# Bansilal Ramnath Agarwal Charitable Trust’s

Vishwakarma Institute of Technology, Pune-37

*(Anautonomous Institute of Savitribai Phule Pune University)*



**Department of Multidisciplinary Engineering**

|  |  |
| --- | --- |
| **Division** | **CS-A** |
| **Batch** | **B1** |
| **Roll no.** | **90** |
| **Name** | **Aditya Shrinivas Kurapati** |
| **Subject** | **OS** |

1. **FCFS SEHEDULING Algorithm Using C.**

#include <stdio.h>

typedef struct fcfs {

int process; // Process Number

int burst; // Burst Time

int arrival; // Arrival Time

int tat; // Turn Around Time

int wt; // Waiting Time

} fcfs;

int sort(fcfs[], int);

int main() {

int n, i, temp = 0, AvTat = 0, AvWt = 0;

printf("Enter the number of processes: ");

scanf("%d", &n);

fcfs arr[n]; // Array of type fcfs

int tct[n];

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

arr[i].process = i;

printf("Enter the process %d data\n", arr[i].process);

printf("Enter CPU Burst: ");

scanf("%d", &(arr[i].burst));

printf("Enter the arrival time: ");

scanf("%d", &(arr[i].arrival));

}

// Sorting the processes according to their arrival time

sort(arr, n);

printf(

"Process\t\tBurst Time\tArrival Time\tTurn Around Time\tWaiting Time\n");

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

tct[i] = temp + arr[i].burst;

temp = tct[i];

arr[i].tat = tct[i] - arr[i].arrival;

arr[i].wt = arr[i].tat - arr[i].burst;

AvTat = AvTat + arr[i].tat;

AvWt = AvWt + arr[i].wt;

printf("%5d\t%15d\t\t%9d\t%12d\t%12d\n", arr[i].process, arr[i].burst,

arr[i].arrival, arr[i].tat, arr[i].wt);

// Print when process completes its execution

printf("Process %d completes execution at time %d\n", arr[i].process,

tct[i]);

}

printf("Average Turn Around Time: %d\nAverage Waiting Time: %d\n", AvTat / n,

AvWt / n);

return 0;

}

// Bubble Sort

int sort(fcfs arr[], int n) {

int i, j;

fcfs k;

for (i = 0; i < n - 1; i++) {

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

// Sorting the processes according to their arrival time

if (arr[i].arrival > arr[j].arrival) {

k = arr[i];

arr[i] = arr[j];

arr[j] = k;

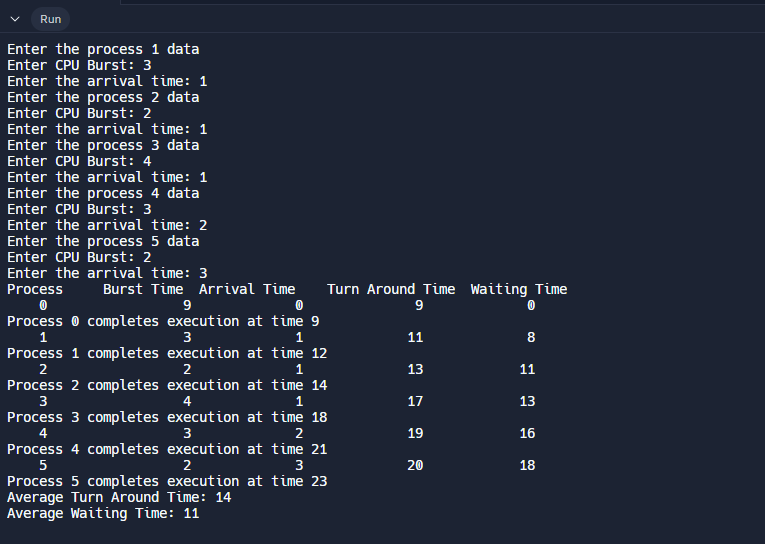
}

}

}

return 0;

}



**2.SJF Seheduling Using C.**

#include <stdio.h>

typedef struct sjf {

int process;

int burst;

int arrival;

int tat;

int wt;

} sjf;

void sort(sjf[], int);

int main() {

int n, i, j, TCT, count\_process = 0, count = 0, minBurst, pos;

float AvTAT = 0.0, AvWT = 0.0;

printf("Enter the number of processes: ");

scanf("%d", &n);

sjf arr[n];

printf("Enter the data of processes\n");

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

arr[i].process = i + 1;

printf("Enter the burst time of process %d: ", arr[i].process);

scanf("%d", &(arr[i].burst));

printf("Enter the arrival time of process %d: ", arr[i].process);

scanf("%d", &(arr[i].arrival));

}

sort(arr, n);

printf("\nPROCESS ARRIVAL TIME BURST TIME\n");

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

printf("%3d\t\t%5d\t\t%5d\n", arr[i].process, arr[i].arrival, arr[i].burst);

TCT = arr[0].tat = arr[0].burst;

arr[0].wt = arr[0].tat - arr[0].burst;

arr[0].arrival = -1;

sort(arr, n);

count\_process = 1;

while (count\_process < n) {

minBurst = 999;

count = 0;

i = count\_process;

while (TCT >= arr[i].arrival && i < n) {

count++;

i++;

}

if (count == 0) {

TCT = arr[i].arrival; // Adjust TCT if no process arrives at this time

continue;

}

for (j = i - count; count != 0 && j < n; j++, count--) {

if (arr[j].burst < minBurst) {

minBurst = arr[j].burst;

pos = j;

}

}

TCT = TCT + arr[pos].burst;

arr[pos].tat = TCT - arr[pos].arrival;

arr[pos].wt = arr[pos].tat - arr[pos].burst;

arr[pos].arrival = -1;

sort(arr, n);

count\_process++;

// Display when each process completes its execution

printf("Process %d completes execution at time %d\n", arr[pos].process,

TCT);

}

printf("\nProcess TAT WT\n");

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

printf("%2d\t\t%2d\t\t%2d\n", arr[i].process, arr[i].tat, arr[i].wt);

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

AvTAT = AvTAT + arr[i].tat;

AvWT = AvWT + arr[i].wt;

}

printf("\nAverage TAT: %.2f\nAverage WT: %.2f\n", AvTAT / n, AvWT / n);

return 0;

}

void sort(sjf arr[], int n) {

int i, j;

sjf temp;

for (i = 0; i < n - 1; i++)

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

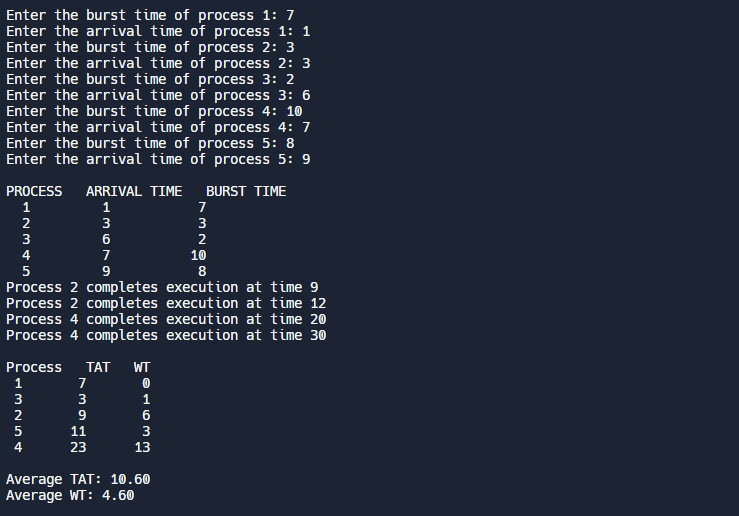
if (arr[i].arrival > arr[j].arrival) {

temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

}  


**3.Shortest Remaining Time Next (SJF Preemptive Algorithm)**

#include <stdio.h>

#include <stdlib.h>

#define MAX 20

int main() {

int a[MAX][7], i, count = 0, totalt, small, n;

float awt, atat;

printf("Enter no of processes: ");

scanf("%d", &n);

printf("\nEnter process name, arrival time and burst time: ");

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

printf("\nProcess name: ");

scanf("%d", &a[i][0]);

printf("\nArrival time: ");

scanf("%d", &a[i][1]);

printf("\nBurst time: ");

scanf("%d", &a[i][2]);

count += a[i][2];

a[i][6] = -1;

}

count = count + a[0][1];

totalt = a[0][1];

while (totalt < count) {

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

if (a[i][6] == -1 && a[i][1] <= totalt) {

small = i;

break;

}

}

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

if (a[i][6] == -1 && a[i][1] <= totalt) {

if (a[small][2] > a[i][2])

small = i;

/\*else if(small==i)

small=i;\*/

}

}

totalt = totalt + a[small][2]; // updation total time

a[small][3] = totalt; // ct of process

a[small][6] = 0; // flag for process status

// printf("\nTime %d",totalt);

}

atat = 0.0;

awt = 0.0;

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

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

printf("\n %d", a[i][0]);

printf("\t %d", a[i][1]);

printf("\t %d", a[i][2]);

printf("\t %d", a[i][3]);

a[i][4] = a[i][3] - a[i][1];

printf("\t %d", a[i][4]);

a[i][5] = a[i][4] - a[i][2];

printf("\t %d", a[i][5]);

awt = awt + a[i][5];

atat = atat + a[i][4];

}

atat = atat / n;

awt = awt / n;

printf("\nAverage TAT: %f", atat);

printf("\nAverage WT: %f", awt);

return 0;

}  
