**Lab assignment 5**

1. First Come First Serve (FCFS)

#include <iostream>

using namespace std;

int main() {

int n, bt[20], wt[20], tat[20];

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

cout << "Enter total number of processes: ";

cin >> n;

cout << "Enter burst time for each process:\n";

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

cout << "P[" << i + 1 << "]: ";

cin >> bt[i];

}

wt[0] = 0;

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

wt[i] = 0;

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

wt[i] += bt[j];

}

cout << "\nProcess\tBT\tWT\tTAT";

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

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

avg\_wt += wt[i];

avg\_tat += tat[i];

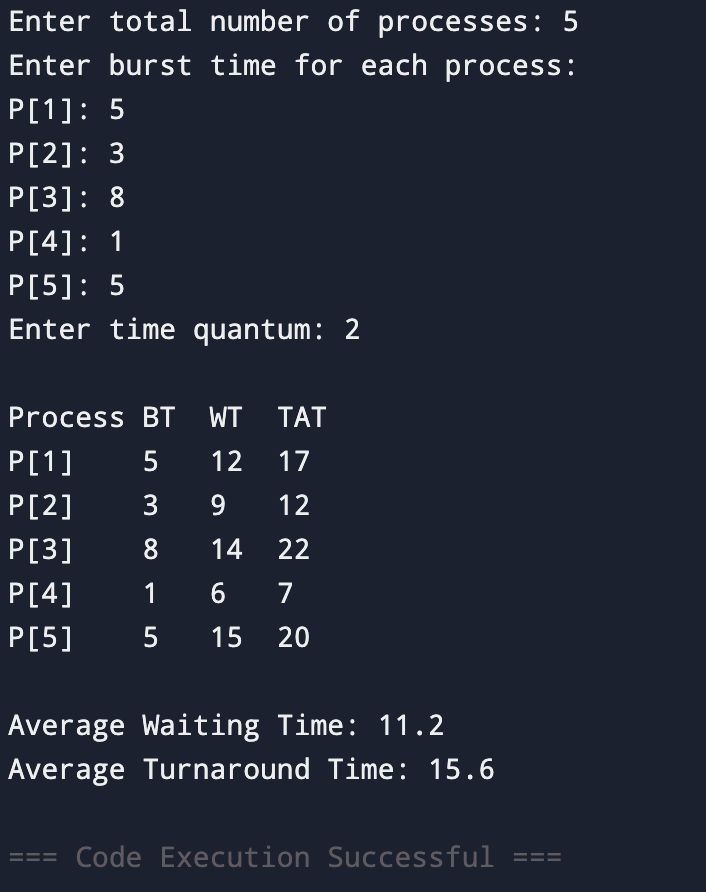
cout << "\nP[" << i + 1 << "]\t" << bt[i] << "\t" << wt[i] << "\t" << tat[i];

}

cout << "\n\nAverage Waiting Time: " << avg\_wt / n;

cout << "\nAverage Turnaround Time: " << avg\_tat / n;

return 0;

}   


**2. Shortest Job First (SJF) – Non-preemptive**

#include <iostream>

using namespace std;

int main() {

int n, bt[20], p[20], wt[20], tat[20], i, j, temp;

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

cout << "Enter total number of processes: ";

cin >> n;

cout << "Enter burst time for each process:\n";

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

cout << "P[" << i + 1 << "]: ";

cin >> bt[i];

p[i] = i + 1;

}

// Sorting burst times

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

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

if (bt[i] > bt[j]) {

temp = bt[i]; bt[i] = bt[j]; bt[j] = temp;

temp = p[i]; p[i] = p[j]; p[j] = temp;

}

}

}

wt[0] = 0;

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

wt[i] = 0;

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

wt[i] += bt[j];

}

cout << "\nProcess\tBT\tWT\tTAT";

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

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

avg\_wt += wt[i];

avg\_tat += tat[i];

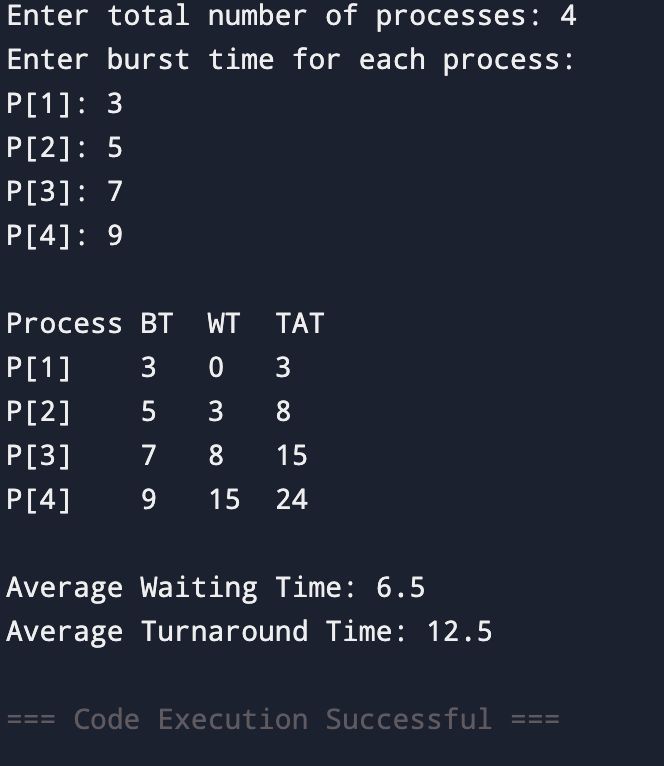
cout << "\nP[" << p[i] << "]\t" << bt[i] << "\t" << wt[i] << "\t" << tat[i];

}

cout << "\n\nAverage Waiting Time: " << avg\_wt / n;

cout << "\nAverage Turnaround Time: " << avg\_tat / n;

return 0;

}

3. **Round Robin Scheduling**

#include <iostream>

using namespace std;

int main() {

int n, time\_quantum, remaining[10], bt[10], wt[10], tat[10], t = 0;

bool done;

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

cout << "Enter total number of processes: ";

cin >> n;

cout << "Enter burst time for each process:\n";

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

cout << "P[" << i + 1 << "]: ";

cin >> bt[i];

remaining[i] = bt[i];

}

cout << "Enter time quantum: ";

cin >> time\_quantum;

do {

done = true;

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

if (remaining[i] > 0) {

done = false;

if (remaining[i] > time\_quantum) {

t += time\_quantum;

remaining[i] -= time\_quantum;

} else {

t += remaining[i];

wt[i] = t - bt[i];

remaining[i] = 0;

}

}

}

} while (!done);

cout << "\nProcess\tBT\tWT\tTAT";

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

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

avg\_wt += wt[i];

avg\_tat += tat[i];

cout << "\nP[" << i + 1 << "]\t" << bt[i] << "\t" << wt[i] << "\t" << tat[i];

}

cout << "\n\nAverage Waiting Time: " << avg\_wt / n;

cout << "\nAverage Turnaround Time: " << avg\_tat / n;

return 0;

}

