**OS LAB WEEK 4**

**-> yapara karthikeya**

**-> 1BM21CS249**

**-> 28/06/23**

**Q: Round Robin using C**

#include <stdio.h>

#define MAX\_SIZE 100

struct Process {

int pid;

int burst\_time;

int remaining\_time;

int waiting\_time;

};

void RoundRobin(struct Process processes[], int n, int time\_quantum) {

int total\_time = 0;

int completed = 0;

int ready\_queue[MAX\_SIZE];

int front = 0, rear = -1;

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

ready\_queue[++rear] = i;

}

while (completed < n) {

int current\_process = ready\_queue[front++];

if (processes[current\_process].remaining\_time > 0) {

if (processes[current\_process].remaining\_time <= time\_quantum) {

total\_time += processes[current\_process].remaining\_time;

processes[current\_process].remaining\_time = 0;

} else {

total\_time += time\_quantum;

processes[current\_process].remaining\_time -= time\_quantum;

}

printf("Time %d: Process %d\n", total\_time, processes[current\_process].pid);

}

if (processes[current\_process].remaining\_time == 0) {

completed++;

processes[current\_process].waiting\_time = total\_time - processes[current\_process].burst\_time;

} else {

ready\_queue[++rear] = current\_process;

}

}

}

int main() {

int n;

int time\_quantum;

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

scanf("%d", &n);

printf("Enter the time quantum: ");

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

struct Process processes[n];

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

printf("Enter burst time for process %d: ", i + 1);

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

processes[i].remaining\_time = processes[i].burst\_time;

processes[i].pid = i + 1;

}

printf("Scheduling order:\n");

RoundRobin(processes, n, time\_quantum);

double total\_waiting\_time = 0;

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

total\_waiting\_time += processes[i].waiting\_time;

}

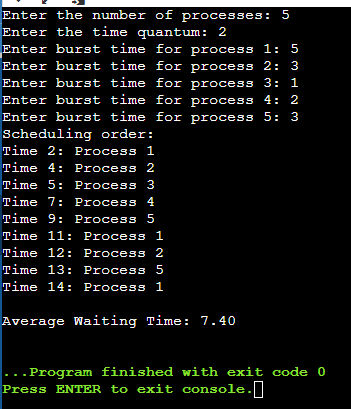
double avg\_waiting\_time = total\_waiting\_time / n;

printf("\nAverage Waiting Time: %.2lf\n", avg\_waiting\_time);

return 0;

}

**Output**

****