Assignment 3

code:

```
#include<stdio.h>

struct process
{
	int id;
	int arv;
	int burst;
	int exc;
	int prio;
	int flag;
};
int d[20],s[20];
struct process temp[20];
void accept(struct process p[20],int n)
{
	int i;
	for(i=0;i<n;i++)
	{
		p[i].id=i+1;
		printf("\n\tfor process p[%d]",i+1);
		printf("\n\tenter arrival time:");
		scanf("%d",&p[i].arv);
		printf("\n\tenter burst time:");
		scanf("%d",&p[i].burst);
		printf("\n\tenter priority:");
		scanf("%d",&p[i].prio);
		p[i].flag=0;
		p[i].exc=0;
	}
}
void sort(struct process temp[20],int n)
{
	int i,j;
	struct process t;
	for(i=0;i<n-1;i++)
	{
		for(j=0;j<n-1-i;j++)
		{
			if(temp[j].arv>temp[j+1].arv)
			{
				t=temp[j];
				temp[j]=temp[j+1];
				temp[j+1]=t;
			}
```

```
                }
        }
}
void display(int n)
{
        int i;
        printf("\n\t");
        for(i=0;i<n-1;i++)
        {

                printf("|\tP%d\t",s[i]);
        }

        printf("|\n\t");
        for(i=0;i<n;i++)
        {
                printf("%d\t\t",d[i]);
        }
}
void cal(struct process a[20],int n)
{
        int i,time=0,tawt;
        float awt,tat;
        for(i=0;i<n;i++)
        {
                time=time+a[i].exc-a[i].arv;
        }
        tawt=time;
        awt=(float)(time)/n;
        time=0;
        for(i=0;i<n;i++)
        {
                time=time+a[i].burst;
        }
        tat=(float)(tawt+time)/n;
        printf("\n\n\taverage waiting time=%.2f",awt);
        printf("\n\n\taverage turnaround time=%.2f",tat);
}


int time_comp(struct process temp[20],struct process a[20],int n)
{
        int i,k=0;
        for(i=0;i<n;i++)
        {
                if(temp[i].arv==0)
                {
                        a[k]=temp[i];
                        k++;
```

```
                }
        }
        return k;
}


void sjf_p(struct process p[20],int n)
{
        int i,j,t,min,time=0,var,in=0,cnt=0;
        for(i=0;i<n;i++)
                temp[i]=p[i];
        sort(temp,n);
        for(i=0;i<n;i++)
        {
                min=99;
                for(j=0;j<=i;j++)
                {
                        if(min>temp[j].burst && temp[j].flag==0)
                        {
                                min=temp[j].burst;
                                t=j;
                        }
                }
                if(i<n-1)
                {
                        var=temp[i+1].arv-temp[i].arv;
                        temp[t].exc=temp[t].burst-var;
                        temp[t].burst=temp[t].burst-var;
                        time=temp[i+1].arv;
                }
                else
                {
                        var=temp[i].arv;
                        time=var+temp[t].burst;
                        temp[t].burst=temp[t].exc=0;
                }
                d[in]=temp[i].arv;
                s[in]=temp[t].id;
                in++;
                if(temp[t].exc<0)
                {
                        temp[t].exc=temp[t].burst=0;
                }
                if(temp[t].burst<=0)
                {
                        p[temp[t].id-1].exc=time-p[t].burst;
                        temp[t].flag=1;
                }
        }
```

```
	do
	{
		min=99;
		for(i=0;i<n;i++)
		{
			if(min>temp[i].burst && temp[i].flag==0 && temp[i].burst>0)
			{
				min=temp[i].burst;
				t=i;
			}
		}
		temp[t].exc=0;
		temp[t].flag=1;
		d[in]=time;
		s[in]=temp[t].id;
		in++;
		time=time+temp[t].burst;
		p[temp[t].id-1].exc=time-p[t].burst;
		temp[t].burst=0;
		cnt=0;
		for(i=0;i<n;i++)
		{
			if(temp[i].flag==1)
				cnt++;
		}
	}while(cnt!=n);
	d[in]=time;
	in++;
	display(in);
	cal(p,n);
}


void round_r(struct process p[20],int n)
{
	int i,cnt,ts,time=0,in=0;
	printf("\n\temter timeslice:");
	scanf("%d",&ts);
	for(i=0;i<n;i++)
	{
		temp[i]=p[i];
	}
	do
	{
		for(i=0;i<n;i++)
		{
			if(temp[i].burst>0)
			{
				if(temp[i].burst>=ts)
```

```
                        {
                                temp[i].exc=temp[i].burst-ts;
                        }
                        else
                        {
                                temp[i].exc=temp[i].burst;
                        }
                        d[in]=time;
                        s[in]=temp[i].id;
                        in++;
                        if(temp[i].burst>=ts)
                        {
                                time=time+ts;
                        }
                        else
                        {
                                time=time+temp[i].burst;
                        }
                        temp[i].burst=temp[i].burst-ts;
                        if(temp[i].burst<=0)
                        {
                                p[temp[i].id-1].exc=time-p[i].burst;
                        }
                }
        }
        cnt=0;
        for(i=0;i<n;i++)
        {
                if(temp[i].burst<=0)
                        cnt++;
        }
    }while(cnt!=n);
    d[in]=time;
    in++;
    display(in);
    cal(p,n);
}
int main()
{
    int ch,n;
    struct process p[20];
    do
    {
        printf("\n\n\t**MENU**");
        printf("\n\t1.Accept");
      printf("\n\t2.Shortest Job First [Premptive]");
      printf("\n\t3.Round Robin");
      printf("\n\t0.Exit");
```

```
        printf("\n\tEnter Your Choice :");
        scanf("%d",&ch);
        switch(ch)
            {
                case 1:
            printf("\n\tEnter no.of processes:");
                scanf("%d",&n);
                accept(p,n);
                break;


                case 2:
            printf("\n\tSJF Preemptive:\n");
                sjf_p(p,n);
                break;


                case 3:
            printf("\n\tRound Robin:\n");
                round_r(p,n);
                break;
            }
    }while(ch!=0);
    return 0;
}
```

Output:

accepting input:

```
[ak-linux-computer@ak ~]$ gcc 3.c -o 3
[ak-linux-computer@ak ~]$ ./3


        **MENU**
        1.Accept
        2.Shortest Job First [Premptive]
        3.Round Robin
        0.Exit
        Enter Your Choice :1

        Enter no.of processes:5

        for process p[1]
        enter arrival time:0

        enter burst time:3

        enter priority:3

        for process p[2]
        enter arrival time:1

        enter burst time:5

        enter priority:1

        for process p[3]
        enter arrival time:3

        enter burst time:2

        enter priority:3

        for process p[4]
        enter arrival time:9

        enter burst time:5

        enter priority:2

        for process p[5]
        enter arrival time:12

        enter burst time:5

        enter priority:1
```

shortest job first - preemptive:

```
**MENU**
1.Accept
2.Shortest Job First [Premptive]
3.Round Robin
0.Exit
Enter Your Choice :2

SJF Preemptive:

|       P1      |       P1      |       P3      |       P2      |      P2       |       P4      |       P5      |
0               1               3               9               12     14              19              24

average waiting time=4.80

average turnaround time=8.80
```

round robbin:

```
**MENU**
1.Accept
2.Shortest Job First [Premptive]
3.Round Robin
0.Exit
Enter Your Choice :3

Round Robin:

emter timeslice:2

|      P1     |      P2     |      P3     |      P4     |      P5     |      P1     |      P2     |      P4     |      P5     |
0             2             4             6             8             10            11            13            15            17


average waiting time=5.80

average turnaround time=9.80
```