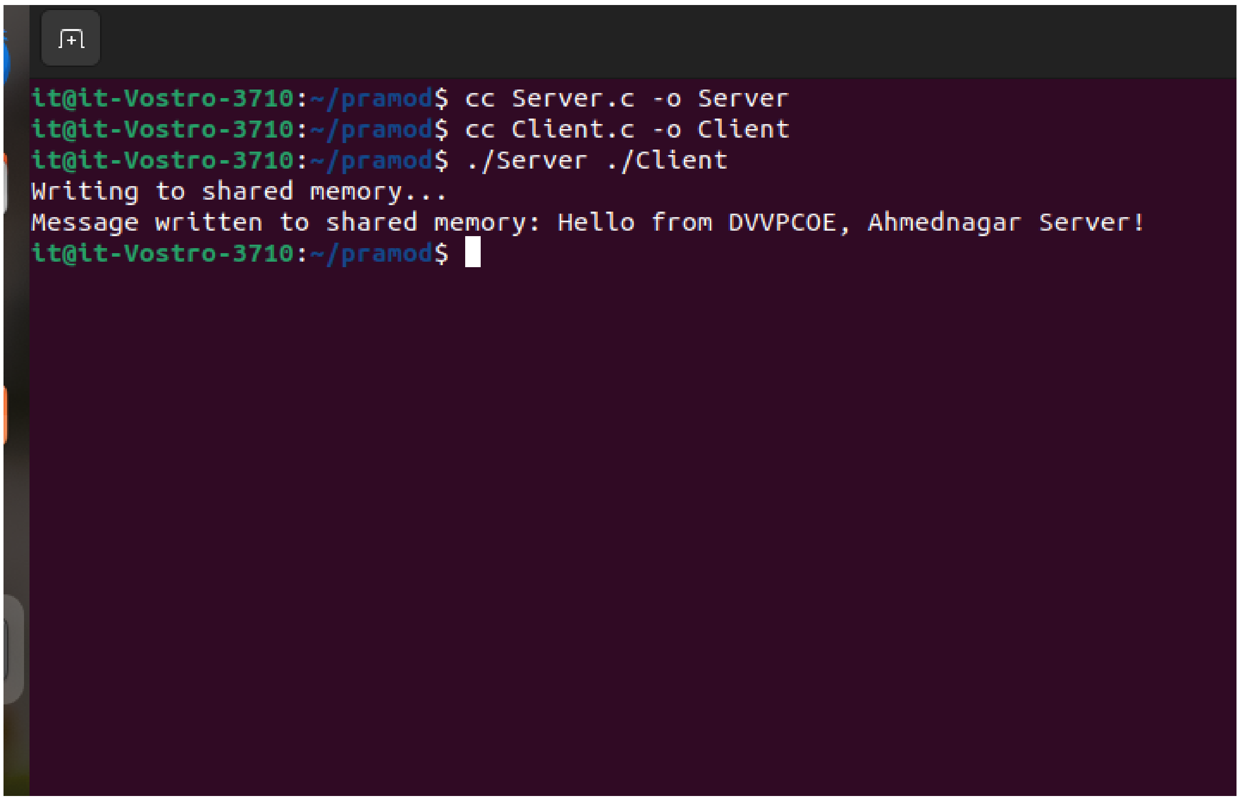
```
#include <stdio.h>

void printFrames(int frames[], int frameSize) {

    for (int i = 0; i < frameSize; i++) {

        if (frames[i] == -1)

            printf("- ");

        else

            printf("%d ", frames[i]);

    }

    printf("\n");

}


int findOptimal(int frames[], int frameSize, int refString[], int refSize, int currentIndex) {

    int farthest = currentIndex;

    int index = -1;
    for (int i = 0; i < frameSize; i++) {
        int j;
        for (j = currentIndex; j < refSize; j++) {
            if (frames[i] == refString[j]) {
                if (j > farthest) {
                    farthest = j;
                    index = i;
                }
                break;
            }
        }
        if (j == refSize) return i; // If not found in future, replace this
    }
    return (index == -1 ? 0 : index);
}

void optimal(int refString[], int refSize, int frameSize) {
    int frames[frameSize];
    for (int i = 0; i < frameSize; i++) frames[i] = -1;

    int pageFaults = 0;
    printf("\nOptimal Page Replacement:\n");

    for (int i = 0; i < refSize; i++) {
        int found = 0;
        for (int j = 0; j < frameSize; j++) {
            if (frames[j] == refString[i]) {
                found = 1;
                break;
            }
        }
        if (!found) {
            int replaceIndex = (i < frameSize) ? i : findOptimal(frames, frameSize, refString, refSize, i + 1);
            frames[replaceIndex] = refString[i];
            pageFaults++;
        }
        printFrames(frames, frameSize);
    }

    printf("Total Page Faults: %d\n", pageFaults);
}

int main() {
    int refSize, frameSize;

    printf("Enter the number of pages in the reference string: ");
    scanf("%d", &refSize);

    int refString[refSize];
    printf("Enter the reference string:\n");
    for (int i = 0; i < refSize; i++) {
        scanf("%d", &refString[i]);
    }
    printf("Enter the number of frames (minimum 3): ");
    scanf("%d", &frameSize);
    if (frameSize < 3) {
        printf("Frame size should be at least 3.\n");
        return 1;
    }
    optimal(refString, refSize, frameSize);
    return 0;
}
```



```
it@it-Vostro-3710:~/pramod$ cc optimal.c -o optimal
it@it-Vostro-3710:~/pramod$ ./optimal
Enter the number of pages in the reference string: 12
Enter the reference string:
7 0 1 2 0 3 0 4 2 3 0 3
Enter the number of frames (minimum 3): 3

Optimal Page Replacement:
7 - -
7 0 -
7 0 1
2 0 1
2 0 1
2 0 3
2 0 3
2 4 3
2 4 3
2 4 3
0 4 3
0 4 3
Total Page Faults: 7
it@it-Vostro-3710:~/pramod$
```

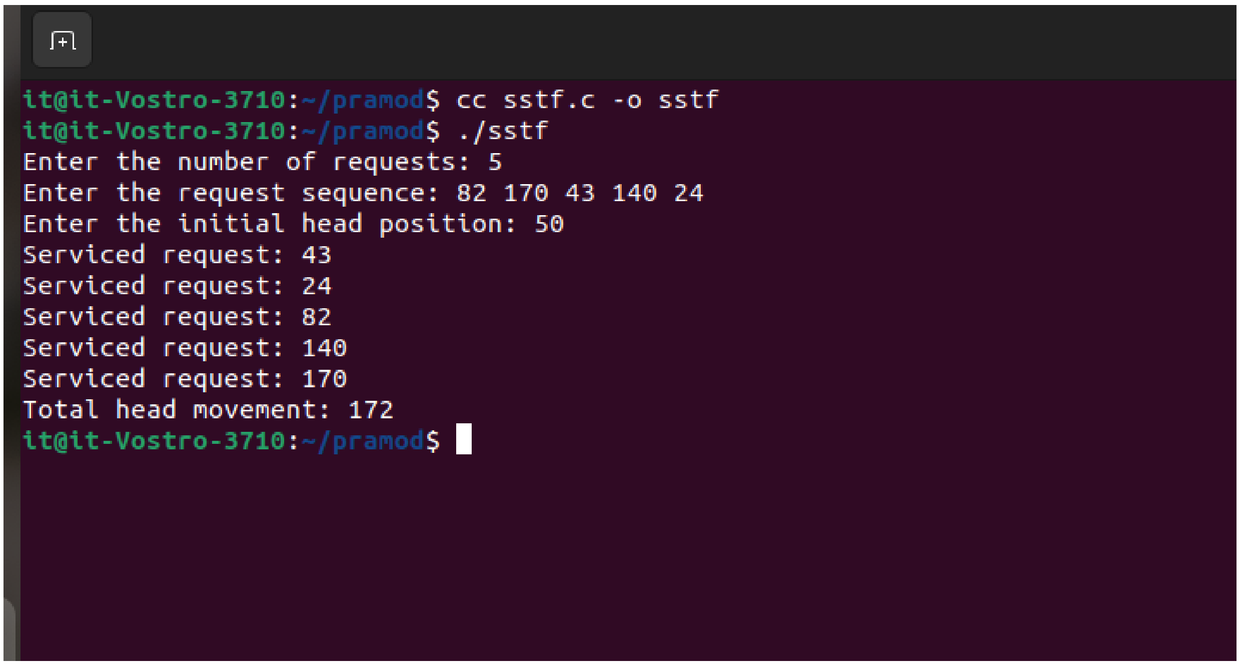
```
#include <stdio.h>
#include <stdlib.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#define SHM_KEY 12345
#define SHM_SIZE 1024
int main() {
int shmid;
char *shmaddr;
shmid = shmget(SHM_KEY, SHM_SIZE, 0666);
if (shmid < 0) {
perror("shmget");
exit(1);
}
shmaddr = shmat(shmid, NULL, 0);
if (shmaddr == (char *) -1) {
perror("shmat");
exit(1);
}
printf("Reading from shared memory...\n");
printf("Message from shared memory: %s\n", shmaddr);
if (shmdt(shmaddr) == -1) {
perror("shmdt");
exit(1);
}
return 0;
}
```

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <string.h>

#define SHM_KEY 12345
#define SHM_SIZE 1024

int main() {
int shmid;
char *shmaddr;

shmid = shmget(SHM_KEY, SHM_SIZE, IPC_CREAT | 0666);
if (shmid < 0) {
perror("shmget");
exit(1);
}

shmaddr = shmat(shmid, NULL, 0);
if (shmaddr == (char *) -1) {
perror("shmat");
exit(1);
}

printf("Writing to shared memory...\n");
char *message = "Hello from DVVPCOE, Ahmednagar Server!";
strncpy(shmaddr, message, SHM_SIZE);

if (shmdt(shmaddr) == -1) {
perror("shmdt");
exit(1);
}

printf("Message written to shared memory: %s\n", message);

return 0;
}
```



```
it@it-Vostro-3710:~/pramod$ cc Server.c -o Server
it@it-Vostro-3710:~/pramod$ cc Client.c -o Client
it@it-Vostro-3710:~/pramod$ ./Server ./Client
Writing to shared memory...
Message written to shared memory: Hello from DVVPCOE, Ahmednagar Server!
it@it-Vostro-3710:~/pramod$
```

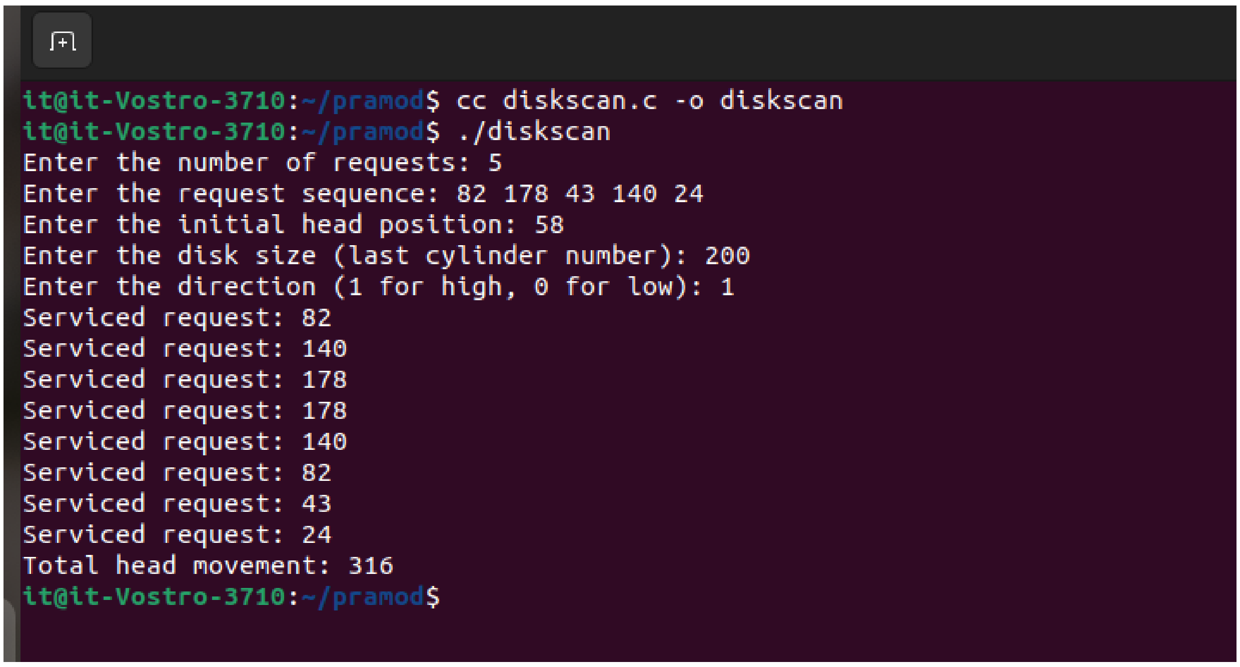
```
#include <stdio.h>
#include <stdlib.h>
int main() {
    int n, i, j, head, total_movement = 0;
    printf("Enter the number of requests: ");
    scanf("%d", &n);
    int requests[n], completed[n];
    printf("Enter the request sequence: ");
    for (i = 0; i < n; i++) {
        scanf("%d", &requests[i]);
        completed[i] = 0; // Mark all requests as uncompleted initially
    }
    printf("Enter the initial head position: ");
    scanf("%d", &head);
    for (i = 0; i < n; i++) {
        int min = 10000, min_index = -1;
        for (j = 0; j < n; j++) {
            if (!completed[j] && abs(head - requests[j]) < min) {
                min = abs(head - requests[j]);
                min_index = j;
            }
        }
        completed[min_index] = 1; // Mark the request as completed
        total_movement += abs(head - requests[min_index]);
        head = requests[min_index];
        printf("Serviced request: %d\n", head);
    }
    printf("Total head movement: %d\n", total_movement);
    return 0;
}
```



```
it@it-Vostro-3710:~/pramod$ cc sstf.c -o sstf
it@it-Vostro-3710:~/pramod$ ./sstf
Enter the number of requests: 5
Enter the request sequence: 82 170 43 140 24
Enter the initial head position: 50
Serviced request: 43
Serviced request: 24
Serviced request: 82
Serviced request: 140
Serviced request: 170
Total head movement: 172
it@it-Vostro-3710:~/pramod$
```

```
#include <stdio.h>
#include <stdlib.h>
int main() {
    int n, i, head, total_movement = 0, direction;
    printf("Enter the number of requests: ");
    scanf("%d", &n);
    int requests[n];
    printf("Enter the request sequence: ");
    for (i = 0; i < n; i++) {
        scanf("%d", &requests[i]);
    }
    printf("Enter the initial head position: ");
    scanf("%d", &head);
    printf("Enter the disk size (last cylinder number): ");
    int disk_size;
    scanf("%d", &disk_size);
    printf("Enter the direction (1 for high, 0 for low): ");
    scanf("%d", &direction);
    // Sort the request array
```

```
for (i = 0; i < n - 1; i++) {
    for (int j = i + 1; j < n; j++) {
        if (requests[i] > requests[j]) {
            int temp = requests[i];
            requests[i] = requests[j];
            requests[j] = temp;
        }
    }
}
// SCAN algorithm
if (direction == 1) { // Move towards higher end
    for (i = 0; i < n && requests[i] < head; i++);
    for (; i < n; i++) {
        printf("Serviced request: %d\n", requests[i]);
        total_movement += abs(head - requests[i]);
        head = requests[i];
    }
    if (head < disk_size - 1) {
        total_movement += abs(head - (disk_size - 1));
        head = disk_size - 1;
    }  for (i--; i >= 0; i--) {
        printf("Serviced request: %d\n", requests[i]);
        total_movement += abs(head - requests[i]);
        head = requests[i];
    }
} else { // Move towards lower end
    for (i = n - 1; i >= 0 && requests[i] > head; i--);
    for (; i >= 0; i--) {
        printf("Serviced request: %d\n", requests[i]);
        total_movement += abs(head - requests[i]);
        head = requests[i];
        }
        if (head > 0) {
            total_movement += head;
            head = 0;
        }
for (i++; i < n; i++) {
            printf("Serviced request: %d\n", requests[i]);
            total_movement += abs(head - requests[i]);
            head = requests[i];
        }
    }
 printf("Total head movement: %d\n", total_movement);
    return 0;
}
```



```
it@it-Vostro-3710:~/pramod$ cc diskscan.c -o diskscan
it@it-Vostro-3710:~/pramod$ ./diskscan
Enter the number of requests: 5
Enter the request sequence: 82 178 43 140 24
Enter the initial head position: 58
Enter the disk size (last cylinder number): 200
Enter the direction (1 for high, 0 for low): 1
Serviced request: 82
Serviced request: 140
Serviced request: 178
Serviced request: 178
Serviced request: 140
Serviced request: 82
Serviced request: 43
Serviced request: 24
Total head movement: 316
it@it-Vostro-3710:~/pramod$
```