**Operating System Assignment**

*Mini Project submitted in partial fulfilment of the requirements for the Degree of*

**BACHELORS OF TECHNOLOGY**

**in**

**COMPUTER SCIENCE AND ENGINEERING**

By

**KARAN DIWAN**

**11714605**

**K17ZH | 31**

Under the Guidance of

Mr. Pardeep Kumar

**ASSISSTANT PROFESSOR**

**School of Computer Science and Engineering**

Lovely Professional University

Phagwara, Punjab (India)

**Email:**

**Karan.diwan170898@gmail.com**

**GitHub Link:**

1. **Explain the problem in terms of Operating System Concepts.**

In **Round Robin** algorithm each process is processed or executed for a particular period of time. This time is given by the user at the run time or is fixed and is known as **Time Quantum.** This assignment consists of a round robin algorithm with non-pre-emption.

It asks for number of processes and their **Arrival and Burst time** respectively and after applying the algorithm it gives out the **Turn Around Time and Waiting time** for each process.

1. **Write the algorithm for the proposed solution of the assigned problem.**

**Steps to find Waiting Time:**

1- Create an array rem\_bt[] to keep track of remaining

burst time of processes. This array is initially a

copy of bt[] (burst times array)

2- Create another array wt[] to store waiting times

of processes. Initialize this array as 0.

3- Initialize time : t = 0

4- Keep traversing the all processes while all processes

are not done. Do following for i'th process if it is

not done yet.

a- If rem\_bt[i] > quantum

(i) t = t + quantum

(ii) bt\_rem[i] -= quantum;

c- Else // Last cycle for this process

(i) t = t + bt\_rem[i];

(ii) wt[i] = t - bt[i]

(ii) bt\_rem[i] = 0; // This process is over

**Function for finding Average Time :**

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

**int quantum)**

**{**

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

**// Function to find waiting time of all processes**

**findWaitingTime(processes, n, bt, wt, quantum);**

**// Function to find turn around time for all processes**

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

**// Display processes along with all details**

**cout << "Processes "<< " Burst time "**

**<< " Waiting time " << " Turn around time\n";**

**// Calculate total waiting time and total turn**

**// around time**

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

**{**

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

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

**cout << " " << i+1 << "\t\t" << bt[i] <<"\t "**

**<< wt[i] <<"\t\t " << tat[i] <<endl;**

**}**

**cout << "Average waiting time = "**

**<< (float)total\_wt / (float)n;**

**cout << "\nAverage turn around time = "**

**<< (float)total\_tat / (float)n;**

**}**

**Function for finding Completion Time :**

**time\_req = 0;**

**// Add time for process on left of p**

**// (Scheduled before p in a round of**

**// 1 unit time slice)**

**for (int i=0; i<p; i++)**

**{**

**if (arr[i] < arr[p])**

**time\_req += arr[i];**

**else**

**time\_req += arr[p];**

**}**

**// step 2 : Add time of process p**

**time\_req += arr[p];**

**// Add time for process on right**

**// of p (Scheduled after p in**

**// a round of 1 unit time slice)**

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

**{**

**if (arr[i] < arr[p])**

**time\_req += arr[i];**

**else**

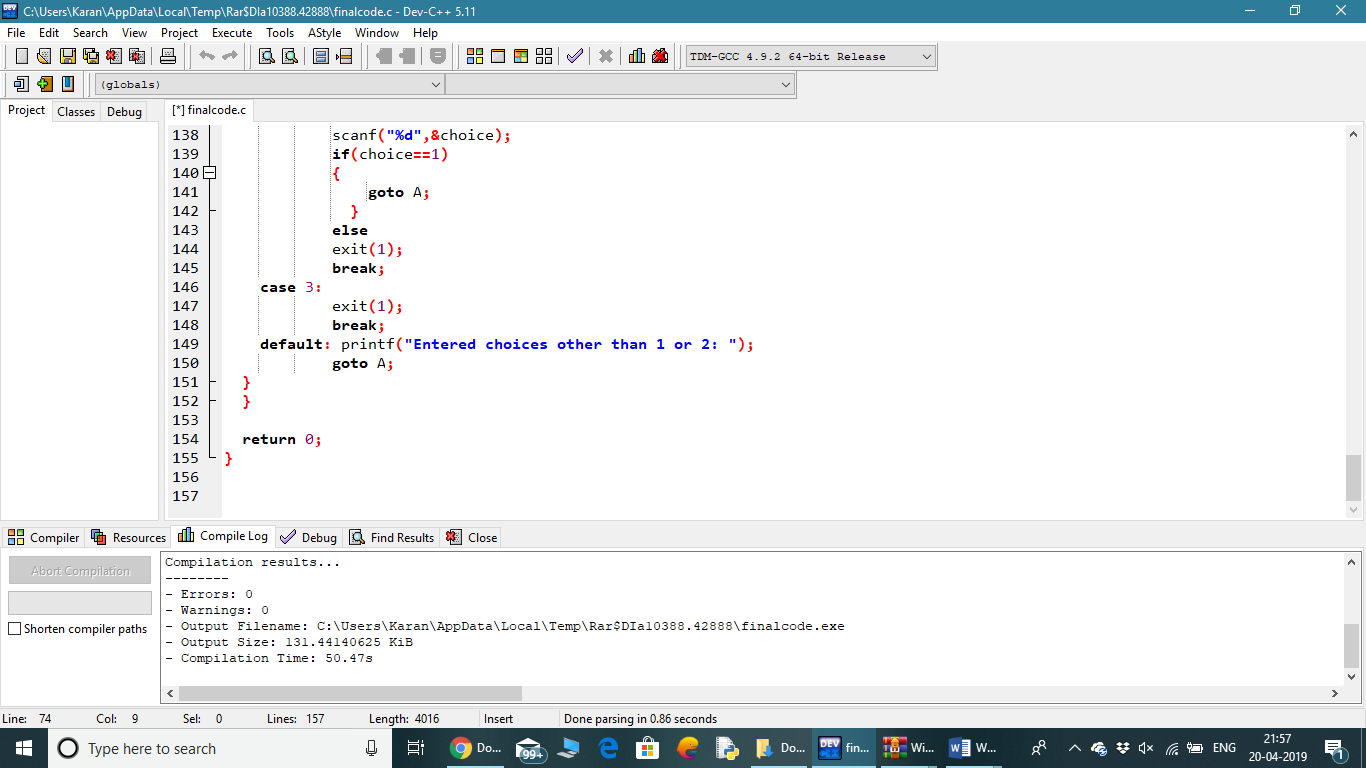
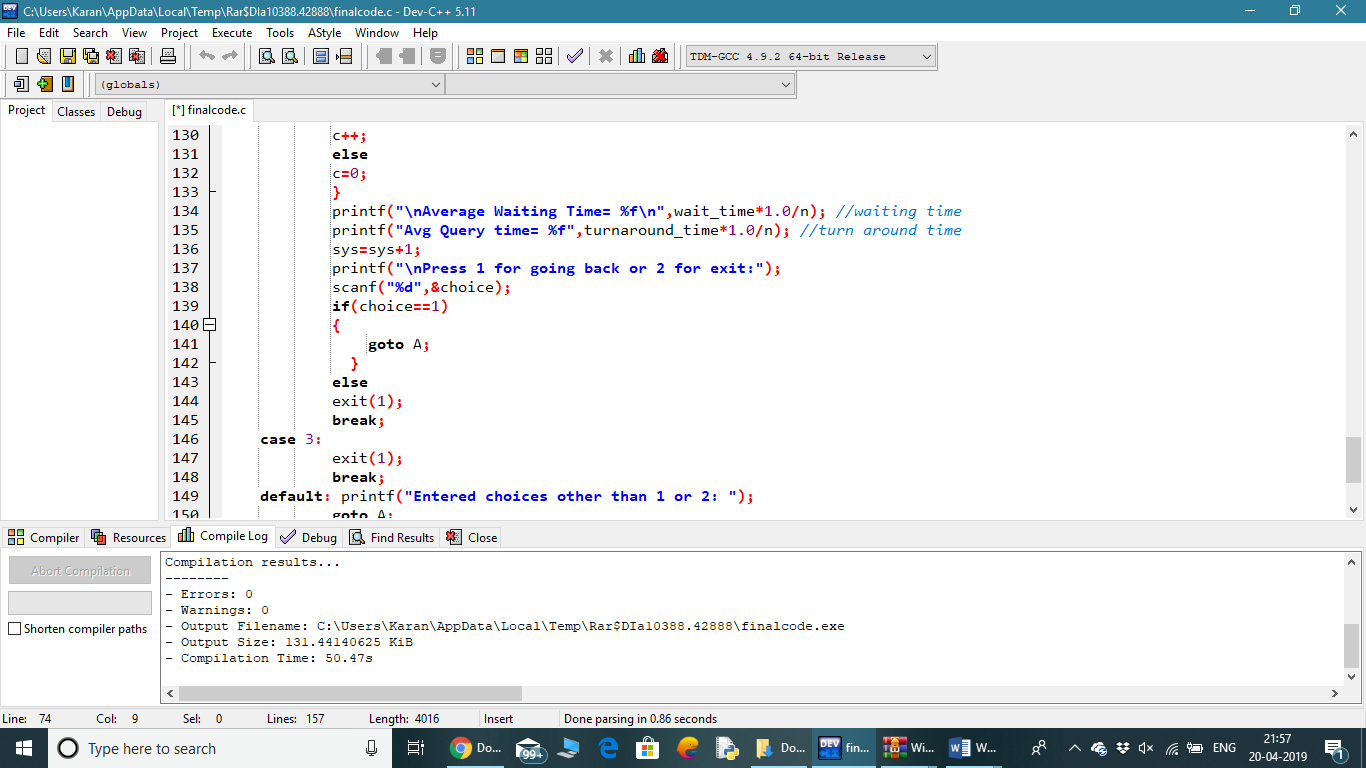
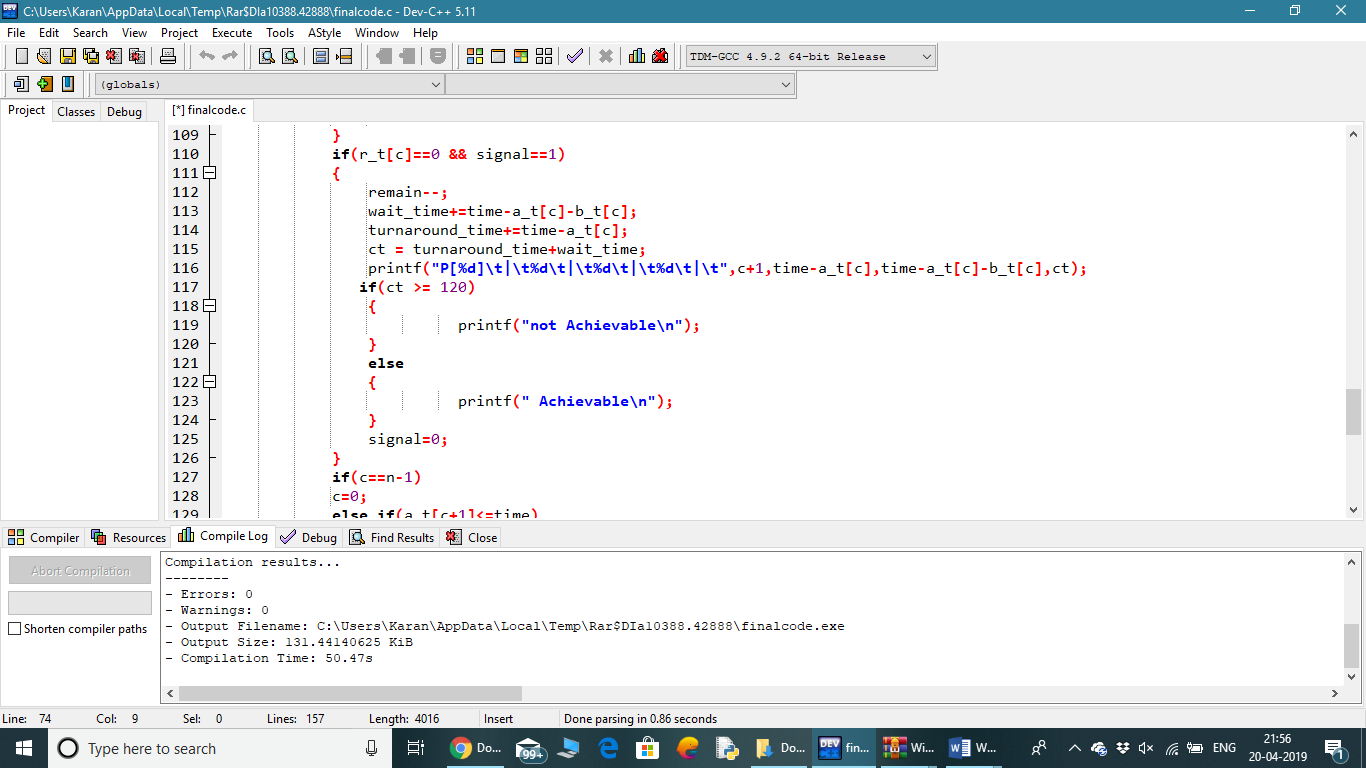
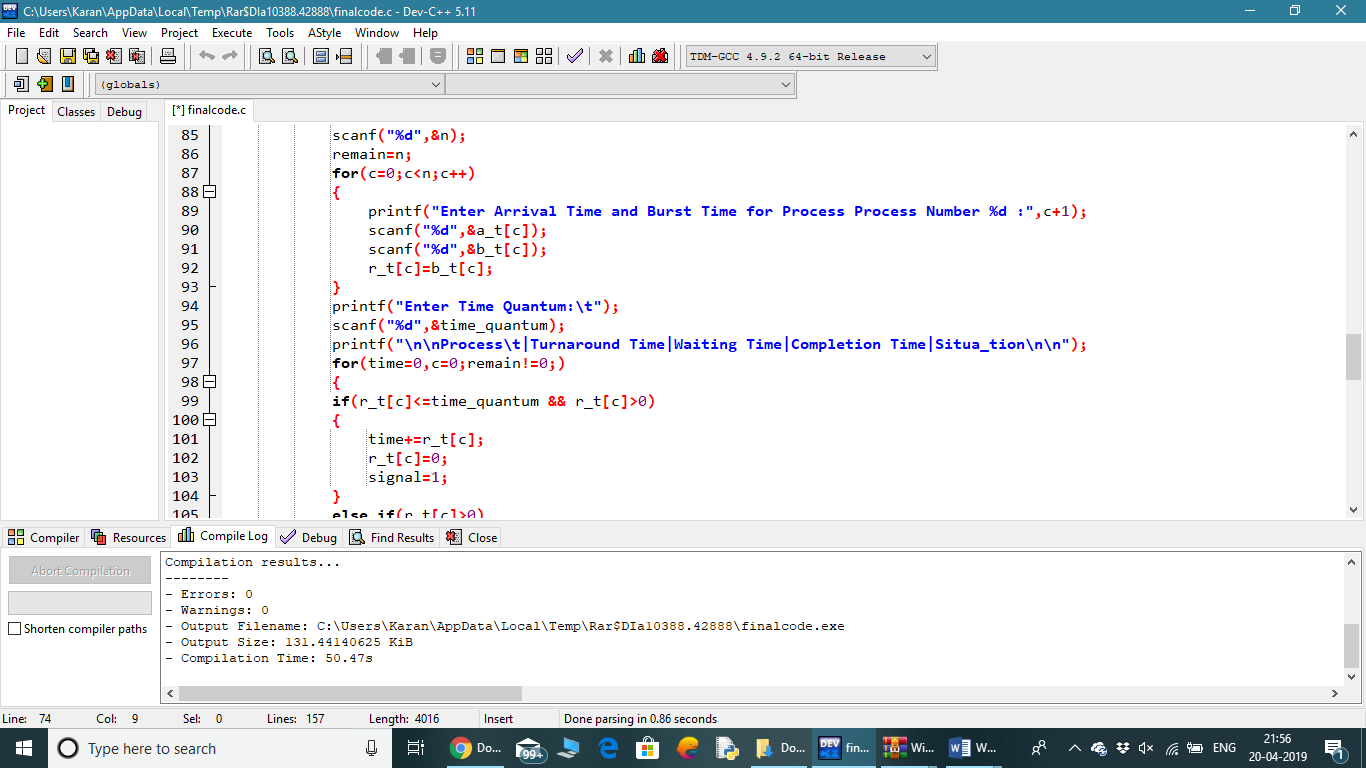
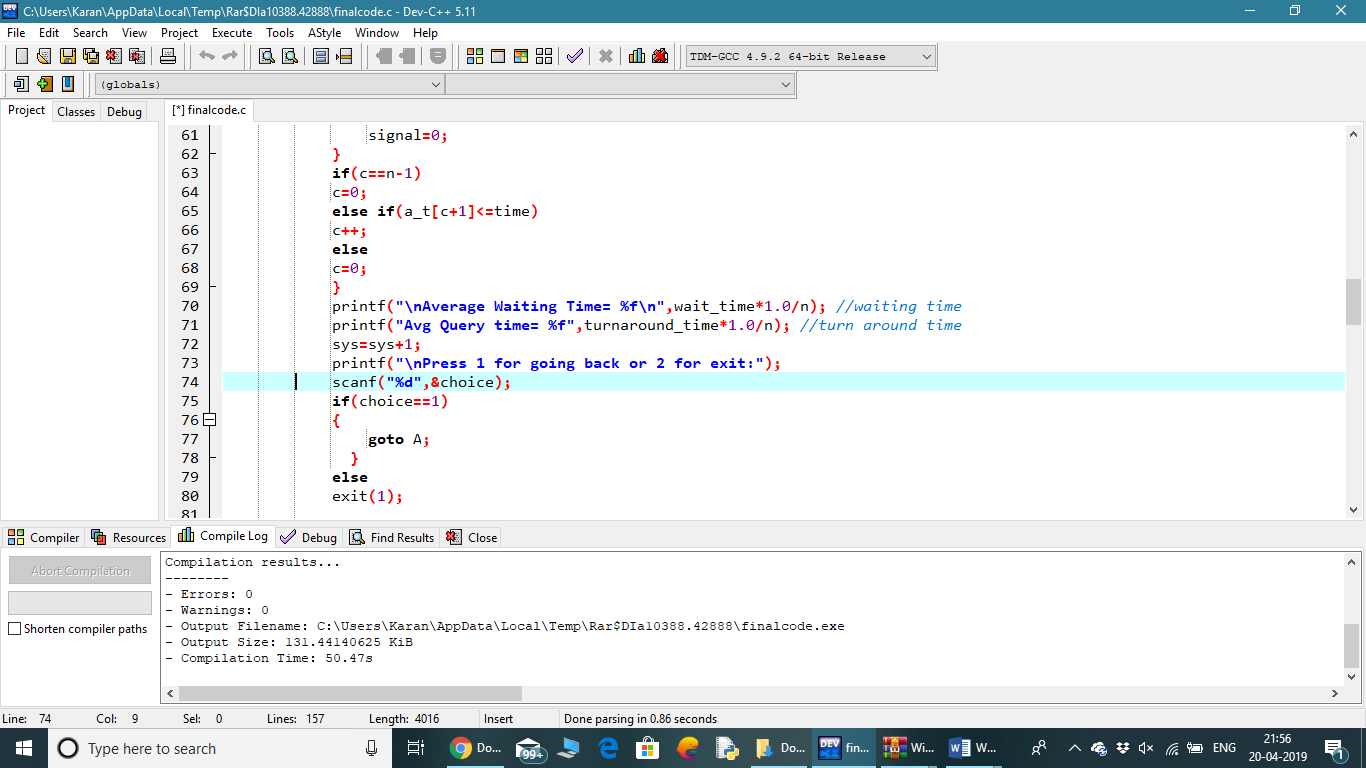
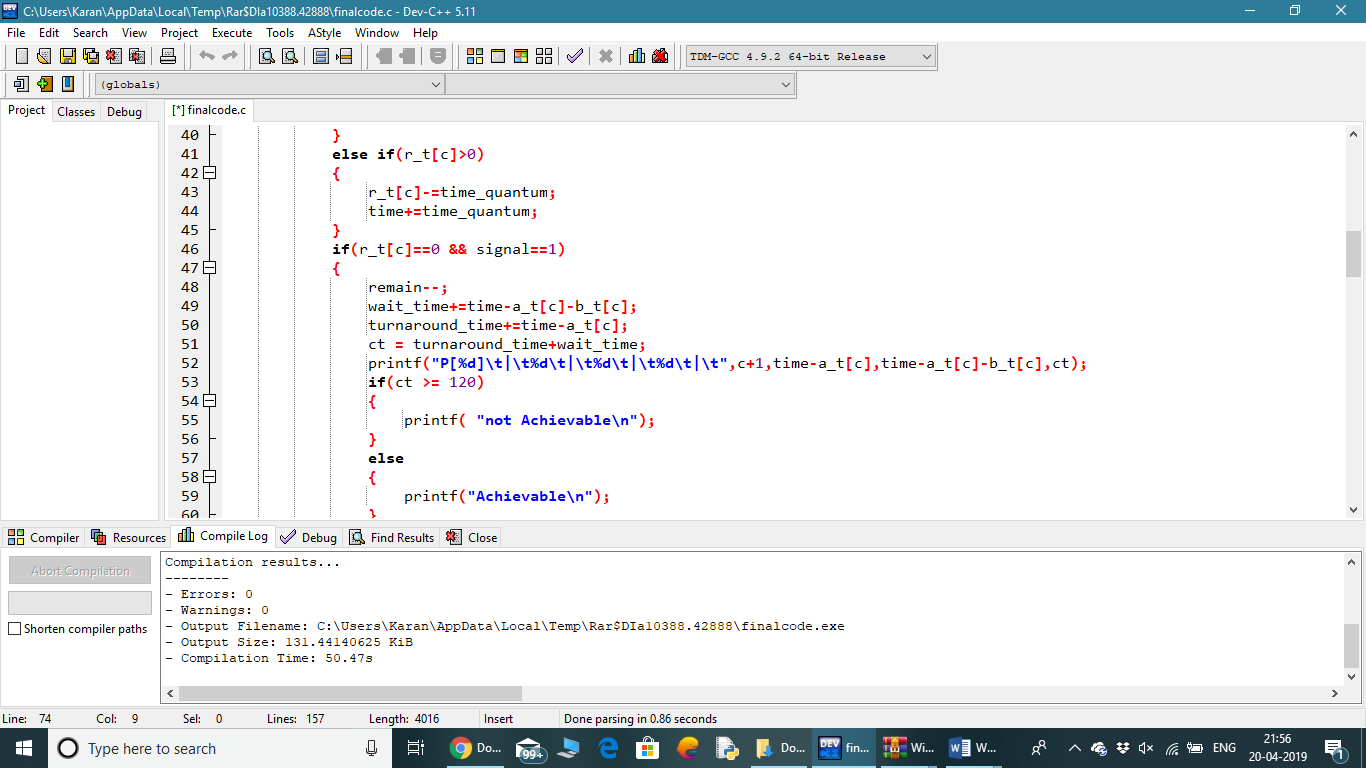
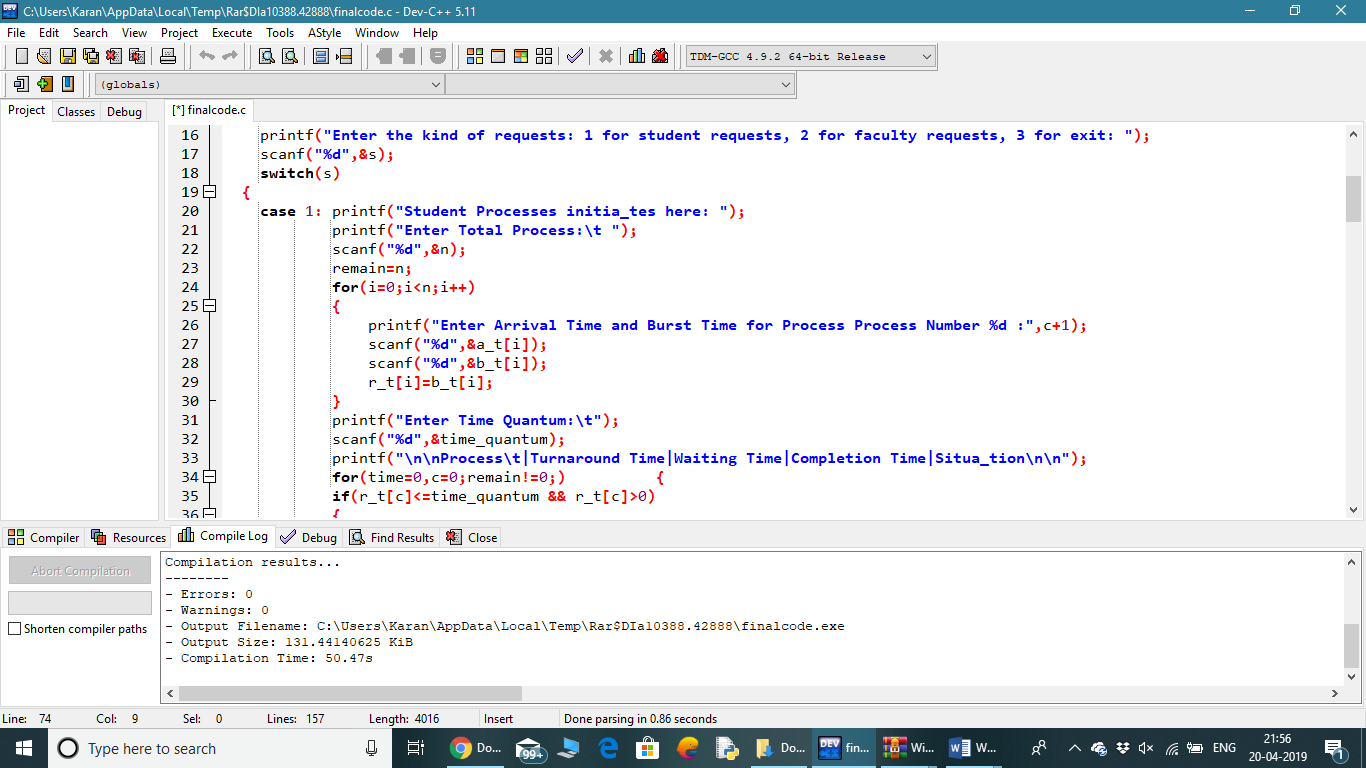
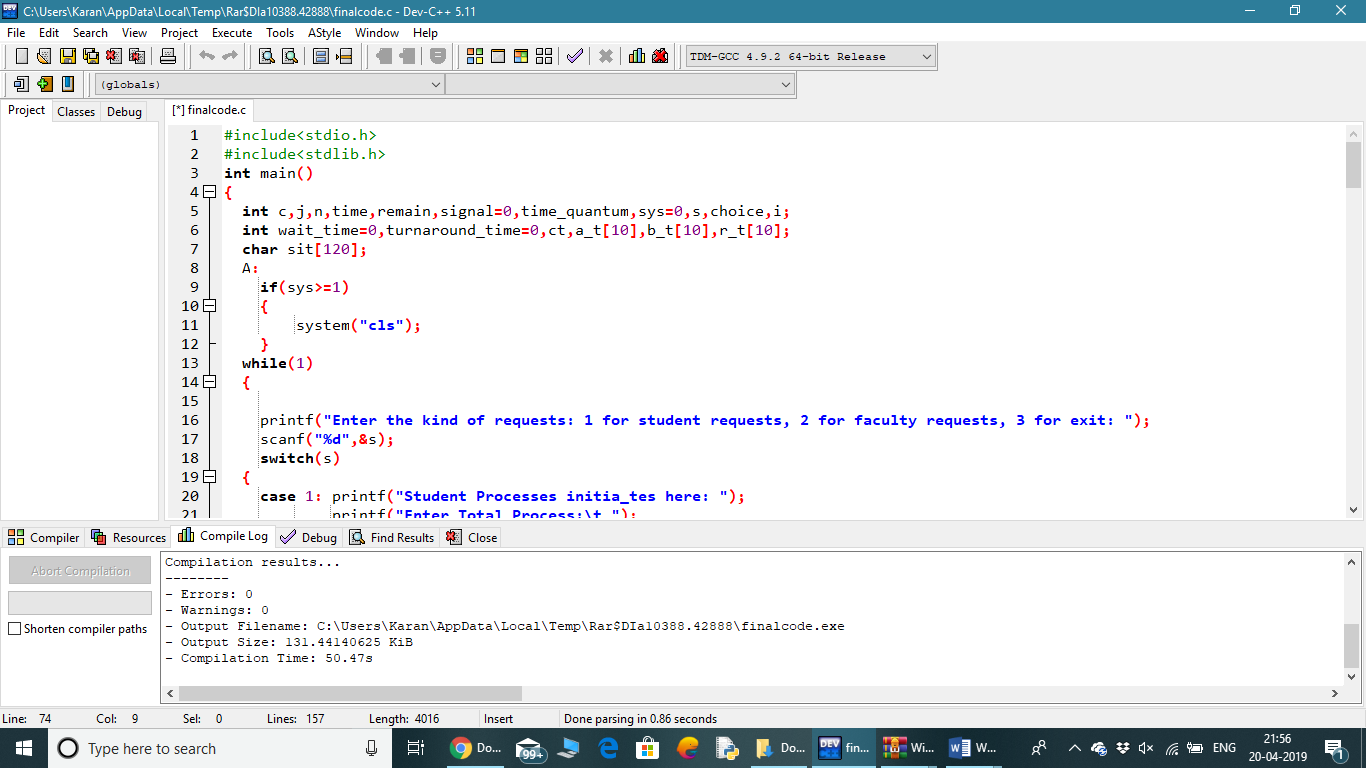
**time\_req += arr[p]-1;**

**}**

1. **Calculate complexity of implemented algorithm.**

**The basic complexity of Round Robin Algorithm is O(1) when it is run using dynamic programming but in our programme it is O(n) where n is the number of processes entered by the user.**

1. **Explain all the constraints given in the problem. Attach the code snippet of the implemented constraint.**



1. **If you have implemented any additional algorithm to support the solution, explain the need and usage of the same.**

The Completion Time Algorithm is there to support the Solution.

1. **Explain the boundary conditions of the implemented code.**
2. The boundary conditions for the Round Robin algorithm is to find the Average and Waiting time for processes.
3. On every request, the system determines whether the process entered into the system will get executed or not.
4. The Turn Around Time cannot exceed the Completion Time.
5. Turn Around Time is equal to Complete Time – Arrival Time.

**7. Explain all the test cases applied on the solution of assigned problem.**

