SSN COLLEGE OF ENGINEERING, KALAVAKKAM
(An Autonomous Institution, Affiliated to Anna University, Chennai)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

UCS1411 - OPERATING SYSTEMS LAB

Lab Exercise 4: Implementation of CPU Scheduling Policies: Priority and Round Robin

NAME: GAYATHRI M
REG NO.: 185001050

PROGRAM:

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>

typedef struct
{
        char pid[10];
        int start,arr,burst,pri,rem,wait,turn,flag;
}job;


void gantt(job arr[], int n, int tot_time)
{
        if(n <= 0)
        return;

        printf("\n\n\tGANTT CHART");

        int i, j;

        printf("\n\n\t ");
        for(i=0; i<n-1; i++)
        {
        for(j=arr[i].start; j<arr[i+1].start; j++)
        printf("--");
        printf(" ");
        }

        for(j=0; j<tot_time - arr[n-1].start; j++)
        printf("--");
        printf(" ");
```

```
        printf("\n\t|");

        for(i=0; i<n-1; i++)
        {
        for(j=arr[i].start; j<arr[i+1].start - 1; j++)
        printf(" ");
        printf("%s", arr[i].pid);

        for(j=arr[i].start; j<arr[i+1].start - 1; j++)
        printf(" ");
        printf("|");
        }

        for(j=0; j<tot_time - arr[n-1].start - 1; j++)
        printf(" ");
        printf("%s", arr[n-1].pid);
        for(j=0; j<tot_time - arr[n-1].start - 1; j++)
        printf(" ");
        printf("|");

        printf("\n\t ");

        for(i=0; i<n-1; i++)
        {
        for(j=arr[i].start; j<arr[i+1].start; j++)
        printf("--");
        printf(" ");
        }

        for(j=0; j<tot_time - arr[n-1].start; j++)
        printf("--");
        printf(" ");

        printf("\n\t");

        for(i=0; i<n-1; i++)
        {
        printf("%d", arr[i].start);
        for(j=arr[i].start; j<arr[i+1].start; j++)
        printf("  ");
        if(arr[i].start > 9)
        printf("\b");
        }

        printf("%d", arr[n-1].start);
        for(j=0; j<tot_time - arr[n-1].start; j++)
        printf("  ");
        if(tot_time > 9)
        printf("\b%d", tot_time);
        printf("\n");
}
```

```
void display(job ar[],int n)
{
	float avgturn=0,avgwait=0;
	job temp;
	for(int i=0;i<n-1;i++)
	{
		for(int j=i+1;j<n;j++)
		{
			if(strcmp(ar[j].pid,ar[i].pid)<0)
			{
				temp=ar[j];
				ar[j]=ar[i];
				ar[i]=temp;
			}

		}
	}
	printf("\n---------------------------------------------------------------------------
\n");
	printf("\nProcess ID\tArrival Time\tBurst Time\tTurnaround\tWaiting
Time");
	printf("\n---------------------------------------------------------------------------
\n");
	for(int i=0;i<n;i++)
	{

	printf("\n%s\t\t%d\t\t%d\t\t%d\t\t%d",ar[i].pid,ar[i].arr,ar[i].burst,ar[i].t
urn,ar[i].wait);
		avgturn+=ar[i].turn;
		avgwait+=ar[i].wait;
	}
	printf("\n\t\t\t     Average: \t\t%.2f\t\t%.2f",avgturn/n,avgwait/n);
	printf("\n");
}


void rr(job ar[],int n)
{
	int time=0;
	float avgturn=0,avgwait=0;
	int prev=-1;
	printf("\n\n\t\tROUND ROBIN\n");
	time=0;
	int remain=0,endtime;
	int q;
	q=2;
	int t=0;
	int i=0;
	job temp;
	job g[10];
```

```
int count=0;

for(int i=0;i<n-1;i++)
{
	for(int j=i+1;j<n;j++)
	{
		if(ar[j].arr<ar[i].arr)
		{
			temp=ar[j];
			ar[j]=ar[i];
			ar[i]=temp;
		}
	}
}

for(int i=0;i<n;i++)
{
	time+=ar[i].burst;
}

while(remain!=n)
{
	if(ar[i].flag==0)
	{

		if(ar[i].rem>q && ar[i].flag==0)
		{
			ar[i].start=t;
			g[count++]=ar[i];
			t=t+q;
			ar[i].rem-=q;
		}
		else if(ar[i].flag==0)
		{
			ar[i].start=t;
			g[count++]=ar[i];
			t=t+ar[i].rem;
			ar[i].rem=0;
		}
		if(ar[i].rem==0)
		{
			remain++;
			endtime=t;
			ar[i].turn=endtime-ar[i].arr;
			ar[i].wait=endtime-ar[i].burst-ar[i].arr;
			ar[i].flag=1;
		}
		i=(i+1)%n;
	}
	else
	{
```

```
                i=(i+1)%n;
            }
        }
        display(ar,n);
        gantt(g,count,time);
}


void priority_np(job ar[],int n)
{
        int time=0;
        float avgturn=0,avgwait=0;
        int min;
        int index;
        job g[10];
        int count=0;

        printf("\n\n\t\tPRIORITY (Non Pre Emptive)\n");
        time=0;
        for(int i=0;i<n;i++)
        {
            time+=ar[i].burst;
        }

        for(int t=0;t<time;)
        {
            min=9999;
            for(int i=0;i<n;i++)
            {
                if(ar[i].arr<=t && ar[i].pri<min && ar[i].flag==0)
                {
                    min=ar[i].pri;
                    index=i;
                }
            }
            ar[index].flag=1;
            ar[index].wait=t-ar[index].arr;
            ar[index].start=t;
            g[count++]=ar[index];
            t+=ar[index].burst;
            ar[index].turn=t-ar[index].arr;
        }
        display(ar,n);
        gantt(g,count,time);
}


void priority_p(job ar[],int n)
{
        int time=0;
        float avgturn=0,avgwait=0;
```

```
	int min;
	int index;
	int prev=-1;
	printf("\n\n\t\tPRIORITY (Pre Emptive)\n");
	time=0;
	int remain=0,endtime;
	job g[10];
	int count=0;

	for(int i=0;i<n;i++)
	{
		time+=ar[i].burst;
	}

	for(int t=0;remain!=n;t++)
	{
		min=9999;
		for(int i=0;i<n;i++)
		{
			if( ar[i].arr<=t && ar[i].pri<min && ar[i].rem>0)
			{
				min=ar[i].pri;
				index=i;
			}
		}
		if(count!=0 && strcmp(g[count-1].pid , ar[index].pid)!=0)
		{
			ar[index].start=t;
			g[count++]=ar[index];
		}
		else if(count==0 && t==0)
		{
			ar[index].start=t;
			g[count++]=ar[index];
		}
		ar[index].rem-=1;

		if(ar[index].rem==0)
		{
			remain++;
			endtime=t+1;
			ar[index].turn=endtime-ar[index].arr;
			ar[index].wait=endtime-ar[index].burst-ar[index].arr;
		}
	}
	display(ar,n);
	gantt(g,count,time);
}


void input(job ar[],int n)
```

```
{
	for(int i=0;i<n;i++)
	{
		printf("\nEnter PID : ");
		scanf("%s",ar[i].pid);
		printf("Enter Arrival Time : ");
		scanf("%d",&ar[i].arr);
		printf("Enter Burst Time : ");
		scanf("%d",&ar[i].burst);
		printf("Enter Priority : ");
		scanf("%d",&ar[i].pri);
		ar[i].rem=ar[i].burst;
		ar[i].flag=0;
	}
	printf("\n");
}

int main()
{
	job ar[10];
	int n;
	int time=0;
	float avgturn=0,avgwait=0;
	int min;
	int index;
	int choice=3;
	while(choice!=0)
	{
		printf("\n\n\t\tCPU SCHEDULING ALGORITHMS\n");
		printf("1.ROUND ROBIN\n2.PRIORITY\n0.EXIT\nEnter Choice : ");
		scanf("%d",&choice);
		switch(choice)
		{
			case 1:
				printf("\t\tROUND ROBIN CPU SCHEDULER\n");
				printf("Enter Number of Processes : ");
				scanf("%d",&n);
				input(ar,n);
				rr(ar,n);
				break;
			case 2:
				printf("\t\tPRIORITY CPU SCHEDULER\n");
				printf("1. Non Preemptive PRIORITY\n2. Pre emptive
PRIORITY\nEnter your option : ");
				scanf("%d",&choice);
				printf("Enter Number of Processes : ");
				scanf("%d",&n);
				input(ar,n);
				if(choice==1)
					priority_np(ar,n);
				else if(choice==2)
```

```
                        priority_p(ar,n);
                    else
                        printf("Invalid Choice !!!\n");
                    break;
                }
            }
            return 0;
}
```

OUTPUT:

```
(base) MSMLs-iMac:ex4 msml$ ./rr_priority

        CPU SCHEDULING ALGORITHMS

1.ROUND ROBIN
2.PRIORITY
0.EXIT
Enter Choice : 1

                ROUND ROBIN CPU SCHEDULER

Enter Number of Processes : 5

Enter PID : P1
Enter Arrival Time : 0
Enter Burst Time : 6
Enter Priority : 2

Enter PID : P2
Enter Arrival Time : 1
Enter Burst Time : 2
Enter Priority : 2

Enter PID : P3
Enter Arrival Time : 1
Enter Burst Time : 3
Enter Priority : 4

Enter PID : P4
Enter Arrival Time : 2
Enter Burst Time : 1
Enter Priority : 1

Enter PID : P5
```

Enter Arrival Time : 2
Enter Burst Time : 2
Enter Priority : 3

ROUND ROBIN

Process ID	Arrival Time	Burst Time	Turnaround	Waiting Time
P1	0	6	14	8
P2	1	2	3	1
P3	1	3	11	8
P4	2	1	5	4
P5	2	2	7	5
		Average:	8.00	5.20

GANTT CHART

P1	P2	P3	P4	P5	P1	P3	P1

0 2 4 6 7 9 11 12 14

CPU SCHEDULING ALGORITHMS

1.ROUND ROBIN
2.PRIORITY
0.EXIT
Enter Choice : 1

PRIORITY CPU SCHEDULER

1. Non Preemptive PRIORITY
2. Pre emptive PRIORITY
Enter your option : 1

Enter Number of Processes : 5

Enter PID : P1
Enter Arrival Time : 0
Enter Burst Time : 6

Enter Priority : 2

Enter PID : P2
Enter Arrival Time : 1
Enter Burst Time : 2
Enter Priority : 2

Enter PID : P3
Enter Arrival Time : 1
Enter Burst Time : 3
Enter Priority : 4

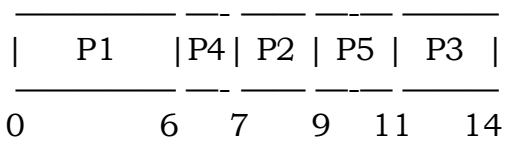
Enter PID : P4
Enter Arrival Time : 2
Enter Burst Time : 1
Enter Priority : 1

Enter PID : P5
Enter Arrival Time : 2
Enter Burst Time : 2
Enter Priority : 3

PRIORITY (Non Pre Emptive)

Process ID	Arrival Time	Burst Time	Turnaround	Waiting Time
P1	0	6	6	0
P2	1	2	8	6
P3	1	3	13	10
P4	2	1	5	4
P5	2	2	9	7
		Average:	8.20	5.40

GANTT CHART



CPU SCHEDULING ALGORITHMS

1.ROUND ROBIN
2.PRIORITY
0.EXIT
Enter Choice : 1

PRIORITY CPU SCHEDULER

1. Non Preemptive PRIORITY
2. Pre emptive PRIORITY
Enter your option : 2

Enter Number of Processes : 5

Enter PID : P1
Enter Arrival Time : 0
Enter Burst Time : 6
Enter Priority : 2

Enter PID : P2
Enter Arrival Time : 1
Enter Burst Time : 2
Enter Priority : 2

Enter PID : P3
Enter Arrival Time : 1
Enter Burst Time : 3
Enter Priority : 4

Enter PID : P4
Enter Arrival Time : 2
Enter Burst Time : 1
Enter Priority : 1

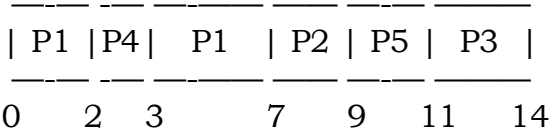
Enter PID : P5
Enter Arrival Time : 2
Enter Burst Time : 2
Enter Priority : 3

PRIORITY (Pre Emptive)

Process ID	Arrival Time	Burst Time	Turnaround	Waiting Time
P1	0	6	7	1
P2	1	2	8	6

P3	1	3	13	10
P4	2	1	1	0
P5	2	2	9	7
		Average:	7.60	4.80

GANTT CHART



CPU SCHEDULING ALGORITHMS

1.ROUND ROBIN
2.PRIORITY
0.EXIT
Enter Choice : 0