**SJF:**

import java.util.\*;

public class SJF {

public static void main(String args[]) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter the number of processes:");

int n = sc.nextInt();

int pid[] = new int[n]; // process IDs

int at[] = new int[n]; // arrival times

int bt[] = new int[n]; // burst times

int ct[] = new int[n]; // completion times

int ta[] = new int[n]; // turnaround times

int wt[] = new int[n]; // waiting times

int f[] = new int[n]; // flag to check if the process is completed

int st = 0, tot = 0; // st: system time, tot: total completed processes

float avgwt = 0, avgta = 0;

// Input arrival and burst times for each process

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

System.out.println("Enter process " + (i + 1) + " arrival time:");

at[i] = sc.nextInt();

System.out.println("Enter process " + (i + 1) + " burst time:");

bt[i] = sc.nextInt();

pid[i] = i + 1;

f[i] = 0; // Mark process as not completed

}

// Implementing SJF scheduling

while (true) {

int c = n, min = 999; // c: selected process index, min: minimal burst time

if (tot == n) // If total number of processes equals completed processes, break loop

break;

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

/\* If the process has arrived and is not completed, and burst time is less than current minimum, select it \*/

if ((at[i] <= st) && (f[i] == 0) && (bt[i] < min)) {

min = bt[i];

c = i;

}

}

// If no process is ready to execute, increase system time

if (c == n)

st++;

else {

ct[c] = st + bt[c]; // Completion time of the selected process

st += bt[c]; // Update system time

ta[c] = ct[c] - at[c]; // Turnaround time = Completion time - Arrival time

wt[c] = ta[c] - bt[c]; // Waiting time = Turnaround time - Burst time

f[c] = 1; // Mark process as completed

tot++; // Increase total completed processes

}

}

// Display process details

System.out.println("\nPID\tArrival\tBurst\tComplete\tTurnaround\tWaiting");

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

avgwt += wt[i];

avgta += ta[i];

System.out.println(pid[i] + "\t" + at[i] + "\t" + bt[i] + "\t" + ct[i] + "\t\t" + ta[i] + "\t\t" + wt[i]);

}

// Display average waiting and turnaround times

System.out.println("\nAverage Turnaround Time: " + (avgta / n));

System.out.println("Average Waiting Time: " + (avgwt / n));

sc.close();

}

}