Shortest Job First Scheduling

Non Premptive

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

struct process
{
  char pname[20];
  int at,bt,wt,tt,status;
}p[30];

struct done
{
  char name[20];
  int st,ct;
}d[20];

main()
{
  int num,found,flag,i,k,j,idle=0,ls,n,l;
  float atat=0,atwt=0;
  printf("Enter the number of processes:");
  scanf("%d",&n);
  for(i=0;i<n;i++)
    {
      printf("\nProcess %d \n",i+1);
      printf("Name: ");
      scanf("%s",p[i].pname);
      printf("Arrival time:");
      scanf("%d",&p[i].at);
      printf("Burst time:");
      scanf("%d",&p[i].bt);
      p[i].status=0;
    }

  printf("\n");
  for(i=0;ls<n;)
    {
      flag=0;
      found=0;
      for(j=0;j<n;j++)
  {
    if(p[j].status==0&&p[j].at<=i&&found==0)
      {
        k=j;
        found++;
```

```
        flag++;
      }
    else if(found>0&&p[j].status==0&&p[j].at<=i)
      {
        if(p[j].bt<p[k].bt)
    {
      k=j;
    }
        else if(p[j].bt==p[k].bt)
    {
      if(p[j].at<p[k].at)
        {
          k=j;
        }
    }
      }
}
      if(flag==0&&idle==0)
{
   idle=1;
   strcpy(d[num].name,"IDLE");
   d[num].st=i;
   i++;
}
      else if(flag>0)
{
   if(idle==1)
     {
       d[num].ct=i;
       num++;
     }
   strcpy(d[num].name,p[k].pname);
   d[num].st=i;
   d[num].ct=i+p[k].bt;
   p[k].wt=d[num].st-p[k].at;
   p[k].tt=p[k].wt+p[k].bt;
   i=d[num].ct;
   p[k].status=1;
   num++;
   idle=0;
   ls++;
}
      else
{
   i++;
}
   }
printf("\n Process Table \n");
```

```
    printf("\nPname\tAT\tBT\tStatus\tWT\tTT\n");
    for(i=0;i<n;i++)
      {
        printf("%s\t%d\t%d\t%d\t%d\t%d\n",p[i].pname,p[i].at,p[i].bt,p[i].sta
      }
    printf("\n");
    printf("\nName\tST\tCT\n");
    for(i=0;i<num;i++)
      {
        printf("%s\t%d\t%d\n",d[i].name,d[i].st,d[i].ct);
      }
    printf("\nGantt Chart\n");
    for(i=0;i<num;i++)
      {
        printf("----------");
      }
    printf("\n");
    for(i=0;i<num;i++)
      {
        printf("|   %s   ",d[i].name);
      }
    printf("|");
    printf("\n");
    for(i=0;i<num;i++)
      {
        printf("----------");
      }
    printf("\n");
    for(i=0;i<num;i++)
      {
        printf("%d        ",d[i].st);
        l=i;
      }
    printf("%d",d[l].ct);
    printf("\n\n");

    for(i=0;i<num;i++)
      {
        atwt+=p[i].wt;
        atwt=atwt/num;
        atat+=p[i].tt;
        atat=atat/num;
      }
    printf("\n Average Waiting Time %2f",atwt);
    printf("\n Average Turnaround Time %2f\n",atat);
}
```

Output

```
42813@user:/mnt/42813/oslab$ gcc sjf.c
42813@user:/mnt/42813/oslab$ ./a.out
Enter the number of processes:4

Process 1
Name: P0
Arrival time:0
Burst time:7

Process 2
Name: P1
Arrival time:5
Burst time:2

Process 3
Name: P2
Arrival time:6
Burst time:4

Process 4
Name: P3
Arrival time:8
Burst time:5

 Process Table

Pname AT  BT  Status  WT  TT
P0  0 7 1 0 7
P1  5 2 1 2 4
P2  6 4 1 3 7
P3  8 5 1 5 10


Name  ST  CT
P0  0 7
P1  7 9
P2  9 13
P3  13  18

Gantt Chart
```

P0	P1	P2	P3
0	7	9	13	18

```
Average Waiting Time 1.468750
Average Turnaround Time 3.027344
```

Priority Scheduling

Non Premptive

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

struct process
{
  char pname[20];
  int at,bt,wt,tt,status,priority;
}p[30];

struct done
{
 char name[20];
  int st,ct;
}d[20];

main()
{
  int num,found,flag,i,k,j,idle=0,ls,n,l;
  float atat=0,atwt=0;
  printf("Enter the number of processes:");
  scanf("%d",&n);
  for(i=0;i<n;i++)
    {
      printf("\nProcess %d \n",i+1);
      printf("Name: ");
      scanf("%s",p[i].pname);
      printf("Enter the priority:\n");
      scanf("%d",&p[i].priority);
      printf("Arrival time:");
      scanf("%d",&p[i].at);
      printf("Burst time:");
      scanf("%d",&p[i].bt);
      p[i].status=0;
    }

  printf("\n");
  for(i=0;ls<n;)
    {
      flag=0;
      found=0;
      for(j=0;j<n;j++)
  {
    if(p[j].status==0&&p[j].at<=i&&found==0)
      {
```

```
        k=j;
        found++;
        flag++;
      }
    else if(found>0&&p[j].status==0&&p[j].at<=i)
      {
        if(p[j].priority<p[k].priority) //Changed for Priority
    {
      k=j;
    }
        else if(p[j].bt==p[k].bt)
    {
      if(p[j].priority<p[k].priority) //Changed for Priority
        {
          k=j;
        }
    }
      }
  }
      if(flag==0&&idle==0)
  {
    idle=1;
    strcpy(d[num].name,"IDLE");
    d[num].st=i;
    i++;
  }
      else if(flag>0)
  {
    if(idle==1)
      {
        d[num].ct=i;
        num++;
      }
    strcpy(d[num].name,p[k].pname);
    d[num].st=i;
    d[num].ct=i+p[k].bt;
    p[k].wt=d[num].st-p[k].at;
    p[k].tt=p[k].wt+p[k].bt;
    i=d[num].ct;
    p[k].status=1;
    num++;
    idle=0;
    ls++;
  }
      else
  {
    i++;
  }
```

```
    }
  printf("\n Process Table \n");
  printf("\nPname\tAT\tBT\tStatus\tWT\tTT\tPRIORITY\n");
  for(i=0;i<n;i++)
    {
      printf("%s\t%d\t%d\t%d\t%d\t%d\t%d\n",p[i].pname,p[i].at,p[i].bt,p[i]
    }
  printf("\n");
  printf("\nName\tST\tCT\n");
  for(i=0;i<num;i++)
    {
      printf("%s\t%d\t%d\n",d[i].name,d[i].st,d[i].ct);
    }
  printf("\nGantt Chart\n");
  for(i=0;i<num;i++)
    {
      printf("----------");
    }
  printf("\n");
  for(i=0;i<num;i++)
    {
      printf("|   %s   ",d[i].name);
    }
  printf("|");
  printf("\n");
  for(i=0;i<num;i++)
    {
      printf("----------");
    }
  printf("\n");
  for(i=0;i<num;i++)
    {
      printf("%d        ",d[i].st);
      l=i;
    }
  printf("%d",d[l].ct);
  printf("\n\n");

  for(i=0;i<num;i++)
    {
      atwt+=(float)p[i].wt;
      atwt=atwt/(float)num;
      atat+=(float)p[i].tt;
      atat=atat/(float)num;
    }
  printf("\n Average Waiting Time %2f",atwt);
  printf("\n Average Turnaround Time %2f\n",atat);
}
```

Output

```
42813@user:/mnt/42813/oslab$ gcc sjfprio.c
42813@user:/mnt/42813/oslab$ ./a.out
Enter the number of processes:4

Process 1
Name: P0
Enter the priority:
2
Arrival time:0
Burst time:5

Process 2
Name: P1
Enter the priority:
3
Arrival time:3
Burst time:4

Process 3
Name: P2
Enter the priority:
1
Arrival time:2
Burst time:2

Process 4
Name: P3
Enter the priority:
4
Arrival time:4
Burst time:3


 Process Table

Pname AT  BT  Status  WT  TT  PRIORITY
P0   0  5  1  0  5  2
P1   3  4  1  4  8  3
P2   2  2  1  3  5  1
P3   4  3  1  7  10  4


Name  ST  CT
P0   0  5
P2   5  7
P1   7  11
```

P3 11 14

Gantt Chart

```
| P0 | P2 | P1 | P3 |
0    5    7    11   14
```

Average Waiting Time 2.000000
Average Turnaround Time 2.957031