Aim:

To Implement the following CPU Algorithms Using C

1.Round Robin

2.Priority

Alogorithm:

1.RoundRobin:

1.Get all inputs required

2. Maintain a time quantum and reduce BT with tq

3.Print GANTT Chart as we iterate

4.Maintain another list where the ct gets updated

5.Exit the loop once all BT=0

6.Print Average for the required Parameters

2.Priority (Non-Premptive):

1.Get all required inputs

2.iterate over all processes and choose the process with highest priority in the arrival queue

3.Once chosen Simultaneously print the completion time

4.tt is added with bt of existing instance of process running

5.Exit once we finish iterating the array

6. Print Average for the required Parameters

Code:

RoundRobin.c

```
#include <stdio.h>
int is_all_zero(int arr[][3],int len){
    int flag=0;
    for(int i=0;i<len;i++){
        if(arr[i][2]<=0){
            flag++;
        }
    }
```

```
        if(flag>=len){
                return 1;
        }
        return 0;
}
void swap(int *a ,int *b){
        int tmp=*a;
        *a=*b;
        *b=tmp;
}
void sort(int proc[][3],int len){
        int tmp_pid,tmp_at,tmp_bt;
        for(int i=0;i<len;i++){
                for(int j=0;j<len-i-1;j++){
                        if(proc[j][1]>proc[j+1][1]){

                                swap(&proc[j][0],&proc[j+1][0]);
                                swap(&proc[j][1],&proc[j+1][1]);
                                swap(&proc[j][2],&proc[j+1][2]);

                        }
                }
        }

}

int main(){
        int pid,tq;
        printf("Enter NO of procs and time quantum: ");
        scanf("%d %d",&pid,&tq);
```

```
int vis[pid*2],ctr=0;
int et=0,at,bt,pc_no,i=0;
int ct[pid];
int proctable[pid][3],dup[pid][3];
printf("Enter pid,At,BT\n");
for(int i=0;i<pid;i++){
        scanf("%d %d %d",&pc_no,&at,&bt);
        proctable[i][0]=pc_no;
        proctable[i][1]=at;
        proctable[i][2]=bt;
        dup[i][0]=pc_no;
        dup[i][1]=at;
        dup[i][2]=bt;
}
printf("\ndone\n");
sort(proctable,pid);
sort(dup,pid);
while (1){
        if(is_all_zero(proctable,pid)==1){
                break;
        }
        if(i>=pid){
                i=0;
                continue;
        }
        if(proctable[i][2]<=0){
                i++;
                continue;
        }
```

```
            if(et+tq>=proctable[i][1]){
                vis[ctr++]=proctable[i][0];

                proctable[i][2]-=tq;
                et+=tq;
                ct[i]=et;
                i++;

            }
            else{
                i=0;
                vis[ctr++]=proctable[i][0];
                //printf("%d\t",proctable[i][0]);
                //et+=proctable[i][1];
                ct[i]=et;
                i++;
            }

        }
        float tat,wt;
        for(int i=0;i<pid;i++){
            printf("%d\t %d \t %d \n",proctable[i][0],ct[i]-proctable[i][1],ct[i]-dup[i][2]);
            tat+=ct[i]-proctable[i][1];
            wt+=ct[i]-dup[i][2];
        }
        printf("\nAverage TAT :%.2f\nAverage WT:%.2f",tat/(float)pid,wt/(float)pid);


}
```

Output:

```
root@LAPTOP-FHHEGJQ5:/mnt/e/oslab/ajay21110103/Exercise6# gcc RoundRobin.c
root@LAPTOP-FHHEGJQ5:/mnt/e/oslab/ajay21110103/Exercise6# ./a.out
Enter NO of procs and time quantum: 4 2
Enter pid,At,BT
1 0 5
2 1 4
3 2 2
4 4 1

done
1        14      9
2        11      8
3        4       4
4        4       7

Average TAT :8.25
Average WT:7.00root@LAPTOP-FHHEGJQ5:/mnt/e/oslab/ajay21110103/Exercise6#
```

2.Priority(NonPremptive):

```
#include <stdio.h>


int  main()
{

        int pid,tq;
        printf("Enter NO of procs:");
        scanf("%d",&pid);
        int vis[pid],ctr=0;
        int et=0,at,bt,pc_no;
        int ct[pid],priority;
        int proctable[pid][4],dup[pid][3];
        printf("Enter pid,At,BT,priority\n");
        for(int i=0;i<pid;i++){
                scanf("%d %d %d %d",&pc_no,&at,&bt,&priority);
                proctable[i][0]=pc_no;
                proctable[i][1]=at;
                proctable[i][2]=bt;
                proctable[i][3]=priority;
```

```
        }
        printf("\ndone\n");
        int i=0,tt=0,j,j_pri;
        float tat=0,wt=0;
        for(int i=0;i<pid;i++){
                vis[i]=0;
        }
        for(i=0;i<pid;i++){
           j=-2,j_pri=123123412;
                for(int k=0;k<pid;k++){
                        /*find k index using priorty if not found and iff within total arr time*/
                        if(proctable[k][1]<=tt && !vis[k]){
                                if(proctable[k][3]<j_pri){
                                        j_pri=proctable[k][3];
                                        j=k;
                                }
                        }
                }
                tt+=proctable[i][2];
                printf("%d %d %d %d\n",proctable[i][0],tt,tt-proctable[i][1],tt-
proctable[i][2]);
                tat+=tt-proctable[i][1];
                wt+=tt-proctable[i][2];
                vis[j]=1;
        }
        printf("Average TAT:%.2f\nAverage WT:%.2f\n",tat/(float)pid,wt/(float)pid);

}
```

Output:

```
root@LAPTOP-FHHEGJQ5:/mnt/e/oslab/ajay21110103/Exercise6# ./a.out
Enter NO of procs:5
Enter pid,At,BT,priority
1 0 4 1
2 0 3 2
3 6 7 1
4 11 4 3
5 12 2 2

done
1 4 4 0
2 7 7 4
3 14 8 7
4 18 7 14
5 20 8 18
Average TAT:6.80
Average WT:8.60
root@LAPTOP-FHHEGJQ5:/mnt/e/oslab/ajay21110103/Exercise6#
```

Result:

Thus the Above algorithms were simulated and implemented in C .