WRITE A PROGRAMME TO STIMULATE THE MULTI-SCHEDULING ALGORITHM USING `C` LANGUAGE….

```
#include<stdio.h>
struct proc {
   int id, at, bt, rt, no;
   int ct, tat, wt;
};
int tat_total=0, wt_total=0;

void sort(struct proc p[], int n) {
  struct  proc temp;
   for (int i = 0; i < n ; i++) {
      for (int j = i+1; j < n; j++) {
         if (p[j].at < p[i].at) {
            temp = p[j];
            p[j] = p[i];
            p[i] = temp;
         }
      }
   }
}

void display(struct  proc p[], int n)
{
   for (int i = 0; i < n; i++) {
      printf("%d\t%d\t%d\t%d\t\t%d\t%d\n", p[i].id, p[i].at,
            p[i].bt,p[i].ct,
            p[i].tat, p[i].wt);
}
}
```

```
void cal(struct proc p[], int n)
{
   for (int i = 0; i < n; i++)
   {
      p[i].tat= p[i].ct - p[i].at;
      p[i].wt = p[i].tat - p[i].bt;
      tat_total+=p[i].tat;
      wt_total+=p[i].wt;
   }
}

void fcfs(struct proc p[], int m)
{   sort(p,m);
   int time=0;
   for(int i=0;i<m;i++)
   {
      if(time<p[i].at)
         time=p[i].at;
      time+=p[i].bt;
      p[i].ct=time;
   }
   cal(p,m);
}


void roundrobin(struct proc p[], int m, int q) {
   sort(p,m);
   int time = 0, completed = 0;
```

```
    while (completed < m) {
        int found = 0;
        for (int i = 0; i < m; i++) {
            if (p[i].at <= time && p[i].rt > 0) {
                found = 1;

                if (p[i].rt > q) {
                    time += q;
                    p[i].rt -= q;
                } else {
                    time += p[i].rt;
                    p[i].ct = time;
                    p[i].rt = 0;
                    completed++;
                }
            }
        }

        if (!found) time++;
    }

    cal(p,m);
}

int main() {
    int n,q;
    int j=0,k=0;

    printf("Enter number of processes: ");
    scanf("%d", &n);
```

```
struct proc pi[n], system[n], user[n];



for (int i = 0; i < n; i++) {
   pi[i].id = i + 1;
   printf("Enter details for Process %d:\n", pi[i].id);
   printf("Arrival Time: ");
   scanf("%d", &pi[i].at);
   printf("Burst Time: ");
   scanf("%d", &pi[i].bt);
   printf("Queue No.(System-1, user-0):");
   scanf("%d", &pi[i].no);
   pi[i].rt = pi[i].bt;

   if(pi[i].no==1)
   {
      system[j]=pi[i];
      j++;
   }
   if(pi[i].no==0)
   {
      user[k]=pi[i];
      k++;
   }
}

tat_total = 0;
wt_total = 0;

q=3;
```

```
    fcfs(system, j);
    fcfs(user,k);
    printf("\nProcess\tArrival\tBurst\tCompletion\tTAT\tWaiting\n");



    display(system,j);
    display(user,k);

    printf("Average waiting time is %f", (float)wt_total/(j+k));
    printf("\nAverage turn around time time is %f", (float)tat_total/(j+k));



    return 0;
}
```

OUTPUT:

1.When Both are FCFS

```
Burst Time: 5
Queue No.(System-1, user-0):0
Enter details for Process 2:
Arrival Time: 3
Burst Time: 1
Queue No.(System-1, user-0):1
Enter details for Process 3:
Arrival Time: 2
Burst Time: 5
Queue No.(System-1, user-0):1
Enter details for Process 4:
Arrival Time: 3
Burst Time: 2
Queue No.(System-1, user-0):0
Enter details for Process 5:
Arrival Time: 6
Burst Time: 4
Queue No.(System-1, user-0):0

Process Arrival Burst   Completion      TAT     Waiting
3       2       5       7               5       0
2       3       1       8               5       4
1       0       5       5               5       0
4       3       2       7               4       2
5       6       4       11              5       1
Average waiting time is 1.400000
Average turn around time time is 4.800000
Process returned 0 (0x0)   execution time : 30.381 s
Press any key to continue.
```

2.When one is Round robin and other is FCFS

```
Burst Time: 5
Queue No.(System-1, user-0):0
Enter details for Process 2:
Arrival Time: 3
Burst Time: 1
Queue No.(System-1, user-0):1
Enter details for Process 3:
Arrival Time: 2
Burst Time: 5
Queue No.(System-1, user-0):1
Enter details for Process 4:
Arrival Time: 3
Burst Time: 2
Queue No.(System-1, user-0):0
Enter details for Process 5:
Arrival Time: 6
Burst Time: 4
Queue No.(System-1, user-0):0

Process Arrival Burst   Completion      TAT     Waiting
3       2       5       8               6       1
2       3       1       6               3       2
1       0       5       5               5       0
4       3       2       7               4       2
5       6       4       11              5       1
Average waiting time is 1.200000
Average turn around time time is 4.600000
Process returned 0 (0x0)   execution time : 65.713 s
Press any key to continue.
```

Therefore we conclude that it is berrter to use the combination of round-robin and fcfs in multilevel Queue scheduling…