Program-7 (Round-Robin)

```
#include<stdio.h>
void main()
{
	int i,NOP,sum=0,count=0,y,quant,wt=0,tat=0,at[10],bt[10],temp[10];
	float avg_wt,avg_tat;
	printf("Total number of process in the system : ");
	scanf("%d",&NOP);
	y=NOP;
	for(int i=0;i<NOP;i++)
	{
		printf("\nEnter the arrival time and the Burst time of the
                                                        process[%d]\n",i+1);
		printf("Arrival time is : \t");
		scanf("%d",&at[i]);
		printf("\nBurst time is : \t");
		scanf("%d",&bt[i]);
		temp[i]=bt[i];
	}

	printf("Enter the time quantum for the process : \t");
	scanf("%d",&quant);
	printf("\nProcess No \t\t Burst time \t\t TAT \t\t Waiting time");
	for(sum=0,i=0;y!=0; )
	{
		if(temp[i]<=quant && temp[i] >0)
		{
			sum=sum+temp[i];
			temp[i]=0;
			count=1;
		}

		else if(temp[i]> 0)
		{
			temp[i]=temp[i] - quant;
			sum=sum+quant;
		}

		if(temp[i]==0 && count==1)
		{
			y--;
			printf("\nProcess no[%d] \t\t %d\t\t\t\t %d\t\t\t %d",i+1,bt[i],sum-
		at[i],sum-at[i]-bt[i]);
			wt=wt+sum-at[i]-bt[i];
			tat=tat+sum-at[i];
			count=0;
		}
```

```
            if(i==NOP-1)
            {
                    i=0;
            }
            else if(at[i+1]<=sum)
            {
                    i++;
            }
            else
            {
                    i=0;
            }
    }



    avg_wt=wt*1.0/NOP;
    avg_tat=tat*1.0/NOP;
    printf("\nAverage turn around time : \t %f",avg_tat);
    printf("\nAverage waiting time : \t %f",avg_wt);
}
```

Shortest Remaining Time

```
#include<stdio.h>
int main()
{
	int arrival_time[10],burst_time[10],temp[10];
	int i,smallest,count=0,time,limit;
	double wait_time=0,turnaround_time=0,end;
	float average_waiting_time,average_turnaround_time;
	printf("\nEnter the total number of processes : \t");
	scanf("%d",&limit);
	printf("\nEnter details of %d processes\n",limit);
	for(i=0;i<limit;i++)
	{
		printf("\nEnter Arrival time : \t");
		scanf("%d",&arrival_time[i]);
		printf("\nEnter Burst time : \t");
		scanf("%d",&burst_time[i]);
		temp[i]=burst_time[i];
	}
	burst_time[9]=9999;
	for(time=0;count!=limit;time++)
	{
		smallest=9;
		for(i=0;i<limit;i++)
		{
if(arrival_time[i]<=time && burst_time[i] < burst_time[smallest] && burst_time[i]>0)
	{
		smallest=i;
	}
		}
	    burst_time[smallest]--;
	    if(burst_time[smallest]==0)
	    {
		count++;
		end=time+1;
		wait_time=wait_time+end-arrival_time[smallest]-temp[smallest];
		turnaround_time=turnaround_time+end-arrival_time[smallest];
	    }
	}
average_waiting_time=wait_time/limit;
average_turnaround_time=turnaround_time/limit;
printf("\n\nAverage waiting time : \t %lf \n",average_waiting_time);
printf("Average turnaround time : \t %lf \n",average_turnaround_time);
return 0;
}
```