```
#include<stdio.h>
#include<stdlib.h>
struct proc
{
        int id;
        int arrival;
        int burst;
        int rem;
        int wait;
        int finish;
        int turnaround;
        float ratio;
}process[10];  //structure to hold the process information
struct proc temp;
int no;
int chkprocess(int);
int nextprocess();
void roundrobin(int, int, int[], int[]);
void srtf(int);
int main()
{
        int n,tq,choice;
        int bt[10],st[10],i,j,k;
        for(; ;)
        {
                printf(" 1. Round Robin\n 2. SRT\n 3. Exit \n");
                printf("Enter your choice\t");
                scanf("%d",&choice);
                switch(choice)
                {
                case 1: printf("Round Robin scheduling algorithm\n");
                        printf("Enter number of processes:\n");
                        scanf("%d",&n);
                        printf("Enter burst time for processes:");
                        for(i=0;i<n;i++)
                        {
                                scanf("%d",&bt[i]);
                                st[i]=bt[i];    //service time
                        }
                        printf("Enter time quantum:");
                        scanf("%d",&tq);
                        roundrobin(n,tq,st,bt);
                        break;
                case 2: printf("\n---SHORTEST REMAINING TIME FIRST---\n ");
                        printf("\n \n Enter the number of processes: ");
                        scanf("%d", &n);
                        srtf(n);
                        break;
               case 3: exit(0);
              }// end of switch
      }// end of for
}//end of main()

void  roundrobin(int n,int tq,int st[],int bt[])
 {
        int tat[10],wt[10],i,count=0,swt=0,stat=0,temp1,sq=0,j,k;
        float awt=0.0,atat=0.0;
        while(1)
        {

          for(i=0,count=0;i<n;i++)
           {
                temp1=tq;
                if(st[i]==0)
                 {
                        count++;
                        continue;
                }

                if(st[i]>tq)
                        st[i]=st[i]-tq;
                else
                         if(st[i]>=0)
                        {
                                temp1=st[i];
                                st[i]=0;
                        }
                sq=sq+temp1;
                tat[i]=sq;
            }
        if(n==count)
        break;
      } //end of while
      for(i=0;i<n;i++)
        {
                wt[i]=tat[i]-bt[i];
                swt=swt+wt[i];       // summation of  wait time
                stat=stat+tat[i];    // summation of turnaround time
        }
        awt=(float)swt/n;          // average wait time
        atat=(float)stat/n;         // average turnaround time
        printf("Process_no Burst time   Wait time       Turn around time\n");
        for(i=0;i<n;i++)
        printf("%d\t\t%d\t\t%d\t\t%d\n",i+1,bt[i],wt[i],tat[i]);
        printf("Avg wait time is %f\n Avg turn around time is %f\n",awt,atat);
 }

int chkprocess(int s)         // function to check process remaining time is zero
or not
{
        int i;
        for(i = 1; i <= s; i++)
        {
                if(process[i].rem != 0)
                return 1;
        }
        return 0;
} // end of chkprocess

int nextprocess()        // function to identify the next process to be executed
{
        int min, l, i;
        min = 32000;   //any limit assumed
        for(i = 1; i <= no; i++)
        {
                if( process[i].rem!=0 && process[i].rem < min)
                {
                        min = process[i].rem;
                        l = i;
                }
        }
        return l;
}  // end of nextprocess
void srtf(int n)
{
int i,j,k,time=0,t;
float tavg=0,wavg=0;
for(i=1;i<=n;i++)
process[i].rem=process[i].wait=process[i].finish=process[i].turnaround=0;
for(i = 1; i <= n; i++)
        {
                process[i].id = i;
                printf("\n\nEnter the arrival time for process %d: ", i);
                scanf("%d", &(process[i].arrival));
                printf("Enter the burst time for process %d: ", i);
                scanf("%d", &(process[i].burst));
                process[i].rem = process[i].burst;
         }
        for(i = 1; i <= n; i++)
        {
                for(j = i + 1; j <= n; j++)
                {
                        if(process[i].arrival > process[j].arrival)

                        {
                                temp = process[i];
                                process[i] = process[j];
                                process[j] = temp;
                        }
                }
        }
        no = 0;
        j = 1;
        while(chkprocess(n) == 1)
        {
                for(t=1;t<=n;t++)
                {
                        if(process[no + 1].arrival == time)
                        {
                                no++;
                                if(process[j].rem==0)
                                        process[j].finish=time;
                                j = nextprocess();
                        }
                }
                if(process[j].rem != 0)
                {
                        process[j].rem--;
                        for(i = 1; i <= no; i++)
                        {
                                if(i != j && process[i].rem != 0)
                                process[i].wait++;
                        }
                }
                else
                {
                        process[j].finish = time;
                        j=nextprocess();
                        time--;
                        k=j;
                }
                time++;
        }
        process[k].finish = time;
        printf("\n\n Process  Arrival  Burst   Waiting  Finishing turnaround  Tr/
Tb \n");
        printf("%5s %9s %7s %10s %8s %9s\n\n", "id", "time", "time", "time",
"time",
                "time");
        for(i = 1; i <= n; i++)
        {

        process[i].turnaround = process[i].wait + process[i].burst;
        process[i].ratio = (float)process[i].turnaround / (float)process[i].burst;
        printf("%5d %8d %7d  %8d %10d %9d %10.1f ", process[i].id,
        process[i].arrival, process[i].burst, process[i].wait, process[i].finish,
        process[i].turnaround, process[i].ratio);
        tavg=tavg+ process[i].turnaround;    //summation of turnaround time
        wavg=wavg+process[i].wait;             // summation of waiting time
        printf("\n\n");
        }

        tavg=tavg/n;        // average turnaround time
        wavg=wavg/n;     // average wait time
        printf("tavg=%f\t wavg=%f\n",tavg,wavg);
}// end of srtf
```