```
#include <stdio.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <string.h>
#include <unistd.h>
void main(){
        char ch[50];
        printf("Enter the string : ");
        scanf("%s",ch);
        printf("Entered string : ");
        puts(ch);
        pid_t pid;
        pid=fork();
        if(pid>=0){
                if(pid==0){
                        int i=0,l=0;
                        while(ch[i]!='\0'){
                                l++;
                                i++;
                        }
                        for(int q=0;q<=(l/2);q++){
                                char temp=ch[q];
                                ch[q]=ch[l-q-1];
                                ch[l-q-1]=temp;
                        }
                        printf("The reversed word is  \n");
                        puts(ch);
                }else{

                        wait(NULL);
                        printf("child process completed \n");
                }
        }else{
                perror("Error..");
        }
}
```

```
jishnu@pop-os:~/Desktop/OS Lab$ gcc exp_03.c
jishnu@pop-os:~/Desktop/OS Lab$ ./a.out
Enter the string : programming
The reversed word : gnimmargorp
child process completed
jishnu@pop-os:~/Desktop/OS Lab$
```

server.c

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <ctype.h>
#define SHMSIZE 100
void main(){
        int shmid,i;int cdtn=0;
        key_t key;char ch,*shm,*s;
        char rev[50];key=5678;
        if((shmid=shmget(key,SHMSIZE,IPC_CREAT|0666))<0){
                perror("shmget error");
                exit(1);
        }
        if((shm=shmat(shmid,NULL,0))==(char *)-1){
                perror("shmat error");
                exit(1);
        }
        printf("Enter the string \n");
        s=shm;scanf("%s",s);
        int l=0;
        if(fork()==0){
                char *args[]={"./client",NULL};
        execv(args[0],args);
      }
        while(*(s+l)!='\0'){
                rev[l]=*(s+l);
                l++;
        }
        *(s+l+2)='Z';
        while(*(s+l+2)=='Z');
        for(int i=0;i<=l;i++){
                if(*(s+i)!=rev[i]){
                        cdtn=1;
                        break;
                }
        }
        if(cdtn==0){
                printf("The given string is palindrome \n");
        }else{
                printf("The given string is not a palindrome \n");
                exit(0);
        }
}
```

client

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

```
#include <stdlib.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <unistd.h>
#include <ctype.h>
#define SHMSIZE 100
void main(){
        int shmid,i;
        key_t key;
        char ch,*shm,*s;
        key=5678;
        if((shmid=shmget(key,SHMSIZE,IPC_CREAT|0666))<0){
                perror("shmget error  \n");
                exit(1);
        }
        if((shm=shmat(shmid,NULL,0))==(char *)-1){
                perror("shmat error \n");
                exit(1);}
        s=shm;int l=0;
        while(*(s+l)!='\0') l++;
        for(int i=0;i<(l/2);i++){
                char temp=*(s+i);
                *(s+i)=*(s+l-i-1);
                *(s+l-i-1)=temp;
        }
        *(s+l+2)='z';
        printf("The client has reversed the word \n");
        printf("The reversed word is \n");
        puts(s);

exit(0);
}
```

```
jishnu@pop-os:~/Desktop/OS Lab$ gcc server
/usr/bin/ld: cannot find server: No such file or directory
collect2: error: ld returned 1 exit status
jishnu@pop-os:~/Desktop/OS Lab$ gcc server_e04.c
jishnu@pop-os:~/Desktop/OS Lab$ ./a.out
Enter the string
jishnu
The given string is not a palindrome
```

```
jishnu@pop-os:~/Desktop/OS Lab$ gcc client_e04.c
jishnu@pop-os:~/Desktop/OS Lab$ ./a.out
The client has reversed the word
The reversed word is
unhsij
jishnu@pop-os:~/Desktop/OS Lab$
```

5.

```
#include<stdio.h>
typedef struct process{int pid;
int bt;
int tat;
int wt;
}proc;
void main(){
int n;
float ttat=0,atat=0;
float twt=0,awt=0;
proc p[10];
printf("Enter the total number of process : \n");
scanf("%d",&n);
printf("Enter the process id,burst time of each process:\n");
printf("\nPid\tBt\n");
printf("---------------\n");
for(int i=1;i<=n;i++){
scanf("%d%d",&p[i].pid,&p[i].bt);
}
//calculating turnaround time
p[0].tat=0;
for(int i=1;i<=n;i++){
p[i].tat = p[i-1].tat + p[i].bt;
}
//calculating waitting time
for(int i=1;i<=n;i++){
p[i].wt = p[i-1].tat;
}
//calculating average turnaround time
for(int i=1;i<=n;i++){
ttat = ttat + p[i].tat;
}
atat = (float)ttat / n;//calculating average waiting time
for(int i=1;i<=n;i++){
twt = twt + p[i].wt;
}
awt = (float)twt / n;
// print the table
printf("\nPid\tBt\tWt\tTat\n");
printf("-------------------------------\n");
for(int i=1;i<=n;i++){
printf("%d\t%d\t%d\t%d\n",p[i].pid,p[i].bt,p[i].wt,p[i].tat);
}
printf("\n");
printf("Average turnaround time= %.2fmsec",atat);
printf("\n");
printf("Average waiting time= %.2fmsec\n",awt);
}
```

```
#include <stdio.h>
# define MAXSIZE 10
int n;
struct process{
int pid;int bt;
int wt;
int tt;
}P[MAXSIZE],temp;
void sort(){
for(int i=1;i<=n;i++){
for(int j=0;j<n-i;j++){
if(P[j].bt>P[j+1].bt){
temp=P[j];
P[j]=P[j+1];
P[j+1]=temp;
}
}
}
}
void waitingTime(){
P[0].wt=0;
for(int i=1;i<n;i++){
P[i].wt=P[i-1].wt+P[i-1].bt;
}
}
void turnAroundTime(){
P[0].tt=P[0].bt;
for(int i=1;i<n;i++){
P[i].tt=P[i-1].tt+P[i].bt;
}
}
float avg_wt(){
float total=0;for(int i=0;i<n;i++){
total=total+P[i].wt;
}
return total/n;
}
float avg_tt(){
float total=0;
for(int i=0;i<n;i++){
total=total+P[i].tt;
}
return total/n;
}
void main()
{
printf("Enter no.of processes:");
scanf("%d",&n);
for(int i=0;i<n;i++){
printf("Enter pid of process %d:",i+1);
scanf("%d",&P[i].pid);
```

```
printf("Enter burst time of process %d:",i+1);
scanf("%d",&P[i].bt);
}
sort();
waitingTime();turnAroundTime();
printf("pid bt wt tt\n");
for(int i=0;i<n;i++){
printf(" %d %d %d %d\n",P[i].pid,P[i].bt,P[i].wt,P[i].tt);
}
float avg_waitingtime,avg_turnaroundtime;
avg_waitingtime=avg_wt();
avg_turnaroundtime=avg_tt();
printf("Avg waiting time=%f",avg_waitingtime);
printf("\nAvg turnaroundtime time=%f\n",avg_turnaroundtime);
}
```

```
#include <stdio.h>
# define MAXSIZE 10
int n,tq;
struct process{
int pid;
int bt;
int wt;
int tt;
int at;
}P[MAXSIZE];
float avg_wt(){
float total=0;
for(int i=0;i<n;i++){
total=total+P[i].wt;
}
return total/n;}
float avg_tt(){
float total=0;
for(int i=0;i<n;i++){
total=total+P[i].tt;
}
return total/n;
}
void main(){
int i,total,x,counter,temp[10];
printf("Enter no.of processes:");
scanf("%d",&n);
x=n;
for(int i=0;i<n;i++){
printf("Enter pid of process %d:",i+1);
scanf("%d",&P[i].pid);
printf("Enter arrival time of process %d:",i+1);
scanf("%d",&P[i].at);
```

```
printf("Enter burst time of process %d:",i+1);
scanf("%d",&P[i].bt);
temp[i]=P[i].bt;
}
printf("Enter time quantum:");
scanf("%d",&tq);
for(total=0,i=0;x!=0;){
if(temp[i]<=tq&&temp[i]>0){total=total+temp[i];
temp[i]=0;
counter=1;
}
else if(temp[i]>0){
total=total+tq;
temp[i]=temp[i]-tq;
}
if(temp[i]==0&&counter==1){
x--;
P[i].tt=total-P[i].at;
P[i].wt=total-P[i].at-P[i].bt;
counter=0;
}
else if(i==n-1){
i=0;
}
else if(P[i+1].at<=total){
i++;
}
else{
i=0;
}
}
printf("pid bt wt tt\n");
for(int i=0;i<n;i++){printf(" %d %d %d %d\n",P[i].pid,P[i].bt,P[i].wt,P[i].tt);
}
float avg_waitingtime,avg_turnaroundtime;
avg_waitingtime=avg_wt();
avg_turnaroundtime=avg_tt();
printf("Avg waiting time=%f",avg_waitingtime);
printf("\nAvg turnaroundtime time=%f\n",avg_turnaroundtime);
}
```

```
#include <stdio.h>
# define MAXSIZE 10
int n;
struct process{
int pid;
int bt;
int wt;
```

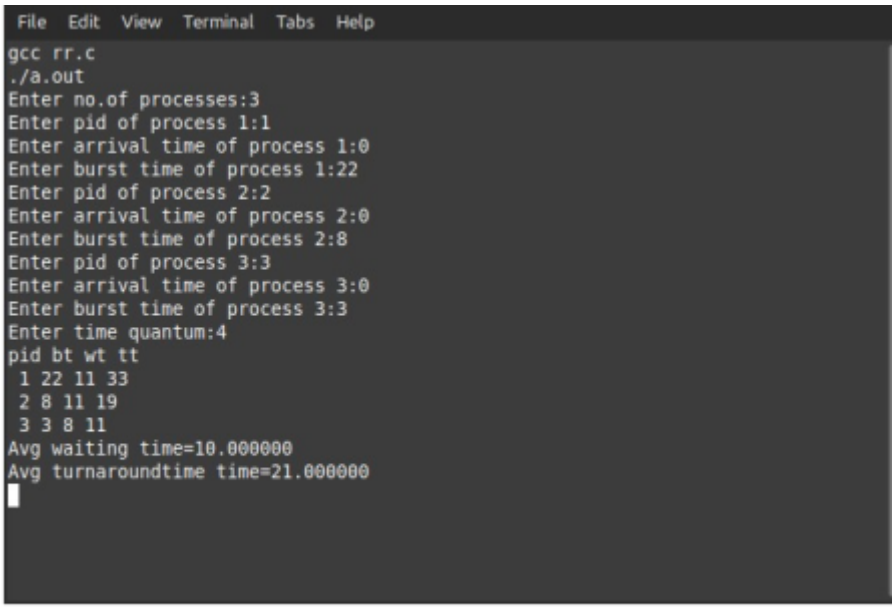
```
int tt;
int priority;
}P[MAXSIZE],temp;
void sort(){
for(int i=1;i<=n;i++){
for(int j=0;j<n-i;j++){
if(P[j].priority>P[j+1].priority){
temp=P[j];
P[j]=P[j+1];
P[j+1]=temp;
}
}}
}
void waitingTime(){
P[0].wt=0;
for(int i=1;i<n;i++){
P[i].wt=P[i-1].wt+P[i-1].bt;
}
}
void turnAroundTime(){
P[0].tt=P[0].bt;
for(int i=1;i<n;i++){
P[i].tt=P[i-1].tt+P[i].bt;
}
}
float avg_wt(){
float total=0;
for(int i=0;i<n;i++){
total=total+P[i].wt;
}
return total/n;
}
float avg_tt(){
float total=0;
for(int i=0;i<n;i++){total=total+P[i].tt;
}
return total/n;
}
void main()
{
printf("Enter no.of processes:");
scanf("%d",&n);
for(int i=0;i<n;i++){
printf("Enter pid of process %d:",i+1);
scanf("%d",&P[i].pid);
printf("Enter burst time of process %d:",i+1);
scanf("%d",&P[i].bt);
printf("Enter priority of process %d:",i+1);
scanf("%d",&P[i].priority);
}
sort();
waitingTime();
```

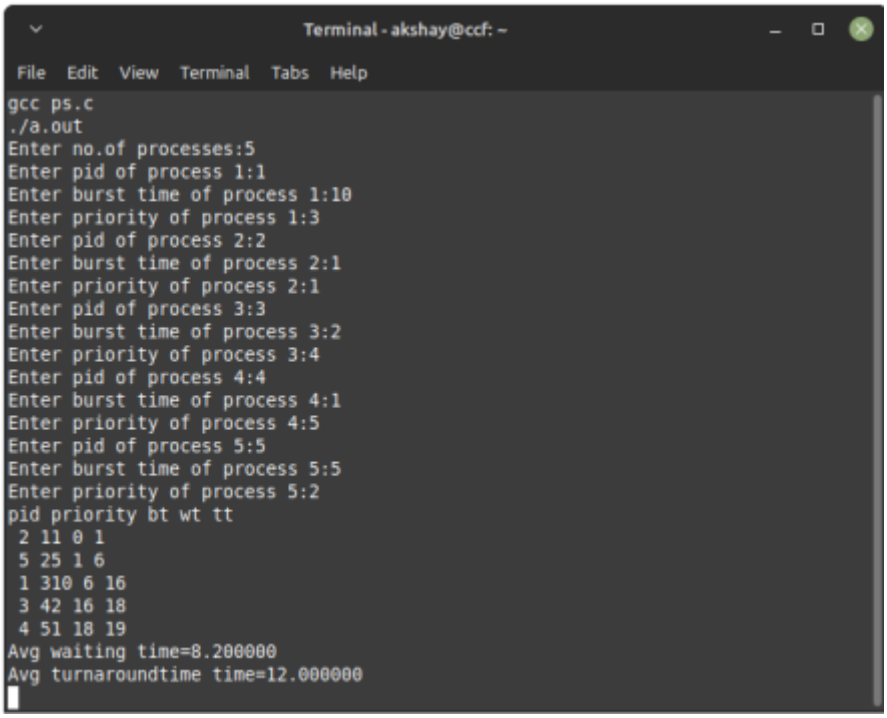
```
turnAroundTime();
printf("pid priority bt wt tt\n");
for(int i=0;i<n;i++){
printf(" %d %d%d %d %d\n",P[i].pid,P[i].priority,P[i].bt,P[i].wt,P[i].tt);
}
float avg_waitingtime,avg_turnaroundtime;
avg_waitingtime=avg_wt();
avg_turnaroundtime=avg_tt();
printf("Avg waiting time=%f",avg_waitingtime);
printf("\nAvg turnaroundtime time=%f\n",avg_turnaroundtime);
}
```

```
ch=1
while(($ch<=4))
do



echo "1.list details of students in your class"
echo "2.Get separate list of Male and Female students"
echo "3.Get details of students belong to a given Blood Group"
echo "4. List of students whose year of birth is 2003"
echo "enter your choice: "

read ch
case $ch in
1) cat < stud.csv ;;

2)  echo " male students "
  printf "\n"
   cat stud.csv|grep -w "Male"|cut -d "," -f 1
   echo " female students"
      printf "\n"
    cat stud.csv|grep -w "Female"|cut -d "," -f 1;;

3) echo "enter the required blood group"
   read bloodgroup
    cat stud.csv|grep -w "$bloodgroup";;
    ##cat stud.csv|cut -f5 -d ",";;
4) cat stud.csv|grep -w "2003"|cut -d "," -f1;;
esac
done
```

```
File  Edit  View  Terminal  Tabs  Help
gcc fcfs.c
./a.out
Enter the total number of process :
3
Enter the process id,burst time of each process:

Pid     Bt
--------------
1       22
2       8
3       3

Pid     Bt      Wt      Tat
------------------------------
1       22      0       22
2       8       22      30
3       3       30      33

Average turnaround time= 28.33msec
Average waiting time= 17.33msec
```

```
Terminal - akshay@ccf: ~
File  Edit  View  Terminal  Tabs  Help
gcc sjf.c
./a.out
Enter no.of processes:3
Enter pid of process 1:1
Enter burst time of process 1:22
Enter pid of process 2:2
Enter burst time of process 2:8
Enter pid of process 3:3
Enter burst time of process 3:3
pid bt wt tt
 3 3 0 3
 2 8 3 11
 1 22 11 33
Avg waiting time=4.666667
Avg turnaroundtime time=15.666667
```


File Edit View Terminal Tabs Help
gcc rr.c
./a.out
Enter no.of processes:3
Enter pid of process 1:1
Enter arrival time of process 1:0
Enter burst time of process 1:22
Enter pid of process 2:2
Enter arrival time of process 2:0
Enter burst time of process 2:8
Enter pid of process 3:3
Enter arrival time of process 3:0
Enter burst time of process 3:3
Enter time quantum:4
pid bt wt tt
1 22 11 33
2 8 11 19
3 3 8 11
Avg waiting time=10.000000
Avg turnaroundtime time=21.000000


Terminal - akshay@ccf: ~
File Edit View Terminal Tabs Help
gcc ps.c
./a.out
Enter no.of processes:5
Enter pid of process 1:1
Enter burst time of process 1:10
Enter priority of process 1:3
Enter pid of process 2:2
Enter burst time of process 2:1
Enter priority of process 2:1
Enter pid of process 3:3
Enter burst time of process 3:2
Enter priority of process 3:4
Enter pid of process 4:4
Enter burst time of process 4:1
Enter priority of process 4:5
Enter pid of process 5:5
Enter burst time of process 5:5
Enter priority of process 5:2
pid priority bt wt tt
2 11 0 1
5 25 1 6
1 310 6 16
3 42 16 18
4 51 18 19
Avg waiting time=8.200000
Avg turnaroundtime time=12.000000