

```

sumedh@Sumedh: $ ls
OS_Practicals  cpp_program
sumedh@Sumedh: $ cd OS_Practicals
sumedh@Sumedh:~/OS_Practicals$ ls
Addition_program.txt  practical1  practical2  practical4  practical6  practical8
a.out                 practical10 practical3  practical5  practical7  practical9
sumedh@Sumedh:~/OS_Practicals$ cd practical6
sumedh@Sumedh:~/OS_Practicals/practical6$ nano SJF.c
sumedh@Sumedh:~/OS_Practicals/practical6$ gcc SJF
/usr/bin/ld: cannot find SJF: No such file or directory
collect2: error: ld returned 1 exit status
sumedh@Sumedh:~/OS_Practicals/practical6$ gcc SJF.c
sumedh@Sumedh:~/OS_Practicals/practical6$ ./a.out
Process Burst Time      Waiting Time      Turnaround Time
1          6             8                 14
2          8            23                 31
3          7            15                 22
4          3             2                  5

Average Waiting Time: 48.00
Average Turnaround Time: 72.00
sumedh@Sumedh:~/OS_Practicals/practical6$ █

```

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

```
#define MAX_PROCESSES 10
```

```
//Function to find the waiting time for each process
```

```
void findWaitingTime(int processes[],int n,int bt[],int wt[])
```

```
{
```

```
int remaining_bt[MAX_PROCESSES];
```

```
for(int i =0;i<n;i++){
```

```
remaining_bt[i] = bt[i];
```

```
}
```

```
int current_time = 0;
```

```
while(1){
```

```
int min_bt = 9999;//Initializing with a large number
```

```
int shortest = -1;
```

```
int done = 1;
```

```
//Find the process with the smallest remaining burst time
```

```
for(int i = 0; i<n;i++){
```

```
if(remaining_bt[i]>0 && remaining_bt[i]<min_bt){
```

```
min_bt = remaining_bt[i];
```

```

shortest = i;
done = 0;
}
}
//if all processes are done, break the loop
if(done){
break;
}
//update the waiting time for the selected process
wt[shortest] = current_time;
current_time++;
//decrease the remaining burst time of the shortest process
remaining_bt[shortest]--;
}
}
//function to calculate turnaround time for each process
void findTurnaroundTime(int processes[], int n, int bt[], int wt[],int tat[]){
for(int i = 0;i<n;i++){
tat[i]=bt[i]+wt[i];
}
}
//function to calculate average times
void calculateAverageTimes(int processes[], int n, int bt[]){
int wt[MAX_PROCESSES],tat[MAX_PROCESSES];
int total_wt = 0,total_tat=0;

//Find waiting time and turnaround time for each process
findWaitingTime(processes,n,bt,wt);
findTurnaroundTime(processes,n,bt,wt,tat);
//calculate total waiting time and total turnaround time
for(int i =0; i<n;i++){

```

```

total_wt+=wt[i];
total_tat+=tat[i];
}
//print the results
printf("Process\tBurst Time\tWaiting Time\tTurnaround Time\n");
for(int i = 0 ; i<n;i++){
printf("%d\t\t%d\t\t%d\t\t%d\n",processes[i],bt[i],wt[i],tat[i]);
}
printf("\nAverage Waiting Time: %.2f\n",(float)total_wt);
printf("Average Turnaround Time: %.2f\n",(float)total_tat);
}

int main(){
int processes[]={1,2,3,4}; //Process IDs
int burst_time[]={6,8,7,3}; //Burst times for each process
int n = sizeof(processes)/sizeof(processes[0]); //Number of processes
calculateAverageTimes(processes,n,burst_time);
return 0;
}

```