WEEK 2

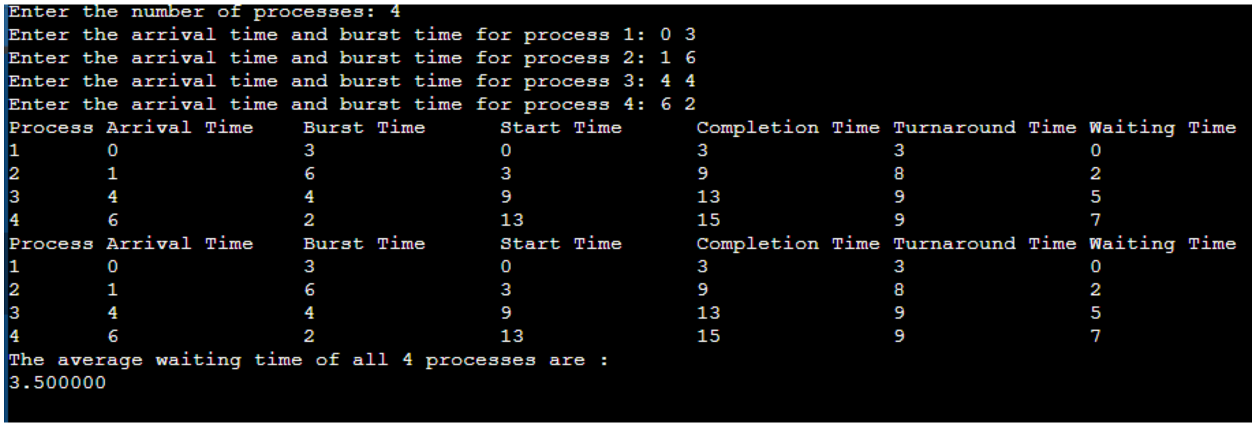
Y. SHAMIL AHAMED

1BM21CS248.

ALGORITHM 1: FCFS

```
#include<stdio.h>
typedef struct
{
	int pID,aT,bT,sT,cT,taT,wT;
} Process;

void calculateTimes(Process p[], int n)
{
	int currT = 0;
	for (int i = 0; i < n; i++)
	{
	p[i].sT = currT;
	p[i].cT = currT + p[i].bT;
	p[i].taT = p[i].cT - p[i].aT;
	p[i].wT = p[i].taT - p[i].bT;
	currT = p[i].cT;
	}
}

void displayp(Process p[], int n)
{
	printf("Process\tArrival Time\tBurst Time\tStart Time\tCompletion Time\tTurnaround
Time\tWaiting Time\n");

	for (int i = 0; i < n; i++)
	{
	printf("%d\t%d\t\t%d\t\t%d\t\t%d\t\t%d\t\t%d\n", p[i].pID, p[i].aT,
	p[i].bT, p[i].sT, p[i].cT,
	p[i].taT, p[i].wT);
	}
}
void averageWaitingTime(Process p[], int n){
        printf("The average waiting time of all %d processes are :\n",n);
        float sum=0.0;
        int k;
        for(k=0;k<n;k++){
        sum+=p[k].wT;
        }
        float avg = (sum/n);
        printf("%f",avg);
}

int main() {
        int n;
        printf("Enter the number of processes: ");
        scanf("%d", &n);
        Process p[n];
        for (int i = 0; i < n; i++) {
        printf("Enter the arrival time and burst time for process %d: ", i + 1);
        scanf("%d %d", &p[i].aT, &p[i].bT);
        p[i].pID = i + 1;
        }
        calculateTimes(p, n);
        displayp(p, n);

        for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++) {
        if (p[j].aT > p[j + 1].aT) {
                Process temp = p[j];
                p[j] = p[j + 1];
                p[j + 1] = temp;
        }
        }
        }

        calculateTimes(p, n);
        displayp(p, n);
        averageWaitingTime(p, n);
        return 0;
}
```

OUTPUT:


```
Enter the number of processes: 4
Enter the arrival time and burst time for process 1: 0 3
Enter the arrival time and burst time for process 2: 1 6
Enter the arrival time and burst time for process 3: 4 4
Enter the arrival time and burst time for process 4: 6 2
Process Arrival Time    Burst Time      Start Time      Completion Time Turnaround Time Waiting Time
1       0               3               0               3               3               0
2       1               6               3               9               8               2
3       4               4               9               13              9               5
4       6               2               13              15              9               7
Process Arrival Time    Burst Time      Start Time      Completion Time Turnaround Time Waiting Time
1       0               3               0               3               3               0
2       1               6               3               9               8               2
3       4               4               9               13              9               5
4       6               2               13              15              9               7
The average waiting time of all 4 processes are :
3.500000
```

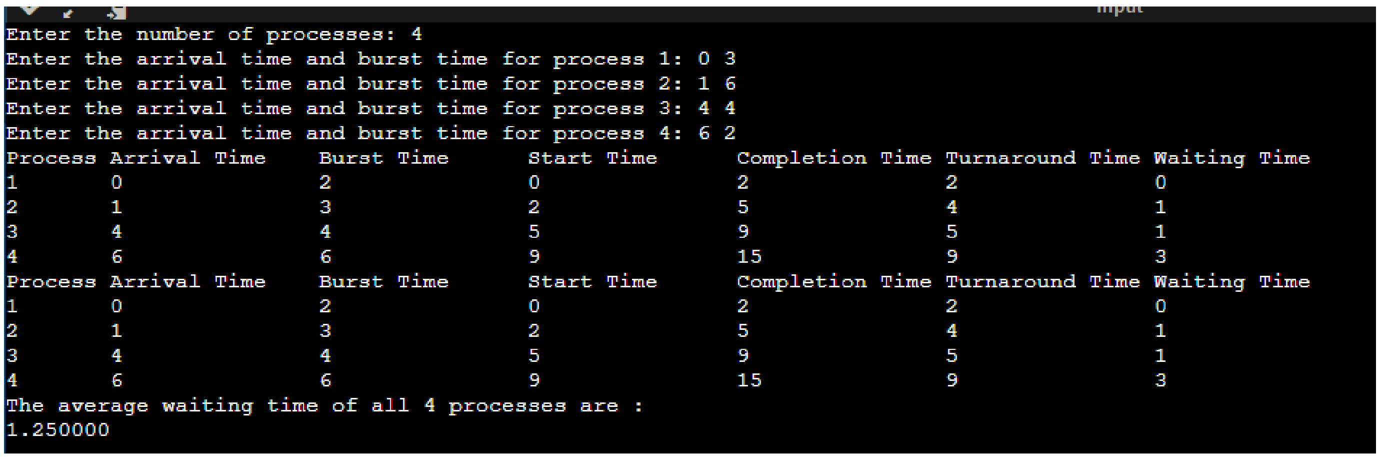
ALGORITHM 2: Shortest Job First

```
#include<stdio.h>
typedef struct
{
	int pID,aT,bT,sT,cT,taT,wT;
} Process;

void calculateTimes(Process p[], int n)
{
	int i,j,t;
	for(i=0;i<n-1;i++){
	for(j=0;j<(n-i-1);j++){
	if(p[j].bT > p[j+1].bT){
		t=p[j+1].bT;
		p[j+1].bT = p[j].bT;
		p[j].bT = t;
	}
	}
	}
	int currT = 0;
	for (int i = 0; i < n; i++)
	{
	p[i].sT = currT;
	p[i].cT = currT + p[i].bT;
	p[i].taT = p[i].cT - p[i].aT;
	p[i].wT = p[i].taT - p[i].bT;
	currT = p[i].cT;
	}
```

```
}

void displayp(Process p[], int n)
{
	printf("Process\tArrival Time\tBurst Time\tStart Time\tCompletion Time\tTurnaround
Time\tWaiting Time\n");

	for (int i = 0; i < n; i++)
	{
	printf("%d\t%d\t\t%d\t\t%d\t\t%d\t\t%d\t\t%d\n", p[i].pID, p[i].aT,
	p[i].bT, p[i].sT, p[i].cT,
	p[i].taT, p[i].wT);
	}
}
void averageWaitingTime(Process p[], int n){
	printf("The average waiting time of all %d processes are :\n",n);
	float sum=0.0;
	int k;
	for(k=0;k<n;k++){
	sum+=p[k].wT;
	}
	float avg = (sum/n);
	printf("%f",avg);
}

int main() {
	int n;
	printf("Enter the number of processes: ");
	scanf("%d", &n);
	Process p[n];
	for (int i = 0; i < n; i++) {
	printf("Enter the arrival time and burst time for process %d: ", i + 1);
	scanf("%d %d", &p[i].aT, &p[i].bT);
	p[i].pID = i + 1;
	}
	calculateTimes(p, n);
	displayp(p, n);;

	for (int i = 0; i < n - 1; i++) {
	for (int j = 0; j < n - i - 1; j++) {
	if (p[j].aT > p[j + 1].aT) {
		Process temp = p[j];
		p[j] = p[j + 1];
		p[j + 1] = temp;
```

```
            }
        }
    }

    calculateTimes(p, n);
    displayp(p, n);
    averageWaitingTime(p,n)
    return 0;
}
```

OUTPUT:



```
Enter the number of processes: 4
Enter the arrival time and burst time for process 1: 0 3
Enter the arrival time and burst time for process 2: 1 6
Enter the arrival time and burst time for process 3: 4 4
Enter the arrival time and burst time for process 4: 6 2
Process Arrival Time    Burst Time      Start Time      Completion Time Turnaround Time Waiting Time
1       0               2               0               2               2               0
2       1               3               2               5               4               1
3       4               4               5               9               5               1
4       6               6               9               15              9               3
Process Arrival Time    Burst Time      Start Time      Completion Time Turnaround Time Waiting Time
1       0               2               0               2               2               0
2       1               3               2               5               4               1
3       4               4               5               9               5               1
4       6               6               9               15              9               3
The average waiting time of all 4 processes are :
1.250000
```