#include <stdio.h>

struct Process {

int pid; // Process ID

int at; // Arrival Time

int bt; // Burst Time

int ct; // Completion Time

int tat; // Turnaround Time

int wt; // Waiting Time

int done; // Flag if process is completed

};

int main() {

int n, i, completed = 0, idx, min\_bt;

float avg\_wt = 0, avg\_tat = 0;

printf("Enter number of processes: ");

scanf("%d", &n);

struct Process p[n];

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

p[i].pid = i + 1;

printf("Enter arrival time for process P%d: ", i + 1);

scanf("%d", &p[i].at);

printf("Enter burst time for process P%d: ", i + 1);

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

p[i].done = 0;

}

int current\_time = 0;

int start, end;

printf("\nGantt Chart:\n");

while (completed < n) {

idx = -1;

min\_bt = 1e9;

// Find shortest job among available processes

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

if (!p[i].done && p[i].at <= current\_time) {

if (p[i].bt < min\_bt) {

min\_bt = p[i].bt;

idx = i;

}

}

}

if (idx == -1) { // CPU idle

current\_time++;

} else {

start = current\_time;

end = start + p[idx].bt;

printf("P%d (%d -- %d) ", p[idx].pid, start, end);

current\_time = end;

p[idx].ct = current\_time;

p[idx].tat = p[idx].ct - p[idx].at;

p[idx].wt = p[idx].tat - p[idx].bt;

p[idx].done = 1;

completed++;

avg\_wt += p[idx].wt;

avg\_tat += p[idx].tat;

}

}

// Print process details

printf("\n\nProcess\tAT\tBT\tCT\tTAT\tWT\n");

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

printf("P%d\t%d\t%d\t%d\t%d\t%d\n",

p[i].pid, p[i].at, p[i].bt,

p[i].ct, p[i].tat, p[i].wt);

}

printf("\nAverage Waiting Time = %.2f", avg\_wt / n);

printf("\nAverage Turnaround Time = %.2f\n", avg\_tat / n);

return 0;

}

OUTPUT

