5. Write a program to compute the average waiting time and turnaround time based on Preemptive shortest remaining processing time first (SRPT) algorithm for the following set of processes, with the arrival times and the CPU-burst times given in milliseconds

Process Arrival Time Burst Time

P1 0 5

P2 1 3

P3 2 3

P4 4 1

Program:-

#include <stdio.h>

struct Process {

int pid;

int arrival\_time;

int burst\_time;

int remaining\_time;

};

void execute\_process(struct Process\* p) {

p->remaining\_time--;

}

int is\_process\_completed(struct Process\* p) {

return p->remaining\_time == 0;

}

int main() {

struct Process processes[] = {{1, 0, 5, 5}, {2, 1, 3, 3}, {3, 2, 3, 3}, {4, 4, 1, 1}};

int n = sizeof(processes) / sizeof(struct Process);

int completed\_processes[n];

int num\_completed\_processes = 0;

int current\_time = 0;

float total\_waiting\_time = 0;

float total\_turnaround\_time = 0;

while (num\_completed\_processes < n) {

int ready\_processes[n];

int num\_ready\_processes = 0;

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

if (processes[i].arrival\_time <= current\_time && !is\_process\_completed(&processes[i])) {

ready\_processes[num\_ready\_processes++] = i;

}

}

if (num\_ready\_processes == 0) {

current\_time++;

continue;

}

int shortest\_remaining\_time = processes[ready\_processes[0]].remaining\_time;

int shortest\_process\_index = ready\_processes[0];

for (int i = 1; i < num\_ready\_processes; i++) {

int process\_index = ready\_processes[i];

if (processes[process\_index].remaining\_time < shortest\_remaining\_time) {

shortest\_remaining\_time = processes[process\_index].remaining\_time;

shortest\_process\_index = process\_index;

}

}

struct Process\* p = &processes[shortest\_process\_index];

execute\_process(p);

if (is\_process\_completed(p)) {

completed\_processes[num\_completed\_processes++] = shortest\_process\_index;

int finish\_time = current\_time + 1;

int waiting\_time = finish\_time - p->arrival\_time - p->burst\_time;

int turnaround\_time = finish\_time - p->arrival\_time;

total\_waiting\_time += waiting\_time;

total\_turnaround\_time += turnaround\_time;

}

current\_time++;

}

float avg\_waiting\_time = total\_waiting\_time / n;

float avg\_turnaround\_time = total\_turnaround\_time / n;

printf("Average Waiting Time: %.2f ms\n", avg\_waiting\_time);

printf("Average Turnaround Time: %.2f ms\n", avg\_turnaround\_time);

return 0;

}

OUTPUT:-

