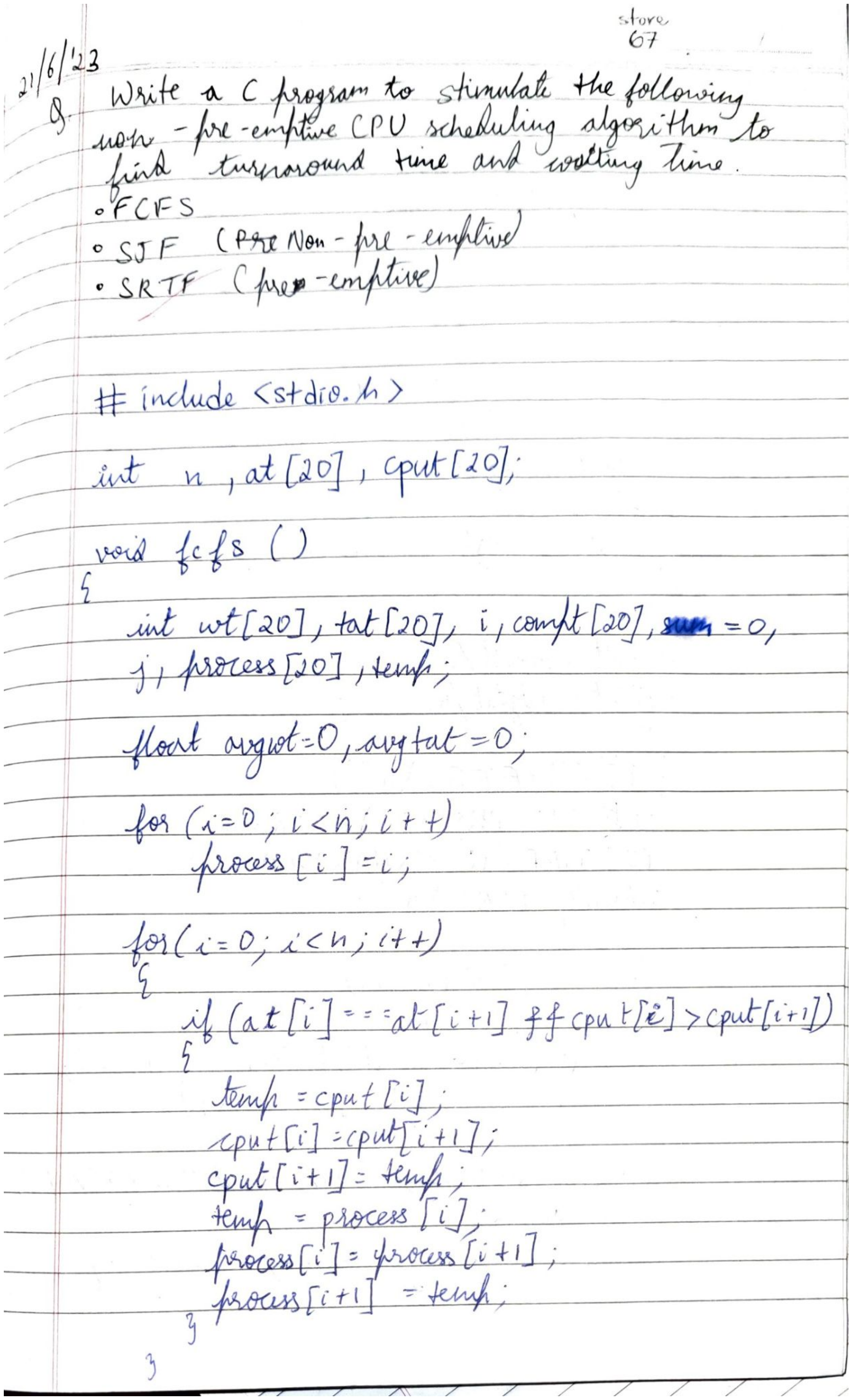
Lab 2: Write a C program to simulate the following non-pre-emptive CPU scheduling algorithm to find turnaround time and waiting time.
→FCFS
→ SJF (pre-emptive & Non-pre-emptive)



store
67

21/6/'23

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

- FCFS
- SJF (~~Pre~~ Non-pre-emptive)
- SRTF (pre-emptive)

```
#include <stdio.h>

int n, at[20], cput[20];

void fcfs ()
{
    int wt[20], tat[20], i, compt[20], ~~sum~~ =0,
    j, process[20], temp;

    float avgwt=0, avgtat=0;

    for (i=0; i<n; i++)
        process[i]=i;

    for(i=0; i<n; i++)
    {
        if (at[i]===at[i+1] && cput[i]>cput[i+1])
        {
            temp = cput[i];
            cput[i]=cput[i+1];
            cput[i+1]= temp;
            temp = process[i];
            process[i]= process[i+1];
            process[i+1] = temp;
        }
    }
    for (i=0; i<n; i++)
    {
        sum + =cput [i];
        comp[i] =sum;
    }

    for (i =0; i<n; i++)
    {
      tat [i] = comp [i] -at [i];
      wt [i] = tat [i] - cput [i];
      avgwt = avgwt + wt [i];
      ~~avgwt =~~
      avgtat = avgtat + tat[i];
    }

  avg wt = avgwt / n;
  avgtat = avgtat / n;

  printf ("\n FCFS : \n");
  printf ("\t PROCESS \t ARRIVAL TIME \t
  CPU TIME \t TURNAROUND TIME \t
  WAITING TIME \n");

  for (i=0; i<n; i++)
      printf ("\n \t P%d \t \t %d \t \t%d
      \t \t %d \t \t \t %d", i, at[i],
      cput [i], tat [i], wt [i]);

  printf (" \n Average Turnaround Time : %f",
      avgtat);
  printf ("\n Average Waiting Time : %f",
      avgwt);
}
```

store
67

```
void sjfnp()
{
   int tat[20], wt[20], cputd[20];
   float avgwt =0, avgtat =0, sum_cpu_time = 0;
   int time =0, i, j , smallest, temp2;
   for (i =0; i<n; i++)
    { sum_cpu_time += cput[i];
      cputd[~~20~~i] = cput[i];   }
   cputd[9] = 9999;
   printf ("\n SJF :\n");

   while (time < sum_cpu_time)
   {
     smallest =9;
     for (i =0; i<n; i++)
     {
        if (at[i] <= time && cputd[i] >0 &&
             cputd[i] < cputd[smallest])
          smallest = i;
     }

   printf ("\t ARRIVAL TIME \t CPU TIME \t
    TURNAROUND TIME \t WAITING TIME \n");
   printf ("P[%d] \t \t %d \t \t %d \t \t %d
        \t \t %d \n", smallest, at[smallest],
    cputd[smallest], time + cputd[smallest] -
    at[smallest], time - at[smallest]);
    avgtat += time + cputd[smallest] - at[smallest];
    avgwt += time - at[smallest];
     time += cputd[smallest];
     cputd[smallest] =0;
 }
```

```
  avgwt = avgwt/n;
  avgtat = avgtat/n;

  printf("\n Average Turnaround time :- %f",
            avgtat);
  printf("\n Average Waiting Time :- %f",
            avgwt);
 }

void srtf()
{
   int remaining_time[20], tat[20], wt[20],
   completion_time[20], smallest, time=0, i,
   count=0;
   float avgwt=0, avgtat=0;

   for(i=0; i<n; i++)
       remaining_time[i] = cput[i];

   while(count != n)
   {
     smallest = -1;
     for(i=0; i<n; i++)
     {
       if (at[i] <= time && remaining_time ~~[i]>0)~~
                                  [i] > 0)
       {
         if (smallest == -1 || remaining_time[i]
                        < remaining_time[smallest])
            smallest = i;
       }
   }
 if (smallest == -1)
 {
   time ++;
   continue;
 }

 remainig_time[smallest]--;

 if (remaining_time [smallest]==0)
 {
   count ++;
   completion_time [smallest] = time +1;
   wt[smallest] = completion_time [smallest]
   - at[smallest] - cput[smallest];
   tat [smallest] = completion_time [smallest]
                              - at[smallest];
 }
 time ++;
}

for (i=0; i<n; i++)
 {
   avgwt+= wt [i];
   avgtat += tat [i];
 }
 avgwt = avg wt/n;
 avgtat = ~~avgwt~~ avgtat /n;

 printf ("\n Process \t Arival Time \t CPU
Time \t Waiting Time \t Turnaround Time \n");
 for( i=0; i<n; i++)
     printf("%d \t %d \t \t %d \t \t %d \t \t %d\n
        i, at[i], cput[i], wt[i], tat[i]);
    printf("\n Average Turnaround Time : %f",
                                    avgtat);
    printf("\n Average Waiting Time : %f",
                                    avgwt);
}

void main()
{
    int i, ch;
    printf(" \n Enter the number of processes:"
                                            );
    scanf("%d", &n);

    for(i=0; i<n; i++)
    {
      printf("\n Enter Arrival Time and
      CPU Time for the Process %d :", i);
      scanf("%d %d, &at[i], &cput[i]);
    }

    while(1)
    {
      printf("\n MENU \n");
      printf("1. FCFS \n 2. SJF \n 3. SRTF \n
          4. Exit \n");
      scanf("%d", &ch);

      switch(ch)
      {
       case1: fcfs();
```

```
            break;
      case2 : sjfnp();
              break;
      case 3 : srtf();
               break;
      case 4 : exit(0);
      default : printf("\nWrong Choice !! Try
                  Again");
   }
  }
}
```

Output:

```
Enter the numbers of processes: 4

Enter Arival Time and CPU Time.

   0   3
   1   6
   4   4
   6   3

 MENU
1. FCFS
2. SJF
3. SRTF
4. Exit
```

1 ~~Gantt chart~~:

FCFS :

Process	Arrival time	CPU time	TAT	WT
P0	0	3	3	0
P1	1	6	8	2
P2	4	4	9	5
P3	6	2	9	7

Average Turnaround Time = 7.250000
Average Waiting Time : 3.500000

2

~~SF~~SJF:

	Arrival time	CPU time	TAT	WT
P[0]	0	3	3	0
P[1]	1	6	8	2
P[3]	6	2	5	3
P[2]	4	4	11	7

Average TAT: 6.75
Average WT: 3

3

Process	Arrival Time	CPU Time	~~WT~~WT	TAT
0	0	3	0	3
1	1	6	8	14
2	4	4	0	4
3	6	2	2	4

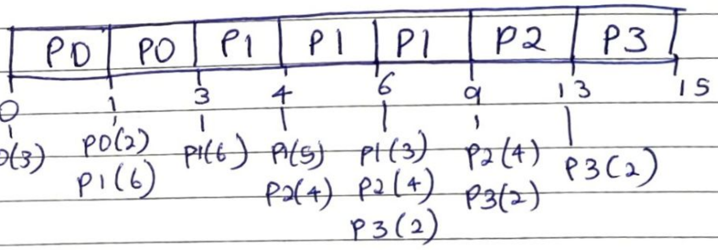
Average TAT = 6.25
Average WT = 2.5

4 //To exit.

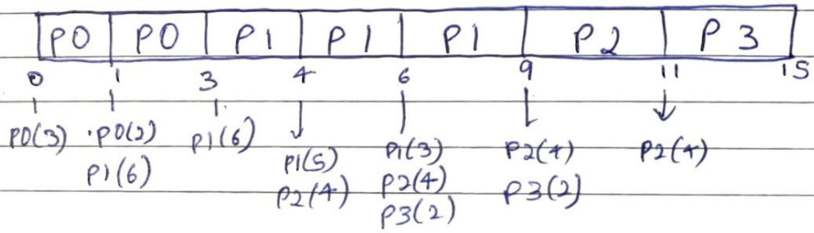
21/6/23

Gantt charts :-

FCFS:



SJF:



SRTF:

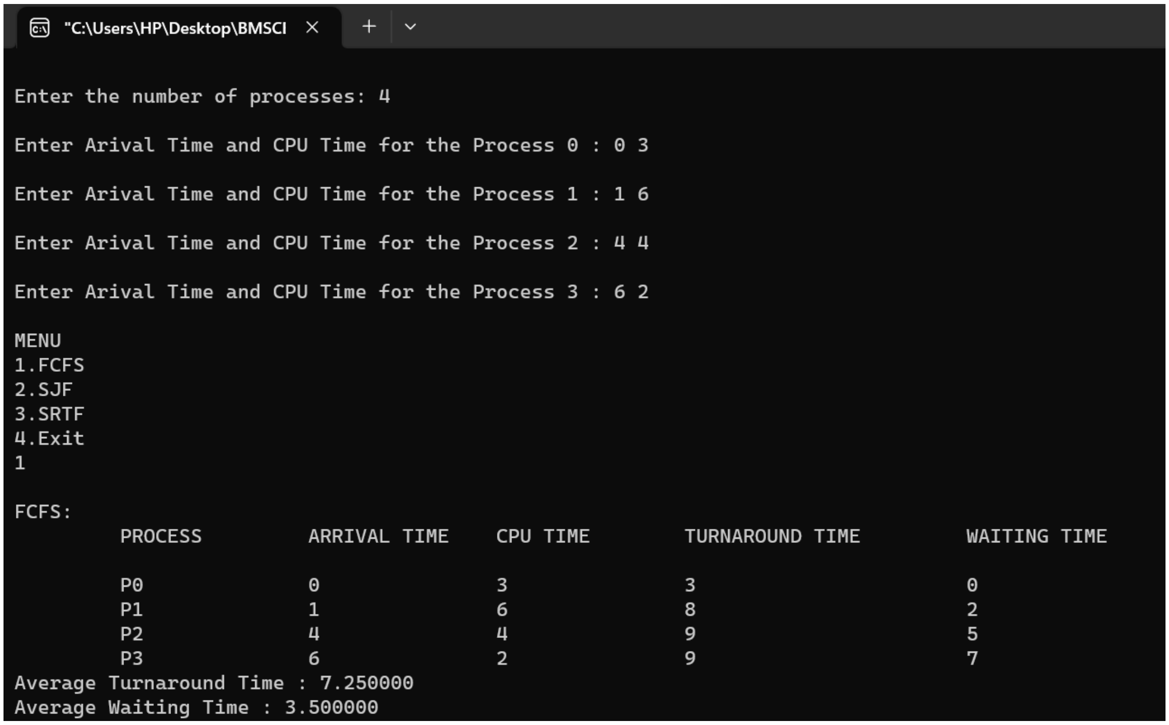
PO	PO	P1	P2	P2	P3	P1

0 1 3 4 6 8 10 15

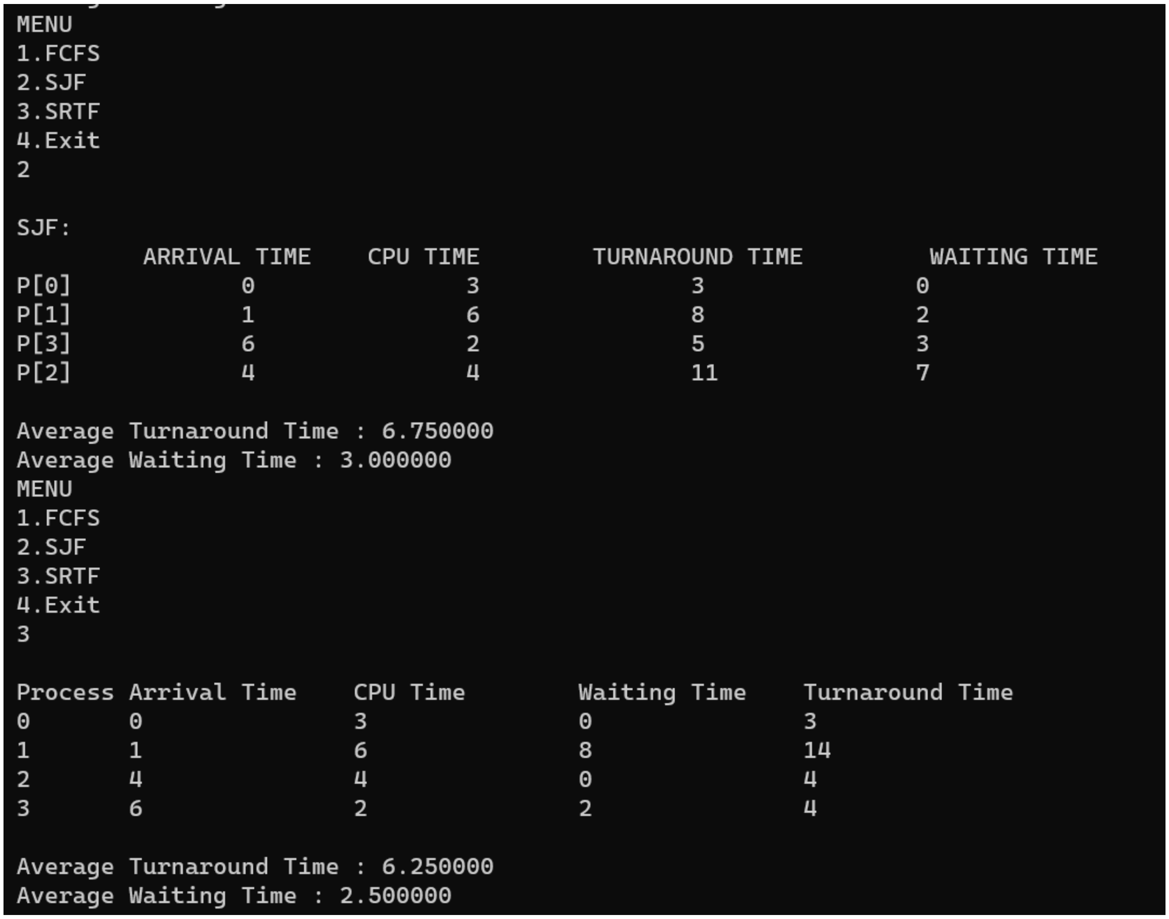
0	1	3	4	6	8	10

PO(3)	PO(2)
	P1(6)
		P1(6)	P1(5)
			P2(4)
				P2(2)
				P1(5)
				P3(2)
					P1(5)
					P3(2
						P1(5)

OUTPUT:



```
Enter the number of processes: 4

Enter Arival Time and CPU Time for the Process 0 : 0 3

Enter Arival Time and CPU Time for the Process 1 : 1 6

Enter Arival Time and CPU Time for the Process 2 : 4 4

Enter Arival Time and CPU Time for the Process 3 : 6 2

MENU
1.FCFS
2.SJF
3.SRTF
4.Exit
1

FCFS:
         PROCESS         ARRIVAL TIME    CPU TIME        TURNAROUND TIME         WAITING TIME

         P0              0               3               3                       0
         P1              1               6               8                       2
         P2              4               4               9                       5
         P3              6               2               9                       7
Average Turnaround Time : 7.250000
Average Waiting Time : 3.500000
```



```
MENU
1.FCFS
2.SJF
3.SRTF
4.Exit
2

SJF:
         ARRIVAL TIME    CPU TIME        TURNAROUND TIME         WAITING TIME
P[0]            0               3               3               0
P[1]            1               6               8               2
P[3]            6               2               5               3
P[2]            4               4               11              7

Average Turnaround Time : 6.750000
Average Waiting Time : 3.000000
MENU
1.FCFS
2.SJF
3.SRTF
4.Exit
3

Process Arrival Time    CPU Time        Waiting Time    Turnaround Time
0       0               3               0               3
1       1               6               8               14
2       4               4               0               4
3       6               2               2               4

Average Turnaround Time : 6.250000
Average Waiting Time : 2.500000
```