WEEK 1

Write a C program to simulate the following non-pre-emptive CPU scheduling algorithm to find turnaround time and waiting time.

- FCFS
- SJF (pre-emptive & Non-pre-emptive)

CODE:

```
#include<stdio.h>

#include<conio.h>

int at[20],bt[20], wt[20], tat[20], i, n;

float wtavg, tatavg;

void fcfs()

{


     wt[0] = wtavg = 0;

     tat[0] = tatavg = bt[0];

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

     {

      wt[i] = wt[i-1] +bt[i-1];

      tat[i] = tat[i-1] +bt[i];

      wtavg = wtavg + wt[i];

      tatavg = tatavg + tat[i];

     }

}

void srjf()

{
 int i, smallest, count = 0, time, temp[10];
    double wait_time = 0, turnaround_time = 0, end;
   // float wtavg, average_turnaround_time;

    bt[9] = 9999;
    for(time = 0; count != n; time++)
    {
        smallest = 9;
        for(i = 0; i < n; i++)
        {
            if(at[i] <= time && bt[i] < bt[smallest] && bt[i] > 0)
            {
                smallest = i;
            }
        }
        bt[smallest]--;
        if(bt[smallest] == 0)
        {
            count++;
            end = time + 1;
            wait_time = wait_time + end - bt[smallest] - temp[smallest];
            turnaround_time = turnaround_time + end - at[smallest];
        }
    }
    wtavg = wait_time /n;
  tatavg = turnaround_time / n;

}
```

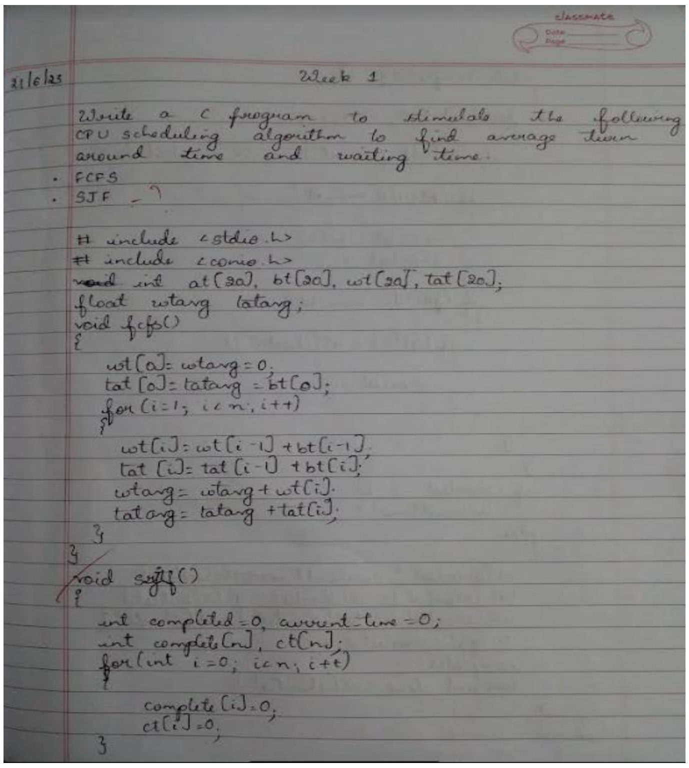
```
void sjf()
{
    int completed = 0;
    int currentTime = 0;
    int complete[n], ct[n];
    double atat, awt;
 for (int i = 0; i < n; i++)
    {
        complete[i] = 0;
        ct[i] = 0;
    }

    while (completed != n)
    {
        int shortest = -1;
        int min_bt = 9999;

        for (int i = 0; i < n; i++)
        {
            if (at[i] <= currentTime && complete[i] == 0)
            {
                if (bt[i] < min_bt)
                {
                    min_bt = bt[i];
                    shortest = i;
                }
                if (bt[i] == min_bt)
                {
```

```
                if (at[i] < at[shortest])
                {
                    shortest = i;
                }
            }
        }
    }

    if (shortest == -1)
    {
        currentTime++;
    }
    else
    {
        ct[shortest] = currentTime + bt[shortest];
        tat[shortest] = ct[shortest] - at[shortest];
        wt[shortest] = tat[shortest] - bt[shortest];
        complete[shortest] = 1;
        completed++;
        currentTime = ct[shortest];
    }
}

for (int i = 0; i < n; i++)
{
    atat += tat[i];
    awt += wt[i];
}
```

```
    atat = atat / n;

    awt = awt / n;


    /*for (int i = 0; i < n; i++)

    {

        printf("P%d\t%d\t%d\n", i, tat[i], wt[i]);

    }*/


    printf("\nAverage TAT = %f\nAverage WT = %f\n", atat, awt);

}

int main()

{

  int ch;

  printf("\nEnter the number of processes  ");

  scanf("%d", &n);

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

  {


   printf("\nEnter Arrival time and Burst Time for Process" );

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


  }

   printf("1. FCFS 2. SJF 3. SRTF");


   printf("\n Enter your choice");

   scanf("%d", &ch);

switch(ch)
```

```
  {
     case 1: fcfs();
             printf("\t PROCESS \tARRIVAL TIME \t \tBURST TIME \t WAITING TIME\t
TURNAROUND TIME\n");
             for(i=0;i<n;i++)
             {
                printf("\n\t P%d \t\t %d \t\t %d \t\t %d \t\t %d", i,at[i], bt[i], wt[i], tat[i]);
             }
                printf("\nAverage Waiting Time  %f", wtavg/n);
                printf("\nAverage Turnaround Time  %f", tatavg/n);
                break;
   case 2 : sjf();
          break;

     case 3 : srjf();



            printf("\n\nAverage Waiting Time:\t%lf\n", wtavg);
             printf("Average Turnaround Time:\t%lf\n", tatavg);
             break;
  }
}
```

OBSERVATION :



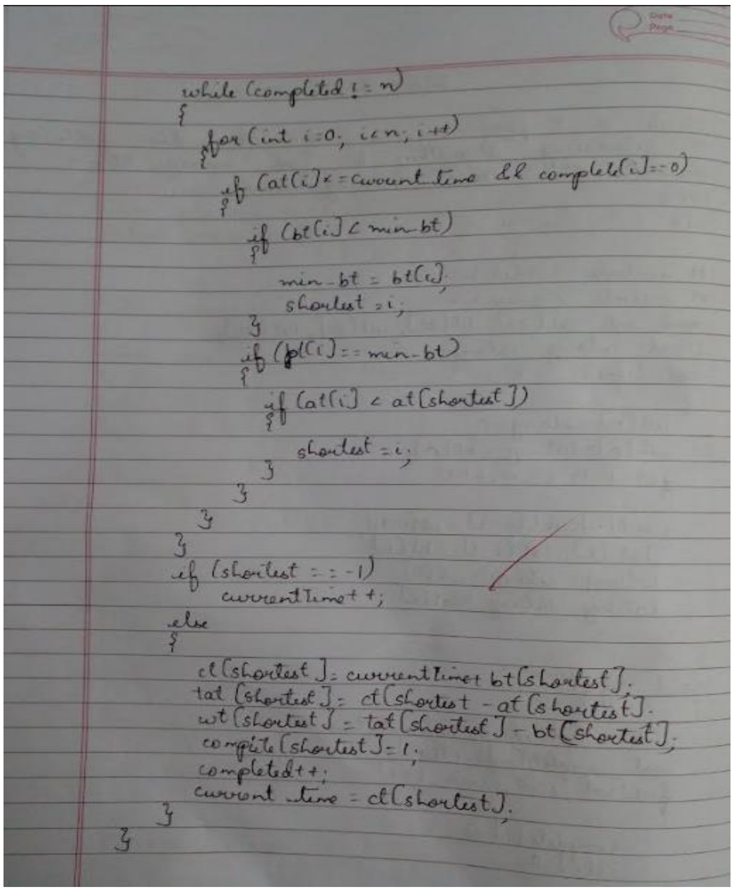
21/6/25

Week 1

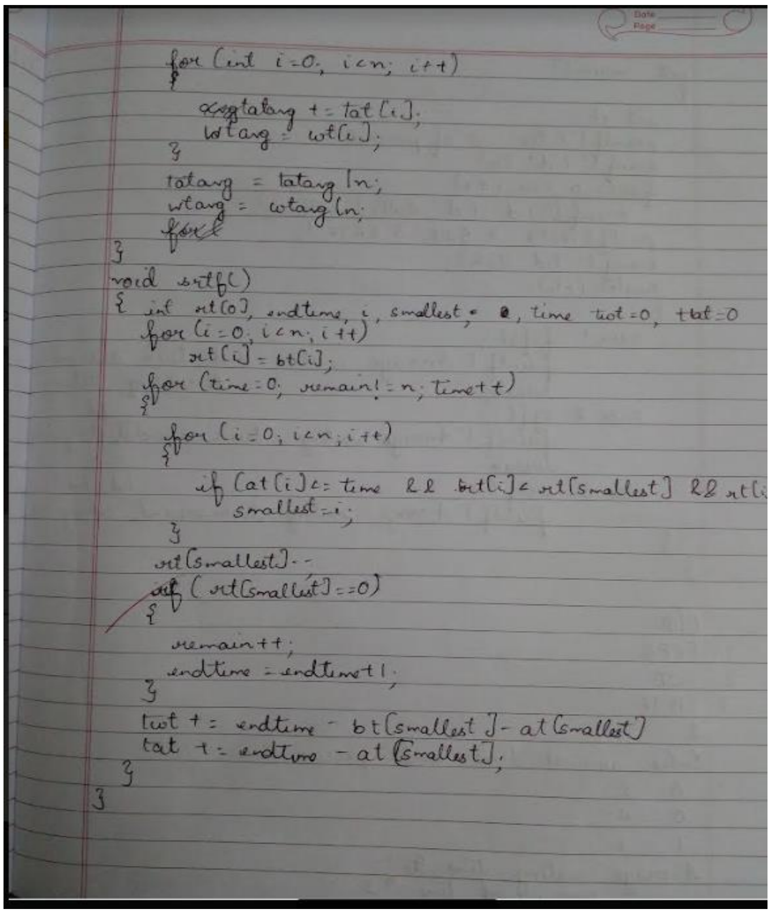
Write a C program to simulate the following CPU scheduling algorithm to find average turn around time and waiting time.

- FCFS
- SJF

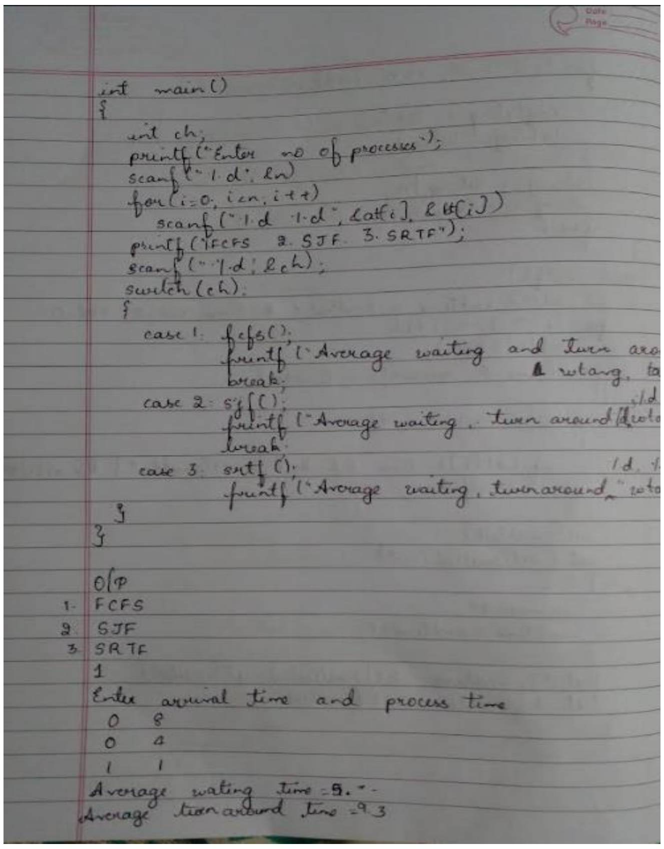
```
# include <stdio.h>
# include <conio.h>
~~void~~ int at[20], bt[20], wt[20], tat[20];
float wtavg tatavg;
void fcfs()
{
   wt[0]= wtavg=0;
   tat[0]= tatavg = bt[0];
   for(i=1; i<n; i++)
   {
     wt[i]= wt[i-1] +bt[i-1];
     tat [i]= tat [i-1] +bt[i];
     wtavg= wtavg+wt[i];
     tatavg= tatavg +tat[i];
   }
}
void sjf()
{
  int completed=0, current_time=0;
  int complete[n], ct[n];
  for(int i=0; i<n; i++)
  {
       complete[i]=0;
       ct[i]=0;
  }
```



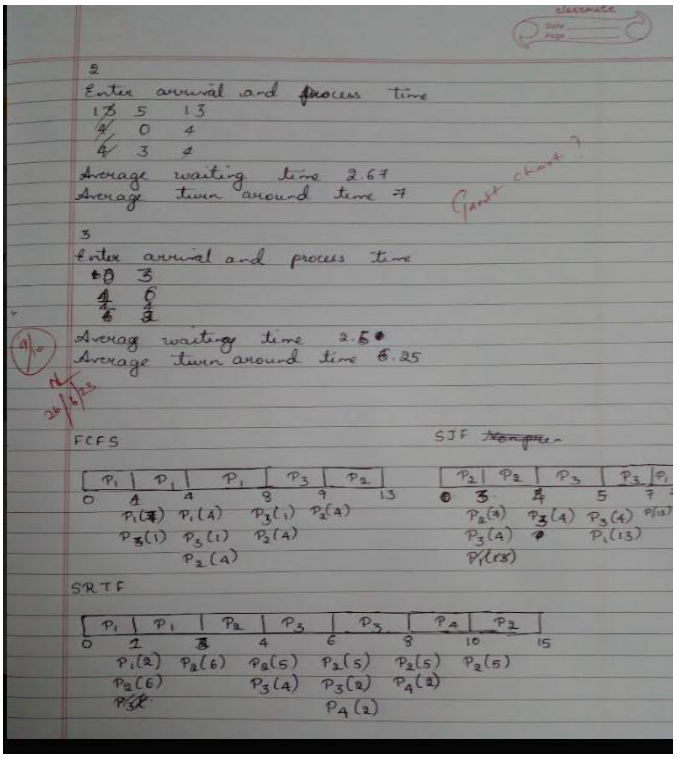
```
        while (completed != n)
        {
          for(int i=0; i<n; i++)
          {
            if (at[i]<=current_time && complete[i]==0)
            {
              if (bt[i] < min_bt)
              {
                min_bt = bt[i];
                shortest =i;
              }
              if (bt[i]== min_bt)
              {
                if (at[i] < at[shortest])
                {
                  shortest =i;
                }
              }
            }
          }
        if (shortest ==-1)
           currentTime++;
        else
        {
          ct[shortest]= currentTime+ bt[shortest];
          tat [shortest]= ct[shortest - at[shortest];
          wt[shortest] = tat[shortest] - bt[shortest];
          complete[shortest]=1;
          completed++;
          current_time = ct[shortest];
        }
     }
```



```
for (int i=0; i<n; i++)
{
    ~~avg~~tatavg += tat[i];
    wtavg = wt[i];
}
tatavg = tatavg/n;
wtavg = wtavg/n;
~~for~~
}
void srtf()
{ int rt[0], endtime, i, smallest = 0, time tot=0, ttat=0
  for (i=0; i<n; i++)
      rt[i] = bt[i];
  for (time=0; remain!=n; time++)
  {
    for (i=0; i<n; i++)
    {
        if (at[i]<=time && bt[i]< rt[smallest] && rt[i
            smallest=i;
    }
    rt[smallest]--
    if ( rt[smallest]==0)
    {
      remain++;
      endtime =endtime+1;
    }
    twt += endtime - bt[smallest]- at[smallest]
    tat += endtime - at[smallest];
  }
}
```



```
int main()
{
   int ch;
   printf("Enter no of processes");
   scanf("%d", &n)
   for(i=0; i<n; i++)
      scanf("%d %d", &at[i], &bt[i])
   printf("1.FCFS 2. SJF 3. SRTF");
   scanf("%d", &ch);
   switch (ch);
   {
     case 1: fcfs();
             printf("Average waiting and turn aro
                                   wtavg, ta
             break;
     case 2: sjf();                          %d
             printf("Average waiting , turn around %d" wto
             break;
     case 3: srtf();                         %d %
             printf("Average waiting , turnaround " wto
   }
}

O/P
1. FCFS
2  SJF
3  SRTF
   1
   Enter arrival time and process time
    0   8
    0   4
    1   1
   Average wating time =5.
  Average turnaround time =9.3
```



2
Enter arrival and ~~process~~ time
1~~3~~ 5 13
~~4~~ 0 4
~~4~~ 3 4
Average waiting time 2.67
Average turn around time 7

Gantt chart ?

3
Enter arrival and process time
0 3
4 6
6 2
Average waiting time 2.5
Average turn around time 6.25

9/10
26/6/23

FCFS

SJF ~~Nonpree~~

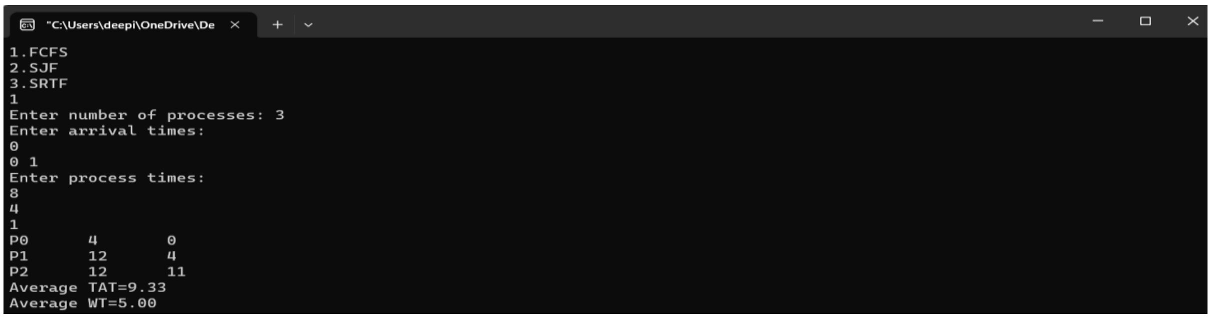
| P1 | P1 | P1 | P3 | P2 |
0 4 4 8 9 13
P1(4) P1(4) P3(1) P2(4)
P3(1) P3(1) P2(4)
P2(4)

| P2 | P2 | P3 | P3 | P1
0 3 4 5 7
P2(4) P3(4) P3(4) P(13)
P3(4) P1(13)
P1(13)

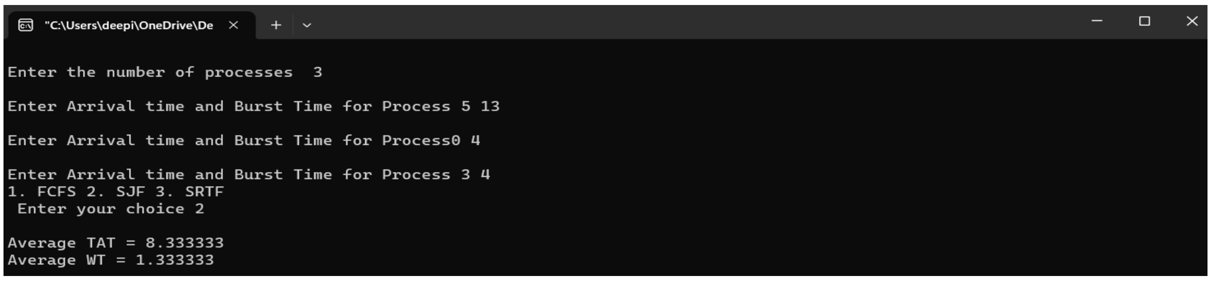
SRTF

| P1 | P1 | P2 | P3 | P3 | P4 | P2 |
0 1 3 4 6 8 10 15
P1(2) P2(6) P2(5) P2(5) P2(5) P2(5)
P2(6) P3(4) P3(2) P4(2)
~~P3~~ P4(2)

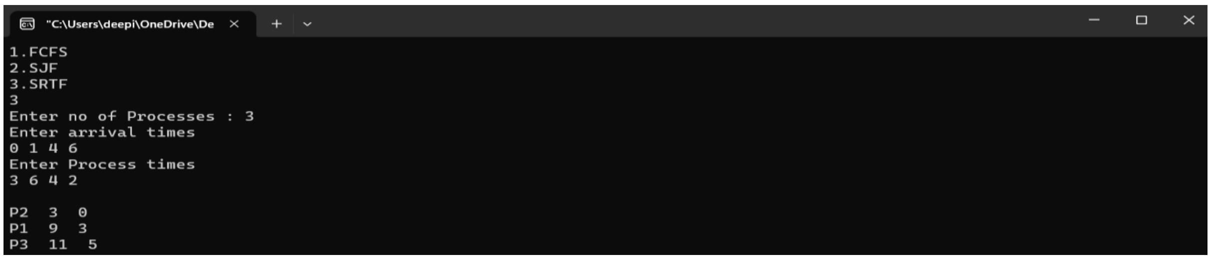
OUTPUT :



```
1.FCFS
2.SJF
3.SRTF
1
Enter number of processes: 3
Enter arrival times:
0
0 1
Enter process times:
8
4
1
P0      4       0
P1      12      4
P2      12      11
Average TAT=9.33
Average WT=5.00
```



```

Enter the number of processes  3

Enter Arrival time and Burst Time for Process 5 13

Enter Arrival time and Burst Time for Process0 4

Enter Arrival time and Burst Time for Process 3 4
1. FCFS 2. SJF 3. SRTF
 Enter your choice 2

Average TAT = 8.333333
Average WT = 1.333333
```



```
1.FCFS
2.SJF
3.SRTF
3
Enter no of Processes : 3
Enter arrival times
0 1 4 6
Enter Process times
3 6 4 2

P2  3  0
P1  9  3
P3  11  5
```