Lab Assignment 1

Chaudhary Hamdan

1905387

Date: 07-01-2022

Question:

Scheduling Algorithm(RR,SJF(preemptive&nonpreemptive),FCFS)

1. FCFS:

Code:

// Author: Chaudhary Hamdan, 1905387

#include<stdio.h>

void findWaitingTime(int processes[], int n, int bt[], int wt[])

{

wt[0] = 0;

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

wt[i] = bt[i - 1] + wt[i - 1] - processes[i];

}

void findTurnAroundTime( int processes[], int n, int bt[], int wt[], int tat[])

{

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

tat[i] = bt[i] + wt[i];

}

void findavgTime( int processes[], int n, int bt[])

{

int wt[n], tat[n], total\_wt = 0, total\_tat = 0;

findWaitingTime(processes, n, bt, wt);

findTurnAroundTime(processes, n, bt, wt, tat);

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

{

total\_wt = total\_wt + wt[i];

total\_tat = total\_tat + tat[i];

}

int s = total\_wt \* 1.0 / n;

int t = total\_tat \* 1.0 / n;

printf("Average waiting time: %d", s);

printf("\nAverage turn around time: %d ", t);

}

int main()

{

#ifndef ONLINE\_JUDGE

freopen("C:\\Users\\KIIT\\input", "r", stdin);

freopen("C:\\Users\\KIIT\\output", "w", stdout);

#endif

int n;

scanf("%d", &n);

int processes[n], burst\_time[n];

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

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

scanf("%d", &burst\_time[i]);

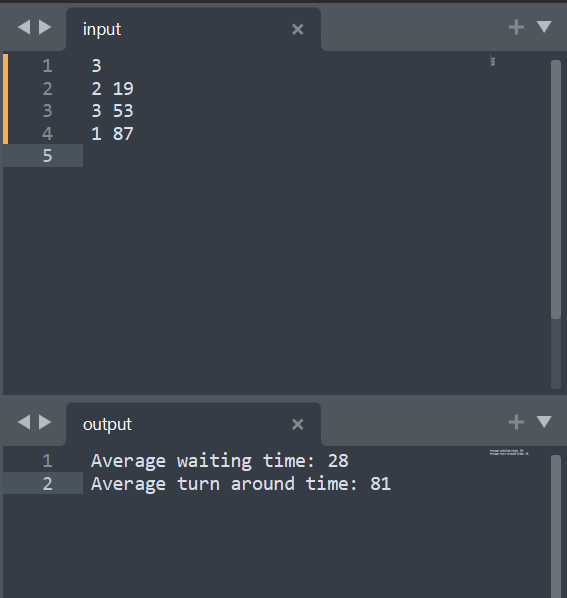
}

findavgTime(processes, n, burst\_time);

return 0;

}

Output:



2. SJF (Preemptive)

Code:

// Author: Chaudhary Hamdan, 1905387

#include <stdio.h>

int main()

{

#ifndef ONLINE\_JUDGE

freopen("C:\\Users\\KIIT\\input", "r", stdin);

freopen("C:\\Users\\KIIT\\output", "w", stdout);

#endif

int arrival\_time[10], burst\_time[10], temp[10];

int i, smallest, count = 0, time, limit;

double wait\_time = 0, turnaround\_time = 0, end;

float average\_waiting\_time, average\_turnaround\_time;

scanf("%d", &limit);

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

{

scanf("%d", &arrival\_time[i]);

scanf("%d", &burst\_time[i]);

temp[i] = burst\_time[i];

}

burst\_time[9] = 9999;

for (time = 0; count < limit; time++)

{

smallest = 9;

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

{

if (arrival\_time[i] <= time && burst\_time[i] < burst\_time [smallest] && burst\_time[i] > 0)

{

smallest = i;

}

}

burst\_time[smallest]--;

if (burst\_time[smallest] == 0)

{

count++;

end = time + 1;

wait\_time = wait\_time + end - arrival\_time[smallest] - temp[smallest];

turnaround\_time = turnaround\_time + end - arrival\_time[smallest];

}

}

average\_waiting\_time = wait\_time / limit;

average\_turnaround\_time = turnaround\_time / limit;

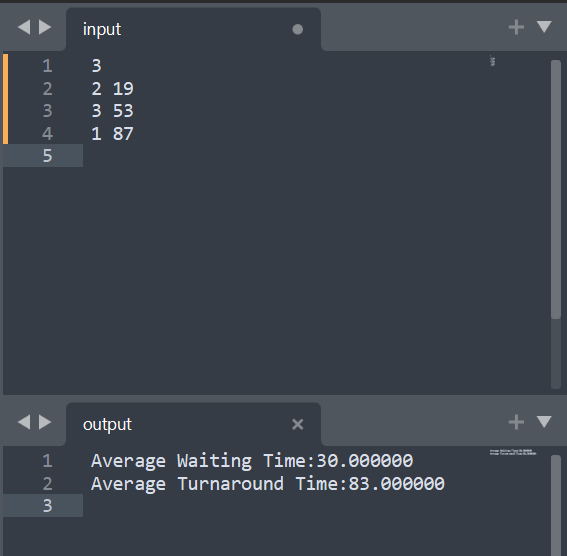
printf("Average Waiting Time:%lf\n", average\_waiting\_time);

printf("Average Turnaround Time:%lf\n", average\_turnaround\_time);

return 0;

}

Output:



3. Shortest Job First (Non Preemptive)

Code:

// Author: Chaudhary Hamdan, 1905387

#include<stdio.h>

int main()

{

#ifndef ONLINE\_JUDGE

freopen("C:\\Users\\KIIT\\input", "r", stdin);

freopen("C:\\Users\\KIIT\\output", "w", stdout);

#endif

int i, n, p[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}, min, k = 1, btime = 0;

int bt[10], temp, j, at[10], wt[10], tt[10], ta = 0, sum = 0;

float wavg = 0, tavg = 0, tsum = 0, wsum = 0;

scanf("%d", &n);

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

{

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

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

}

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

{

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

{

if (at[i] < at[j])

{

temp = p[j];

p[j] = p[i];

p[i] = temp;

temp = at[j];

at[j] = at[i];

at[i] = temp;

temp = bt[j];

bt[j] = bt[i];

bt[i] = temp;

}

}

}

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

{

btime = btime + bt[j];

min = bt[k];

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

{

if (btime >= at[i] && bt[i] < min) {

temp = p[k];

p[k] = p[i];

p[i] = temp;

temp = at[k];

at[k] = at[i];

at[i] = temp;

temp = bt[k];

bt[k] = bt[i];

bt[i] = temp;

}

} k++;

}

wt[0] = 0;

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

{

sum = sum + bt[i - 1];

wt[i] = sum - at[i];

wsum = wsum + wt[i];

}

wavg = (wsum / n);

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

{

ta = ta + bt[i];

tt[i] = ta - at[i];

tsum = tsum + tt[i];

}

tavg = (tsum / n);

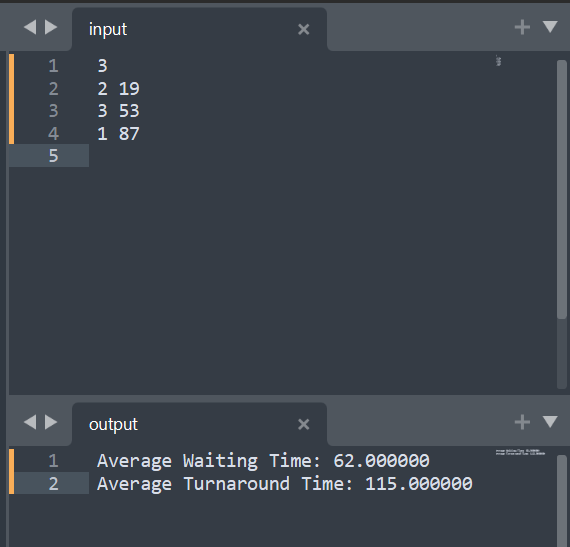
printf("Average Waiting Time: %f", wavg);

printf("\nAverage Turnaround Time: %f", tavg);

return 0;

}

Output:



4. Round Robin

Code:

// Author: Chaudhary Hamdan, 1905387

#include<stdio.h>

int main()

{

#ifndef ONLINE\_JUDGE

freopen("C:\\Users\\KIIT\\input", "r", stdin);

freopen("C:\\Users\\KIIT\\output", "w", stdout);

#endif

int i, limit, total = 0, x, counter = 0, time\_quantum;

int wait\_time = 0, turnaround\_time = 0, arrival\_time[10], burst\_time

[10], temp[10];

float average\_wait\_time, average\_turnaround\_time;

scanf("%d", &limit);

x = limit;

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

{

scanf("%d", &arrival\_time[i]);

scanf("%d", &burst\_time[i]);

temp[i] = burst\_time[i];

}

scanf("%d", &time\_quantum);

for (total = 0, i = 0; x != 0;)

{

if (temp[i] <= time\_quantum && temp[i] > 0)

{

total = total + temp[i];

temp[i] = 0;

counter = 1;

}

else if (temp[i] > 0)

{

temp[i] = temp[i] - time\_quantum;

total = total + time\_quantum;

}

if (temp[i] == 0 && counter == 1)

{

x--;

wait\_time = wait\_time + total - arrival\_time[i] -

burst\_time[i];

turnaround\_time = turnaround\_time + total -

arrival\_time[i];

counter = 0;

}

if (i == limit - 1)

{

i = 0;

}

else if (arrival\_time[i + 1] <= total)

{

i++;

}

else

{

i = 0;

}

}

average\_wait\_time = wait\_time \* 1.0 / limit;

average\_turnaround\_time = turnaround\_time \* 1.0 / limit;

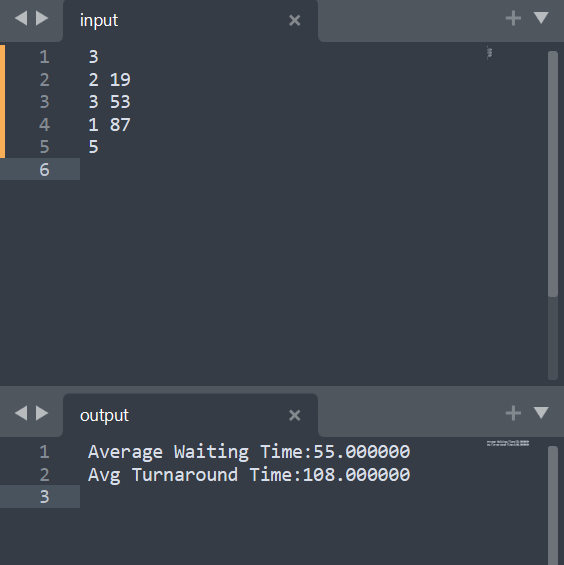
printf("Average Waiting Time:%f", average\_wait\_time);

printf("\nAvg Turnaround Time:%f\n", average\_turnaround\_time);

return 0;

}

Output:



Average Waiting Time:

FCFS: 28

SJF Preemptive: 30

SJF Non Preemptive: 62

Round Robin: 55

Graph:

