THE UNIVERSITY OF AZAD JAMMU AND KASHMIR, MUZAFFARABAD

Department of Software Engineering

Submitted to:	Engr. Sidra Rafique
Course Title:	Operating System
Course Code:	CS-2105
Session:	2024-2028
Lab No:	05
Roll No:	2024-SE-15
Submitted By:	Shahzad Ahmed Awan
Submission date:	November 27, 2025

Table of Contents

Lab 05 – CPU Scheduling: First Come First Served (FCFS) 3

1. Objective 3

2. Background 3

Key Terms: 3

3. Task – FCFS Scheduling Demonstration 3

3.1 Aim 3

3.2 Procedure 3

4. Source Code 4

5. Output Table 5

6. Conclusion 5

Table of Figure

Figure 1: Source Code for the implementing FCFS Algorithm 4

Figure 2: Output Table demonstrating FCFS Workflow 5

Lab 05 – CPU Scheduling: First Come First Served (FCFS)

1. Objective

To demonstrate the First Come First Served (FCFS) CPU scheduling algorithm in C++ and calculate Completion Time (CT), Turnaround Time (TAT), and Waiting Time (WT) for a set of processes.

2. Background

CPU scheduling decides which process gets CPU access when multiple processes are waiting. FCFS is non-preemptive and executes processes in order of arrival (FIFO).

Key Terms:

- **AT (Arrival Time):** Time process enters the ready queue
- **BT (Burst Time):** CPU time required by the process
- **CT (Completion Time):** Time process finishes execution
- **TAT (Turnaround Time):** CT − AT
- **WT (Waiting Time):** TAT − BT

3. Task – FCFS Scheduling Demonstration

3.1 Aim

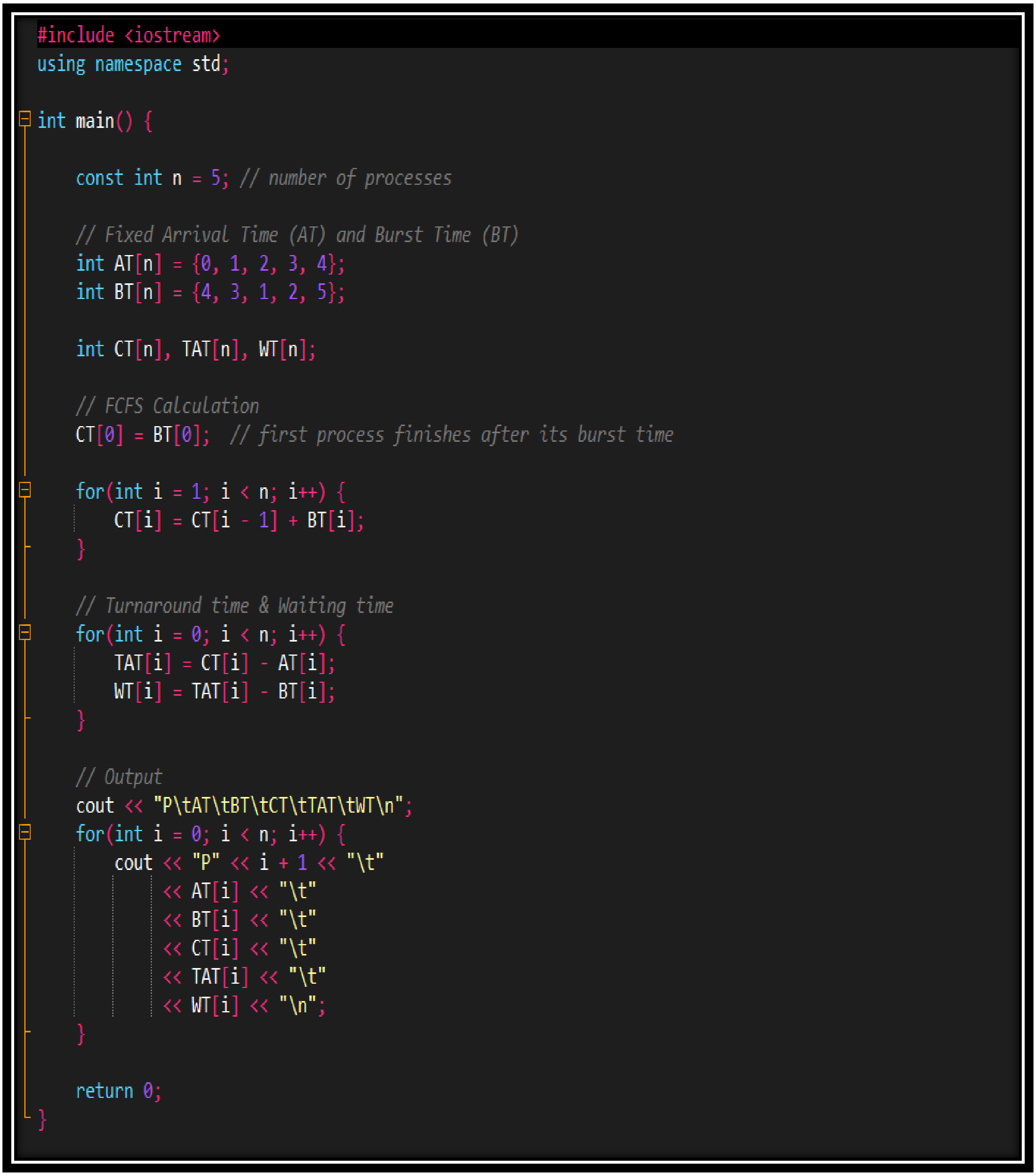
To implement FCFS scheduling in C++ and compute CT, TAT, and WT for 5 processes.

3.2 Procedure

1. Define Arrival Time (AT) and Burst Time (BT) for each process.
2. Calculate Completion Time (CT):
    - First process completes after its burst time.
    - Each subsequent process completes after the previous one finishes.
3. Compute Turnaround Time (TAT) and Waiting Time (WT) for each process.
4. Display the results in a tabular format.

4. Source Code

Refer to the attached image for the C++ source code implementation.

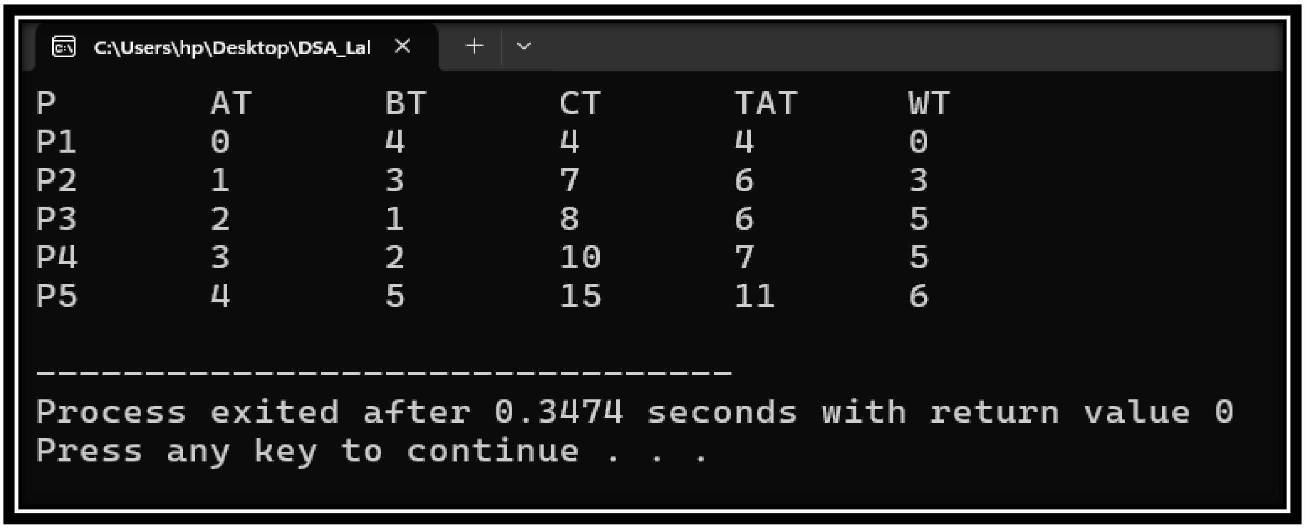


```cpp
#include <iostream>
using namespace std;

int main() {

    const int n = 5; // number of processes

    // Fixed Arrival Time (AT) and Burst Time (BT)
    int AT[n] = {0, 1, 2, 3, 4};
    int BT[n] = {4, 3, 1, 2, 5};

    int CT[n], TAT[n], WT[n];

    // FCFS Calculation
    CT[0] = BT[0];  // first process finishes after its burst time

    for(int i = 1; i < n; i++) {
        CT[i] = CT[i - 1] + BT[i];
    }

    // Turnaround time & Waiting time
    for(int i = 0; i < n; i++) {
        TAT[i] = CT[i] - AT[i];
        WT[i] = TAT[i] - BT[i];
    }

    // Output
    cout << "P\tAT\tBT\tCT\tTAT\tWT\n";
    for(int i = 0; i < n; i++) {
        cout << "P" << i + 1 << "\t"
             << AT[i] << "\t"
             << BT[i] << "\t"
             << CT[i] << "\t"
             << TAT[i] << "\t"
             << WT[i] << "\n";
    }

    return 0;
}
```

Figure 1: Source Code for the implementing FCFS Algorithm

5. Output Table

The table below shows the calculated CT, TAT, and WT for all processes as per FCFS scheduling.



```
P       AT      BT      CT      TAT     WT
P1      0       4       4       4       0
P2      1       3       7       6       3
P3      2       1       8       6       5
P4      3       2       10      7       5
P5      4       5       15      11      6

--------------------------------
Process exited after 0.3474 seconds with return value 0
Press any key to continue . . .
```

Figure 2: Output Table demonstrating FCFS Workflow

6. Conclusion

The LAB demonstrated the FCFS CPU scheduling algorithm where processes are executed in arrival order. It calculates CT, TAT, and WT efficiently, providing a clear understanding of process scheduling in operating systems.