2024/05/30
Thrusday

1) Round Robin(Experiment with differenent quantum size of algorithm)

```
#include<stdio.h>
 voidmain()
{ struct proc

      int no,at, bt,ct, tat,wt,rt;
 };

 struct proc read(int i)
 {
     struct proc p.
    printf("\n process No:%d\n",i);
     p.no=i;
     printf("Enter Arrival time: ");
     Scanf("%d", &p.at);
     printf("enter the Bust time:");
      Scanf("%d", &p.bt);
      p.rt=p.bt;
      return p;
   }
   Int main()
   {
      Struct proc p[10],temp;
      Float avgtat=0,avgwt=0;
      int n,tq,ct=0,flag=0,remaning;
      printf("<--RoundRobin Scheduling Algorithm-->\n");
      printf("enter No. of process:");
       Scanf("%d",&n);
      printf(" Enter the quantum:");
       Scanf("%d", &tq);
        For(int i=0; i<n; i++)
           P[i]=read(i+1);
```

```
For(int i=0;i<n-1;i++)
    For(int J=0;J<n-i-1;J++)
        if(p[J].at>p[J+1].at)
        {
            temp=p[J];
            P[J]=p[J+1];
            P[J+1]=temp;
        }
    remaining=n;
    printf("\n processNo\tAT\tBT\tCT\tTAT\twt\n");
    For (int i=0; remaining!=0;)
    {   if( p[i].rt<=tq && p[i].rt>0)
        {ct+=p[i].rt;
          P[i].rt=0;
          Flag=1;
        }
        else if(p[i].rt>0)
          { p[i].rt-=tq;
             ct+=tq;
          }
          if(p[i].rt==0&&flag==1)
          {
             Flag=0;
             remaining--;
             P[i].ct=ct;
             p[i].tat=p[i].ct-p[i].at;
             avgtat+=p[i].tat;
             P[i].wt=p[i].tat-p[i].bt;
             avgwt+=p[i].wt;
             printf("p%d\t\t%d\t%d\t%d\t%d\t%d\n",
    p[i].no,p[i].at,p[i].bt,p[i].ct,p[i].tat,p[i].wt);
  }
    if(i!=n-1&&p[i+1].at<=ct)
            i++;
     else i=0;
    }
    avgtat/=n, avgwt/=n;
     printf("\n Average turnaround time=%f \n Avera
       waiting time=%f", avgtat,avgwt);
```

}

Output

Processor	AT	BT
P1	0	8
P2	1	4
P3	2	2
P4	3	1
P5	4	3
P6	5	2

ATA = 7.38MS
AWT = 4 MS.
Throughput → 3.3MS.

//priority preemptive

```
#include<stdio.h>
#define Max 9999;
 struct proc
  { int no,at,bt,rt,ct,wt,tat,pri,temp;
  };

  struct proc read(int i)
  { struct proc p;
   printf("\n process No: %d\n",i);
    p.no=i;
   printf("Enter Arrival time:");
   scanf("%d",&p.at);
   printf("Enter Burst time:");
    scanf("%d",&p.bt);
     p.rt=p.bt;
    printf("Enter priority:");
     scanf("%d",&p.pri);
      p.temp=p.pri);
       return p;
  }

  voidmain()
  { int i,n,c,remaining,min_val,min_index;
     struct proc p[10],temp;
     Float avgtat=0,avgwt=0;
    printf("<--Smallest priority first scheduling
         Algorithm (Preemptive)-->\n");
   printf("Enter Number of Processes:");
     scanf("%d",&n);
    For(int i=0;i<n;i++)
         p[i]=read(i+1);
    remaining=n;
    For(int i=0;i<n-1;i++)
          for(int j=0;j<n-i-1;j++)
```

```
        if(p[j].at >p[j+1].at)
         {  temp=p[j];
            p[j]=p[j+1];
            p[j+1]= temp;
            }
   min_val=p[0].temp,min_index=0;
    For(int j=0;j<n&&p[j].at<=p.at;j++)
        if(p[j].temp<min_val)
            min_val=p[j].temp,min_index=j;
      i=min_index;
      C=p[i].ct=p[i].at+1;
        p[i].rt--;
        if(p[i].rt==0)
        {   p[i].temp=max;
             remaining--;
        }
         while(remaining>0)
          {
           min_val=p[0].temp,min_index=0;
          For(int j= 0;j<n&&p[j].at<=c;j++)
              if(p[j].temp <min_val)
                   min_val=p[j].temp,min_index=j;
             i=main_index;
             p[i].ct=c=c+1;
             p[i].rt--;
             if(p[i].rt==0)
            {p[i].temp=max;
                remaining--;
          }  }
           avg tat/=n,avgwt/=n;
          printf("\nAverage Turnaround time=%f\
   nAverage waiting time=%f",avgtat,avgwt);
}
```

Processor	AT	BT
P1	3	4
P2	4	3
P3	6	2
P4	7	1

ATA = 4.7349

AWT = 5.32.

// Priority non-preemptive

```
#include<stdio.h>
 #define MAX 9999;
  struct proc
  {
    int no,at,bt,ct,wt,tat,pri,status;
  };

  struct proc read(int i)
   {
     struct proc p;
       printf("\n process no:%d\n",i);
        p.no=i;
      printf("Enter Arrival time:");
       scanf("%d",&p.at);
       printf("Enter Burst time:");
       scanf("%d",&p.bt);
       printf("Enter priority:");
       scanf("%d",&p.pri);
       p.status=0;
        return p;
     }
    void main()
    { int n,s,ct=0,remaining;
      struct proc p[10],temp;
        float avgtat=0,avgwt=0;
        printf("<--Smallest priority first scheduling
    Algorithm (non-preemptive)-->\n");
      printf("Enter number of processes:");
          scanf("%d",&n);
        for(int i=0;i<n;i++)
              p[i]=read(i+1);
              For(int i=0;i<n-1;i++)
              For(int j=0;j<
if(p[j].at>p[j+1].at)
  {temp=p[j];
   p[j]=p[j+1],
   p[j+1]=temp,
   }
 p[9].pri=MAX',
 remaining=n',
 printf("\n\nprocessNo\tAT\tBT\tpri\tCT\tTAT\tWT\
      tRT\n");
For(ct=p[0];remaining;){=0;)
 {s=9;
  for(int i=0; i<n;i++)
    if(p[i].at<=ct&&p[i].status!=1 &&p[i].
  pri<p[s].pri) s=i;
  p[s].ct=ct=ct+p[s].bt;
  p[s]. tat=p[s].ct-p[s].at',
   avgtat+=p[s].tat',
   remaining--;
   printf("P.%d\t\t%d\t%d\t%d\t%d\t%d\t%d\n",p[s].
  no,p[s].no,p[s],p[s],ct,p[s].tat,p[s].wt,p[s].wt);
 }
avgtat/=n avgwt/=n',
 printf("\n Average Turn Around time=%f\n Average
     Waiting time=%f",avgtat,avgwt)',
}
```

Processor	AT	BT
1	2	3
2	3	4
3	4	5
4	1	6
5	3	1
6	2	2

ATA= 7.32MS

ATAT= 4.33MS

```
// SJF(preemptive) stuf
#include<stdio.h>
void main()
 {
   int a[10],b[10],x[10];
   int waiting[10], turnaround[10], completion[10];
   int i,j,smallest, count=0,time,n;
   double avg=0, tt=0,end;

   printf("\n Enter the number of process:");
   scanf("%d",&n);
   for(i=0;i< ;i++)
   {
    printf("\nenter arrival time of process %d",i+1);
    scanf("%d",&a[i]);
    }

    for(i=0;i<n;i++)
     x[i]=b[i];
      b[9]=9999;
      for(time=0;count!=n;time++)
     {
      smallest=9;
       for(i=0;i<n;i++)
      {
        if(a[i]<=time &&b[i].<b[smallest] && b[i]>0)
           smallest=i;

     }
    b[smallest]--;
     if(b[smallest]==0)
     {
       count++;
       end=time+1;
      completion[smallest]=end;
      waiting[smallest]=end-a[smallest]-x[smallest];
      turnaround[smallest]=end-a[smallest];
   }
 }
printf("\n%d\t %d\t %d\t %d\t %d\t %d", i+1,
        bt[i], at[i], waiting[i], turnaround[i], completion[i]);
        avg = avg + waiting[i];
        att = att + turnaround[i];
    }

printf("\n %.2f %.2f", avg, att);
printf("\n\nAvg waiting time = %.2f\n", avg/n);
printf("Average Turnaround time = %.2f", att/n);

}
```

Output.

```
Enter processor 4
Enter arrival time : 1, 2, 6, 4
Enter BT           : 3 4 5 6

Average waiting time = 3.2500
Average turnaround time = 7.7500
```

[illegible]
30/5/24