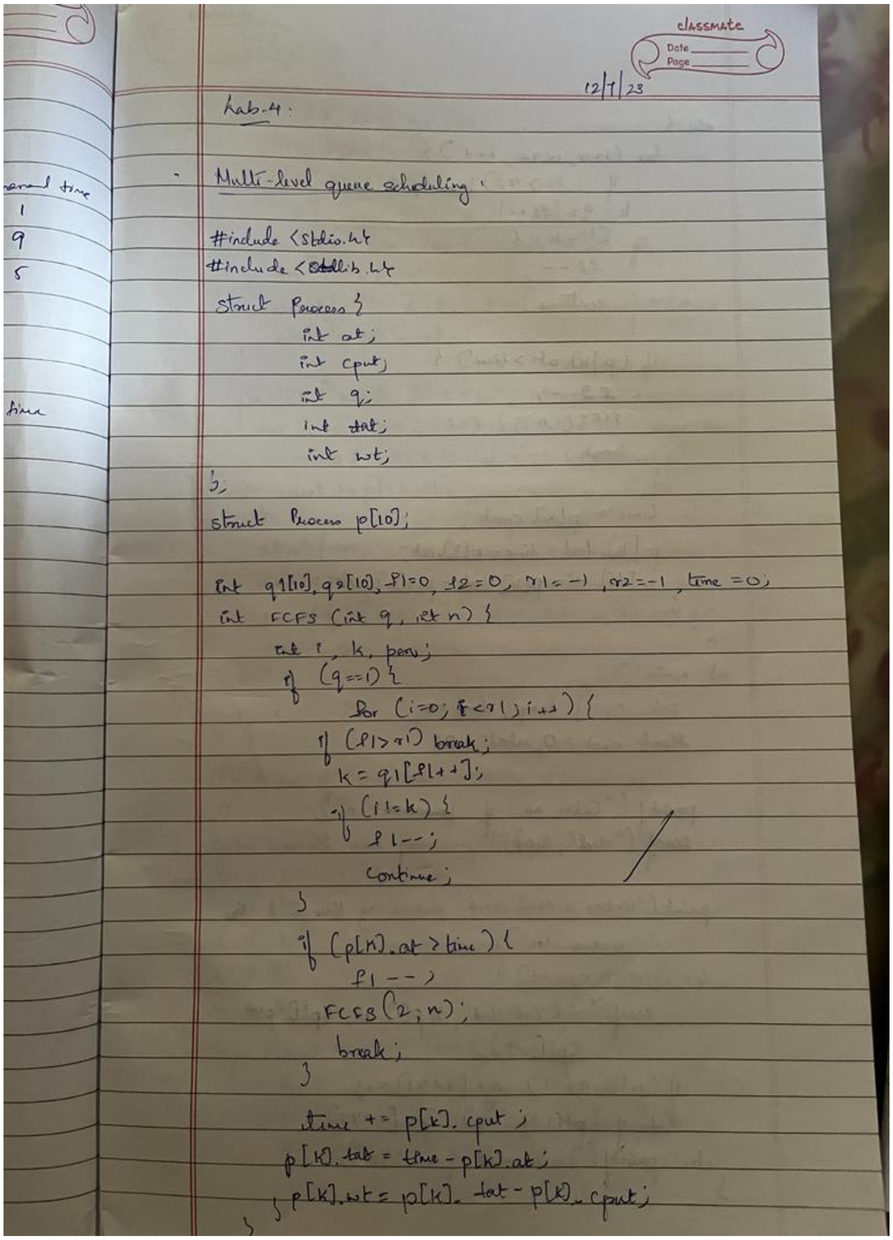
LAB 3

Write a C program to simulate multi-level queue scheduling algorithm considering the following scenario. All the processes in the system are divided into two categories ± system processes and user processes. System processes are to be given higher priority than user processes. Use FCFS scheduling for the processes in each queue.



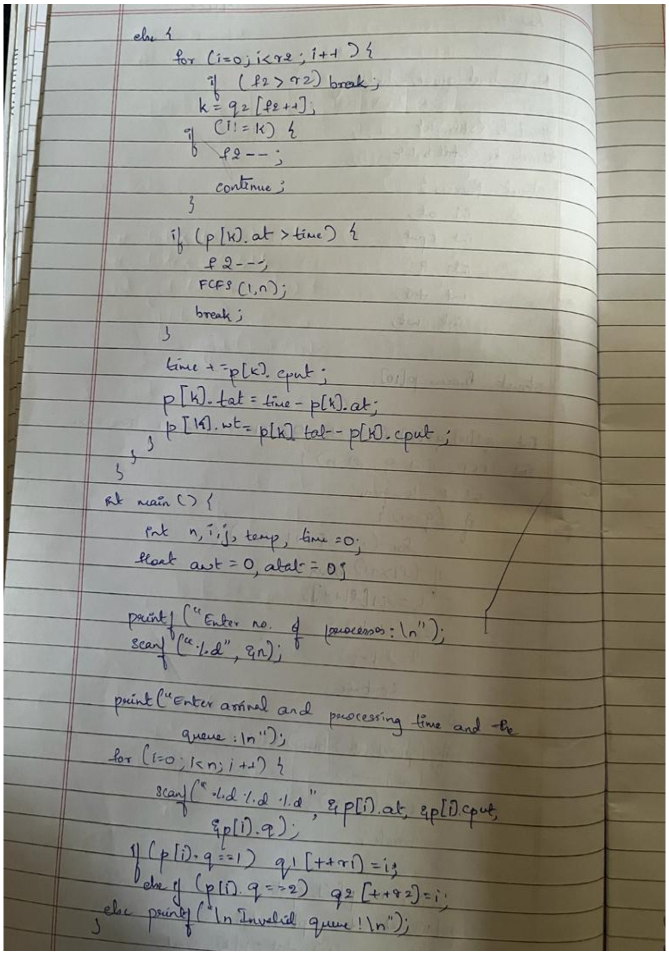
classmate
Date
Page

12/7/23

Lab-4:

Multi-level queue scheduling:

```
#include <Stdio.h>
#include <Stdlib.h>
struct Process {
        int at;
        int cput;
        int q;
        int tat;
        int wt;
};
struct Process p[10];

int q1[10], q2[10], f1=0, f2=0, r1=-1, r2=-1, time =0;
int FCFS (int q, int n) {
    int i, k, prev;
    if (q==1) {
          for (i=0; i<r1; i++) {
       if (f1>r1) break;
        k = q1[f1++];
        if (i!=k) {
          f1--;
          continue;
    }
    if (p[k].at > time) {
         f1--;
       FCFS(2, n);
        break;
    }
    time += p[k]. cput;
   p[k]. tat = time - p[k].at;
  } p[k].wt = p[k]. tat - p[k]. cput;
}
```



```
else {
    for (i=0; i<r2; i++) {
        if (f2 > r2) break;
        k = q2 [f2++];
        if (i!= k) {
            f2--;
            continue;
        }
        if (p[k].at > time) {
            f2--;
            FCFS (1,n);
            break;
        }
        time += p[k]. cput;
        p[k]. tat = time - p[k].at;
        p[k]. wt = p[k] tat - p[k]. cput;
    }
  }
}
int main () {
    int n, i, j, temp, time =0;
    float awt = 0, atat = 0;

    printf ("Enter no. of processes : \n");
    scanf ("%d", &n);

    print ("Enter arrival and processing time and the
            queue : \n");
    for (i=0; i<n; i++) {
        scanf (" %d %d %d", &p[i].at, &p[i].cput,
               &p[i].q);
        if (p[i].q==1) q1 [++r1] = i;
        else if (p[i]. q == 2)  q2 [++r2] = i;
    else printf ("\n Invalid queue ! \n");
}
```

```
while (f1<=r1 && f2<=r2) {
        FCFS(1,n);
        FCFS(2,n);
  }

  printf ("Turn around time : \n");
   for (i=0; i<n; i++) {
   printf ("P%d \t %d\n", i+1, p[i].tat)
   atat += p[i].tat;
 }

printf ("wait time : \n");
 for (i=0; i<n; i++) {
     printf ("P%d \t %d\n", i+1, p[i].wt);
    awt += p[i].wt;
  }
  atat /=n;
  awt/= n;
  printf ("\n ATAT = %f   AWT = %f", atat, awt);
  return 0;
}
```

Output:

```
Enter no. of processes :
5
Enter arrival and processing time and the queue:
0 5 1
1 3 1
2 7 2
3 2 1
4 4 2

     Turn around time        Wait time
P1        5
P2                              0
          6                     3
P3        11
P4                              4
          4
P5                              1
          8                     4
```

ATAT= 6.8000

AWT = 2.4000

```
| P1 | P2 | P4 | P3 | P5 | P1 | P3 | P5 |
0    5    8    11   13   17   21   25   29
```

→ Rate Monotonic:

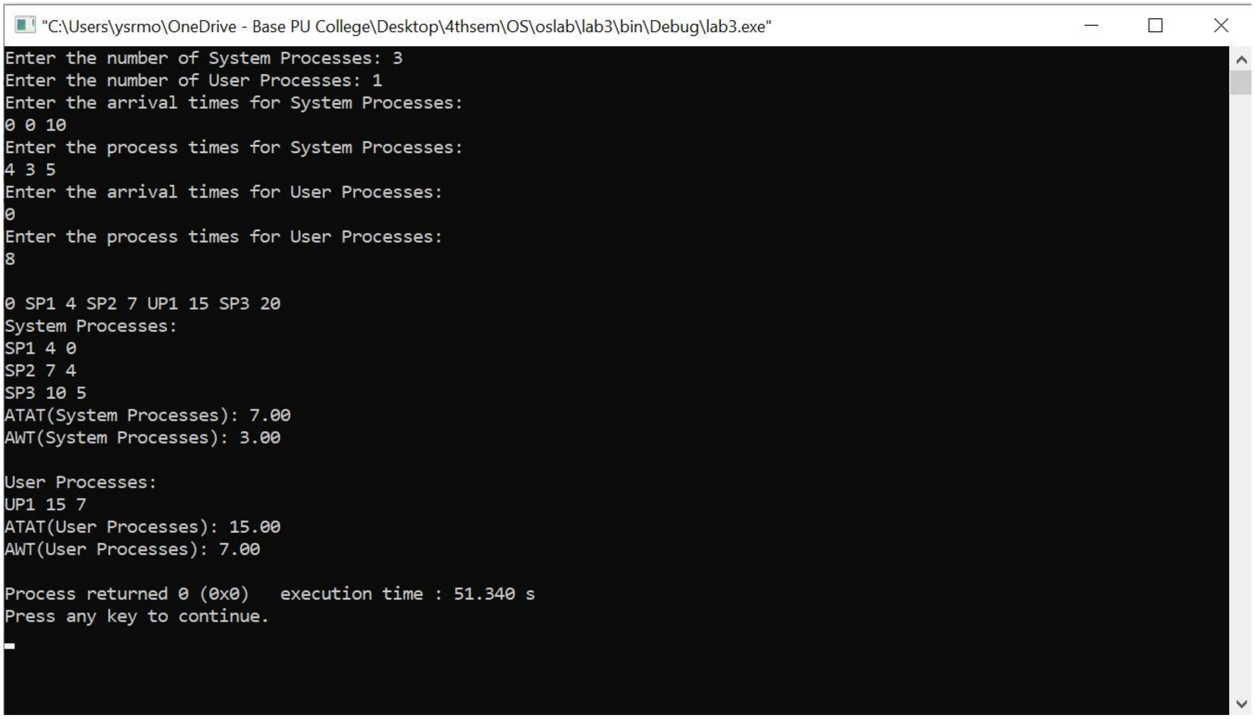
```
#include<stdio.h>
#include<stdlib.h>
int gcd (int a, int b) {
    if (b==0) return a;
    else gcd (b,a%b);
}

int  lcm (int a, int b){
    return ((a*b) / gcd (a,b));
}
int hyperperiod (int period [], int n) {
    int k= period[0];
    n--;
    while (n>=1) {
        k=lcm (k, period[n--]);
    }
    return k;
}

int main () {
    int period[10], cput[10], n, i, temp, hyp;
    printf("Enter no. of processes:\n");
    scanf ("%d", &n);

printf ("Enter period and processing time :\n");
for (i= 0; i<n; i++) {
    scanf ("%d %d", &period [i], &cput [i]);
}
hyp = hyperperiod (period, n);
printf ("Hyperperiod = %d \n", hyp);
int util [n];
float utilization = 0;
for (i=0; i<n; i++) {
    util [i]= hyp/period [i];
    util [i] *= cput [i];
    utilization += util [i];
    printf ("util %d: %d\n", i+1, util [i]);
  }
  utilization /= hyp;
  printf ("utilization = %f \n", utilization);
  return 0;
}
```

Output:

```
Enter no. of process : 3
Enter period & processing time:
6   2
8   3
10  2
Hyper period = 120
util 1: 40
util 2: 45
util 3: 20
utilization = 0.908333
```

Late
19/7/2023

OUTPUT:


"C:\Users\ysrmo\OneDrive - Base PU College\Desktop\4thsem\OS\oslab\lab3\bin\Debug\lab3.exe"

```
Enter the number of System Processes: 3
Enter the number of User Processes: 1
Enter the arrival times for System Processes:
0 0 10
Enter the process times for System Processes:
4 3 5
Enter the arrival times for User Processes:
0
Enter the process times for User Processes:
8

0 SP1 4 SP2 7 UP1 15 SP3 20
System Processes:
SP1 4 0
SP2 7 4
SP3 10 5
ATAT(System Processes): 7.00
AWT(System Processes): 3.00

User Processes:
UP1 15 7
ATAT(User Processes): 15.00
AWT(User Processes): 7.00

Process returned 0 (0x0)   execution time : 51.340 s
Press any key to continue.
```