Q FCFS scheduling

```
#include<stdio.h>

void compTime(int n, int a[], int b[], int c[]){
    int t=0;
    for(int i=0; i<n; i++){
        if(a[i]>t){
            t=a[i];
            t+=b[i];
            c[i]=t;
        }
    }
}

void tatTime(int n, int a[], int c[], int t[]){
    for(int i=0; i<n; i++){
        t[i]=c[i]-a[i];
}

void waitTime(int n, int b[], int t[], int w[]){
    for(int i=0; i<n; i++){
        w[i]=t[i]-b[i];
    }
}

void avgTime(int n, int a[], int b[]){
    int c[n], t[n], w[n];
    int sumTat=0, sumWt=0;
    compTime(n,a,b,c);
    tatTime(n,a,c,t);
    waitTime(n,b,t,w);
    printf("P A B C T W \n");
```

```
for (int i=0; i<n; i++) {
    printf("%d %d %d %d %d %d \n",
            i+1, a[i], b[i], c[i], t[i], w[i]);
    sumTat += t[i];
    sumWt += w[i];
}
float avgTat = (float) sumTat/n;
float avgWt = (float) sumWt/n;
printf("Average TAT = %f \n Average WT = %f
        \n", avgTat, avgWt);
}

int main() {
    int n=4
    int a[] = {0,1,5,6};
    int b[] = {2,2,3,4};
    avgTime(n, a, b);
    return 0;
}
```

O/P

P	A	B	C	T	W
1	0	2	2	2	0
2	1	2	4	3	1
3	5	3	8	3	0
4	6	4	12	6	2

Average TAT = 3.500000
Average WT = 0.750000

Q SJF scheduling.

```
#include<stdio.h>

void findCompletionTime(int process[], int n, int bt[],
    int at[], int wt[], int tat[], int rt[], int ct[]){
    int completion[n];
    int remaining[n];
    for(int i=0; i<n; i++){
        remaining[i] = bt[i];
    }
    int currentTime=0;
    for(int i=0; i<n; i++){
        int shortest = -1;
        for(int j=0; j<n; j++){
            if(at[j] <= currentTime &&
                remaining[j] > 0){
                if(shortest == -1 || remaining[j]
                    < remaining[shortest]){
                    shortest = j;
                }
            }
        }
        if(shortest == -1){
            currentTime++;
            continue;
        }
        completion[shortest] = currentTime +
            remaining[shortest];
        currentTime = completion[shortest];
        wt[shortest] = currentTime - bt[shortest]
            - at[shortest];
        tat[shortest] = currentTime - at[shortest];
        rt[shortest] = wt[shortest];
        remaining[shortest] = 0;
    }
```

```
printf("Process \t Arrival \t Burst \t waiting \t
        Turnaround \t Response \t completion \n");
for(int i=0; i<n; i++){
    ct[i] = completion[i];
    printf("%.d \t %.d \t %.d \t %.d \t %.d \t %.d \t
                %d \n", process[i],
            at[i], bt[i], wt[i], tat[i], rt[i], ct[i]
    }
}

void main(){
    int n;
    printf("Enter the number of process: \n");
    scanf("%d", &n);
    int process[n];
    int burstTime[n];
    int arrivalTime[n];
    printf("Enter the process number: \n");
    for(int i=0; i<n; i++){
        scanf("%d", &process[i]);
    }
    printf("Enter arrival time \n");
    for(int i=0; i<n; i++){
        scanf("%d", &arrivaltime[i]);
    }
    printf("Enter Burst time \n");
    for(int i=0; i<n; i++){
        scanf("%d", &burstTime[i]);
    }
    int wt[n], tat[n], rt[n], ct[n];
    for(int i=0; i<n; i++){
        rt[i] = -1;
    }
```

```
printf("\n\n STF (Non premptive scheduling)\n");
findCompletionTime(process, n, burstTime, arrivalT
                   -ime, wt, tat, rt, ct);
}
```

O/P

```
Enter the number of process: 5
Enter the process numbers:
1 2 3 4 5
Enter the arrival time:
2 1 4 0 2
Enter burst time:
1 5 1 6 3
```

STF (Non premptive scheduling)

process	Arrival	Burst	waiting	Turnaround	Response	comp-
1	2	1	4	5	4	7
2	1	5	10	15	10	16
3	4	1	3	4	3	8
4	0	6	0	6	0	6
5	2	3	6	9	6	11

Average turnaround time : 7.800000
Average waiting time : 4.600000