```
#include<stdio.h>
#include<limits.h>
#include<stdbool.h>

struct P{
        int AT,BT,ST[20],WT,FT,TAT,pos;
};

int quant;
int main(){
        int n,i,j;
        printf("Enter the no. of processes :");
        scanf("%d",&n);
        struct P p[n];

        printf("Enter the quantum \n");
        scanf("%d",&quant);

        printf("Enter the process numbers \n");
        for(i=0;i<n;i++)
                scanf("%d",&(p[i].pos));

        printf("Enter the Arrival time of processes \n");
        for(i=0;i<n;i++)
                scanf("%d",&(p[i].AT));

        printf("Enter the Burst time of processes \n");
        for(i=0;i<n;i++)
                scanf("%d",&(p[i].BT));

        int c=n,s[n][20];
        float time=0,mini=INT_MAX,b[n],a[n];

        int index=-1;
        for(i=0;i<n;i++)
                {
                        b[i]=p[i].BT;
                        a[i]=p[i].AT;
                        for(j=0;j<20;j++)
                        {
                                s[i][j]=-1;
                        }
                }

        int tot_wt,tot_tat;
        tot_wt=0;
        tot_tat=0;
        bool flag=false;

        while(c!=0){

        mini=INT_MAX;
        flag=false;

        for(i=0;i<n;i++){
                        float p=time+0.1;
                        if(a[i]<=p && mini>a[i] && b[i]>0){
                        index=i;
                        mini=a[i];
                        flag=true;

                        }
        }
        if(!flag){
                        time++;
                        continue;
        }


        j=0;

        while(s[index][j]!=-1){
        j++;
        }

        if(s[index][j]==-1){
        s[index][j]=time;
        p[index].ST[j]=time;
        }

        if(b[index]<=quant){
        time+=b[index];
        b[index]=0;
        }
        else{
        time+=quant;
        b[index]-=quant;
        }

        if(b[index]>0){
        a[index]=time+0.1;
        }


        if(b[index]==0){
                c--;
                p[index].FT=time;
                p[index].WT=p[index].FT-p[index].AT-p[index].BT;
                tot_wt+=p[index].WT;
                p[index].TAT=p[index].BT+p[index].WT;
                tot_tat+=p[index].TAT;

                }

        }


        printf("Process number ");
        printf("Arrival time ");
        printf("Burst time ");
        printf("\tStart time");
        j=0;
        while(j!=10){
                j+=1;
                printf(" ");
        }
        printf("\t\tFinal time");
        printf("\tWait Time ");
        printf("\tTurnAround Time \n");


        for(i=0;i<n;i++){
                printf("%d \t\t",p[i].pos);
                printf("%d \t\t",p[i].AT);
                printf("%d \t",p[i].BT);
                j=0;
                int v=0;
                while(s[i][j]!=-1){
                printf("%d ",p[i].ST[j]);
                j++;
                v+=3;
                }
                while(v!=40){
                printf(" ");
                v+=1;
                }
                printf("%d \t\t",p[i].FT);
                printf("%d \t\t",p[i].WT);
                printf("%d \n",p[i].TAT);

        }


        double avg_wt,avg_tat;
        avg_wt=tot_wt/(float)n;
        avg_tat=tot_tat/(float)n;


        printf("The average wait time is : %lf\n",avg_wt);
        printf("The average TurnAround time is : %lf\n",avg_tat);

        return 0;
}
```