#include <stdio.h>

int main()

{

int n, bt[20], at[20], p[20], wt[20], tat[20], priority[20], i, j, total = 0, pos, temp;

float avgwt = 0, avgtat = 0;

// Reading number of processes from input.

printf("Enter number of processes: ");

scanf("%d", &n);

// Reading all burst times and Priority of processes from input. Assuming Arrival time is 0 for all.

printf("Enter Burst Time and Priority value of each process: \n");

for (i = 0; i < n; i++)

{

p[i] = i;

printf("Burst Time of P[%d]: ", i);

scanf("%d", &bt[i]);

printf("Priority of P[%d]: ", i);

scanf("%d", &priority[i]);

at[i] = 0; // Assuming AT is zero for all.

}

// Sorting the table according to the PRIORITY Value (priority[i])

for (i = 0; i < n; i++)

{

pos = i;

for (j = i + 1; j < n; j++)

{

if (priority[j] < priority[pos])

{

pos = j;

}

}

temp = bt[i];

priority[i] = priority[pos];

priority[pos] = temp;

// Also sorting corresponding PID values.

temp = p[i];

p[i] = p[pos];

p[pos] = temp;

// Also sorting Corresponding BT values

temp = bt[i];

bt[i] = bt[pos];

bt[pos] = temp;

}

// After this sorting step, the array priority[] is in ascending order. So the first element in the table doesn't have to wait. Its wt is now 0

/\* CALCULATING WAITING TIME \*/

wt[0] = 0; // The current first Process doesnt have to wait, (after sorting)

// Setting wt values for all other processes.

for (i = 1; i < n; i++) // wt of 0 is set as 0. So we start with index 1 of wt[].

{

wt[i] = 0; // initialize wt of i'th index as 0

for (j = 0; j < i; j++)

{

wt[i] = wt[i] + bt[j]; // adding waiting times of previous processes and the current burst time.

}

}

/\* CALCULATING TURNAROUND TIME \*/

for (i = 0; i < n; i++)

{

tat[i] = wt[i] + bt[i]; // Equation for TAT is TAT= WT+BT

}

// Displaying Table

printf("\n\tPID\tAT\tBT\tWT\tTAT\n");

for (i = 0; i < n; i++)

{

printf("\t%d\t%d\t%d\t%d\t%d\n", p[i], at[i], bt[i], wt[i], tat[i]);

}

// Calculating Average Waiting time and Average Turnaround Time

for (i = 0; i < n; i++)

{

avgwt = avgwt + wt[i];

avgtat = avgtat + tat[i];

}

avgwt = avgwt / n;

avgtat = avgtat / n;

// Displaying Average WT and Average TAT

printf("\nAverage WT = %f\n", avgwt);

printf("\nAverage TAT = %f\n", avgtat);

}