```
//First Come First Served Scheduling
#include <stdio.h>
#define MAX 100

struct process
{
    int AT;
    int BT;
    int id;
    int CT;
    int WT;
    int TAT;
};

void swap(process &a, process &b)
{
    process t = a;
    a = b;
    b = t;
}

int get_partion(process A[], int start , int end)
{
    int pivot = A[end].AT;
    int i=start-1;
    for(int j=start ; j<end ; j++)
    {
        if(A[j].AT <= pivot)
        {
            i++;
            swap(A[i], A[j]);
        }
    }
    swap(A[end], A[i+1]);
    return i+1;
}

void quicksort(process A[], int start, int end)
{
    if(start < end)
    {
        int partion = get_partion(A, start, end);
        quicksort(A, start, partion-1);
        quicksort(A, partion+1, end);
    }
}

void calculateCT(process P[], int n)
{
    P[0].CT = P[0].AT + P[0].BT;
    int curr=P[0].CT;
    for (int i = 1; i < n; ++i)
    {
        P[i].CT = curr+P[i].BT;
        curr = P[i].CT;
    }
}

void calculateTAT(process P[], int n)
{
    for (int i = 0; i < n; ++i)
    {
        P[i].TAT = P[i].CT - P[i].AT;
    }
}

void calculateWT(process P[], int n)
{
    for (int i = 0; i < n; ++i)
```

```
    {
        P[i].WT = P[i].TAT - P[i].BT;
    }
}

int main()
{
    process P[100];
    int n;
    printf("Number of processes : ");
    scanf("%d", &n);
    for(int i=0 ; i<n ; i++)
    {
        printf("Process %d:\n", i+1);
        printf("AT : ");
        scanf("%d", &P[i].AT);
        printf("BT : ");
        scanf("%d", &P[i].BT);
        P[i].id = i+1;
        P[i].CT = 0;
        P[i].WT = 0;
        P[i].TAT = 0;
    }
    quicksort(P, 0, n);
    calculateCT(P, n);
    calculateTAT(P, n);
    calculateWT(P, n);
    printf("\n\n");
    printf("P\tAT\tBT\tCT\tTAT\tWT\n");
    float avgWT=0,avgTAT=0;
    for (int i = 0; i < n; ++i)
    {
        printf("%d\t%d\t%d\t%d\t%d\t%d\n", P[i].id, P[i].AT, P[i].BT, P[i].CT, P
[i].TAT, P[i].WT);
        avgTAT+=P[i].TAT;
        avgWT+=P[i].WT;
    }
    printf("Average Turn Around Time : %f\n", (avgTAT/n)*1.0);
    printf("Average Wating Time : %f\n", (avgWT/n)*1.0);
}

/*
OUTPUT

Number of processes : 6
Process 1:
AT : 6
BT : 4
Process 2:
AT : 2
BT : 5
Process 3:
AT : 3
BT : 3
Process 4:
AT : 1
BT : 1
Process 5:
AT : 4
BT : 2
Process 6:
AT : 5
BT : 6


P   AT  BT  CT  TAT WT
4   1   1   2   1   0
2   2   5   7   5   0
3   3   3   10  7   4
```

```
5   4   2   12  8   6
6   5   6   18  13  7
1   6   4   22  16  12
Average Turn Around Time : 8.333333
Average Wating Time : 4.833333
*/
```