**Round Robin(RR):**

#include<bits/stdc++.h>

#include <algorithm>

#include <iomanip>

#include <queue>

#include <cstring>

using namespace std;

struct process {

int arrivalTime;

int burstTime;

int completionTime;

int turnTime;

int waitingTime;

int startTime;

};

int main() {

int n, tq;

struct process p[50];

float avgWaiting;

int totalTurn = 0, burstArr[50];

float avgTurn;

int totalWaiting = 0, index;

cout << setprecision(2) << fixed;

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

cin >> n;

cout << "Enter time quantum of CPU: ";

cin >> tq;

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

cout << "Enter arrival time of process " << i + 1 << ": ";

cin >> p[i].arrivalTime;

cout << "Enter burst time of process " << i + 1 << ": ";

cin >> p[i].burstTime;

burstArr[i] = p[i].burstTime;

cout << endl;

}

queue < int > q;

int current\_time = 0;

q.push(0);

int completed = 0;

int mark[100];

memset(mark, 0, sizeof(mark));

mark[0] = 1;

while (completed != n) {

index = q.front();

q.pop();

if (burstArr[index] == p[index].burstTime) {

p[index].startTime = max(current\_time, p[index].arrivalTime);

current\_time = p[index].startTime;

}

if (0 < burstArr[index] - tq) {

burstArr[index] -= tq;

current\_time += tq;

} else {

current\_time += burstArr[index];

p[index].completionTime = current\_time;

p[index].turnTime = p[index].completionTime - p[index].arrivalTime;

p[index].waitingTime = p[index].turnTime - p[index].burstTime;

totalWaiting += p[index].waitingTime;

totalTurn += p[index].turnTime;

completed++;

burstArr[index] = 0;

}

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

if (burstArr[i] > 0 && p[i].arrivalTime <= current\_time && mark[i] == 0) {

mark[i] = 1;

q.push(i);

}

}

if (0 < burstArr[index]) q.push(index);

if (q.empty()) {

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

if (0 < burstArr[i]) {

mark[i] = 1;

q.push(i);

break;

}

}

}

}

avgWaiting = (float) totalWaiting / n;

avgTurn = (float) totalTurn / n;

cout << endl;

int totalAT=0;

for(int i=0;i<4;i++){

totalAT+=p[i].arrivalTime;

}

float avgRT= totalAT/n;

cout<<"Process Id\t"<<"Arrival Time\t"<<"Burst Time\t"<<"Completion Time\t"<<" Turn around Time\t"<<"Waiting Time\t"<<"Response Time"<<"\n"<<endl;

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

cout<<i+1<<"\t"<<"\t"<<p[i].arrivalTime<<"\t"<<"\t"<<p[i].burstTime<<"\t"<<"\t"<<p[i].completionTime<<"\t"<<"\t"<<"\t"<<p[i].turnTime<<"\t"<<"\t"<<p[i].waitingTime<<"\t"<<"\t"<<"\t"<<p[i].arrivalTime<<"\n"<<endl;

}

cout << "Average Waiting Time = " << avgWaiting << endl;

cout << "Average Turnaround Time = " << avgTurn << endl;

cout<<"Average Response Time = "<<avgRT<<endl;

return 0;

}

**OUTPUT:**

Enter the number of processes: 4

Enter time quantum of CPU: 2

Enter arrival time of process 1: 0

Enter burst time of process 1: 5

Enter arrival time of process 2: 1

Enter burst time of process 2: 4

Enter arrival time of process 3: 2

Enter burst time of process 3: 2

Enter arrival time of process 4: 4

Enter burst time of process 4: 1

Process Id AT BT CT TAT WT RT

1 0 5 12 12 7 0

2 1 4 11 10 6 1

3 2 2 6 4 2 2

4 4 1 9 5 4 4

Average Waiting Time = 4.75

Average Turnaround Time = 7.75

Average Response Time = 1.00