**CODE FOR DYNAMIC IMPLEMENTATION OF FCFS SCHEDULING**

* The programming code for implementing the FCFS scheduling is given as follows :

#include<stdio.h>

int main()

{

int n,bt[20],wt[20],tat[20],avwt=0,avtat=0,i,j;

printf("Enter total number of processes(maximum 20):");

scanf("%d",&n);

printf("nEnter Process Burst Timen");

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

{

printf("P[%d]:",i+1);

scanf("%d",&bt[i]);

}

wt[0]=0;

for(i=1;i<n;i++)

{

wt[i]=0;

for(j=0;j<i;j++)

wt[i]+=bt[j];

}

printf("nProcessttBurst TimetWaiting TimetTurnaround Time");

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

{

tat[i]=bt[i]+wt[i];

avwt+=wt[i];

avtat+=tat[i];

printf("nP[%d]tt%dtt%dtt%d",i+1,bt[i],wt[i],tat[i]);

}

avwt/=i;

avtat/=i;

printf("nnAverage Waiting Time:%d",avwt);

printf("nAverage Turnaround Time:%d",avtat);

return 0;

}

**EXPLANATION**

* In the above code, the demonstration of the first come first serve scheduling algorithm is shown. The user is asked to enter the number of processes. On entering the number of processes, we have to enter the burst times for each of the processes.
* The waiting time is calculated first. First, the waiting time of the first process is zero.

for(i=1;i<n;i++)

{

wt[i]=0;

for(j=0;j<i;j++)

wt[i]+=bt[j];

}

* Calculation of the waiting time is done by adding the burst time of the previous process. Consider the previous process had a burst time of 10, then the waiting time of second will be 10. Similarly, for the third process, the waiting time will be the sum of burst times of first and second processes.

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

{

tat[i]=bt[i]+wt[i];

avwt+=wt[i];

avtat+=tat[i];

printf("nP[%d]tt%dtt%dtt%d",i+1,bt[i],wt[i],tat[i]);

}

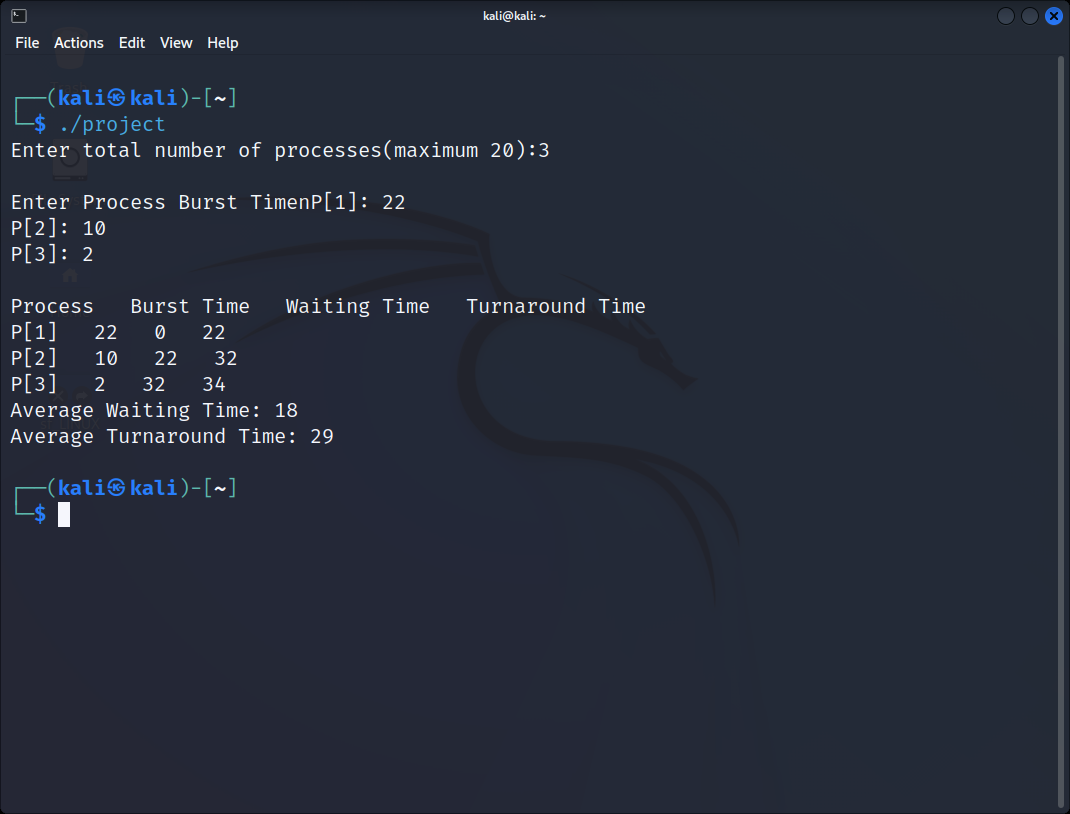
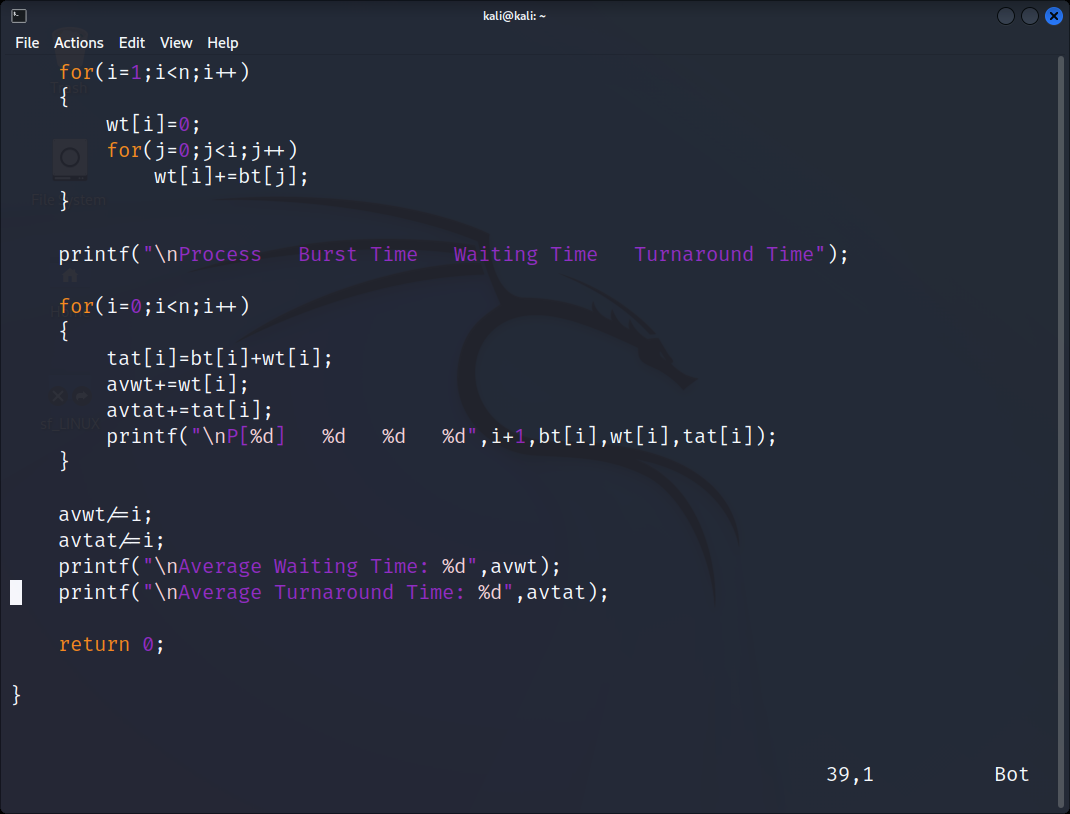
* The next part we calculate the turnaround time. The turnaround time for each process is calculated by adding the burst time and the waiting time.Last, the average turn around time and the average waiting time is calculated.

avwt/=i;

avtat/=i;

* It gives the total number of processes. We divide the sum of all the waiting times and turn around times to get the average. This is how the first come first serve algorithm works.

**OUTPUT**

****