# Course: Operating System

## Semester Project

**Submitted by:**

**Muhammad Taqui (01-136221-021)**

**Babar Ali (01-136221-051)**

**Muhammad Usman Iftikhar (01-136221-022)**

**Department of Computer Science**

**Bahria University Islamabad Campus**

## Objectives:

- Create a mini project using any concepts related to Operating system(CPU scheduling, process management, memory management, threads, context switching)

  **For this project, you need to hand in Two distinct items:**
- Your source code
- A README file with some basic documentation about your code

  You can use c or c++ language

# Operating System Mini Project - Round Robin CPU Scheduler

## Introduction:

This mini project implements a simple Round Robin CPU scheduling algorithm in C++. The round-robin scheduling algorithm is widely used in operating systems for sharing the CPU time among multiple processes. The project simulates the execution of processes, allowing users to input the burst time for each process and the time quantum for the scheduling algorithm.

## Features:

1. **Dynamic Process Creation:** The program prompts the user to input the number of processes and their respective burst times.

2. **Round Robin Scheduler:** The implemented Round Robin scheduler distributes the CPU time among processes based on the user-defined time quantum.

3. **Process Management:** The project simulates the execution of processes and displays messages when each process is scheduled, executes, and completes.

## How to Use:

### 1. Compile the Code:

Use a C++ compiler to compile the source code (e.g., g++ main.cpp -o scheduler).

### 2. Run the Executable:

Execute the compiled program (e.g., ./scheduler).

Enter the number of processes and burst time for each process as prompted.

### 3. Enter Time Quantum:

Input the time quantum for the Round Robin scheduler.

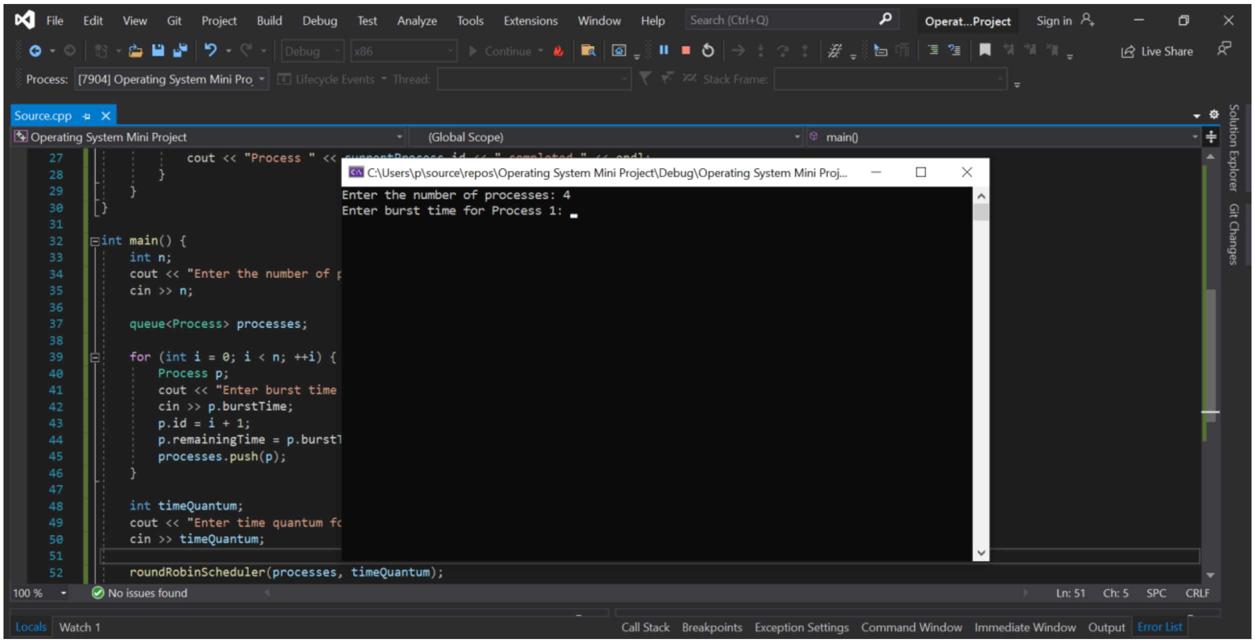
### 4. Observe Output:

The program will simulate the execution of processes and display relevant information, such as the execution of each process and its completion.

## Code:

```cpp
#include <iostream>
#include <queue>

using namespace std;

struct Process {
    int id;
    int burstTime;
    int remainingTime;
};

void roundRobinScheduler(queue<Process>& processes, int timeQuantum) {
    while (!processes.empty()) {
        Process currentProcess = processes.front();
        processes.pop();

        int executionTime = min(timeQuantum, currentProcess.remainingTime);

        cout << "Executing Process " << currentProcess.id << " for time: " <<
executionTime << " units." << endl;

        currentProcess.remainingTime -= executionTime;

        if (currentProcess.remainingTime > 0) {
            processes.push(currentProcess);
        }
        else {
            cout << "Process " << currentProcess.id << " completed." << endl;
        }
    }
}

int main() {
    int n;
    cout << "Enter the number of processes: ";
    cin >> n;

    queue<Process> processes;

    for (int i = 0; i < n; ++i) {
        Process p;
        cout << "Enter burst time for Process " << i + 1 << ": ";
        cin >> p.burstTime;
        p.id = i + 1;
        p.remainingTime = p.burstTime;
        processes.push(p);
    }

    int timeQuantum;
    cout << "Enter time quantum for Round Robin scheduling: ";
    cin >> timeQuantum;

    roundRobinScheduler(processes, timeQuantum);

    return 0;
}
```

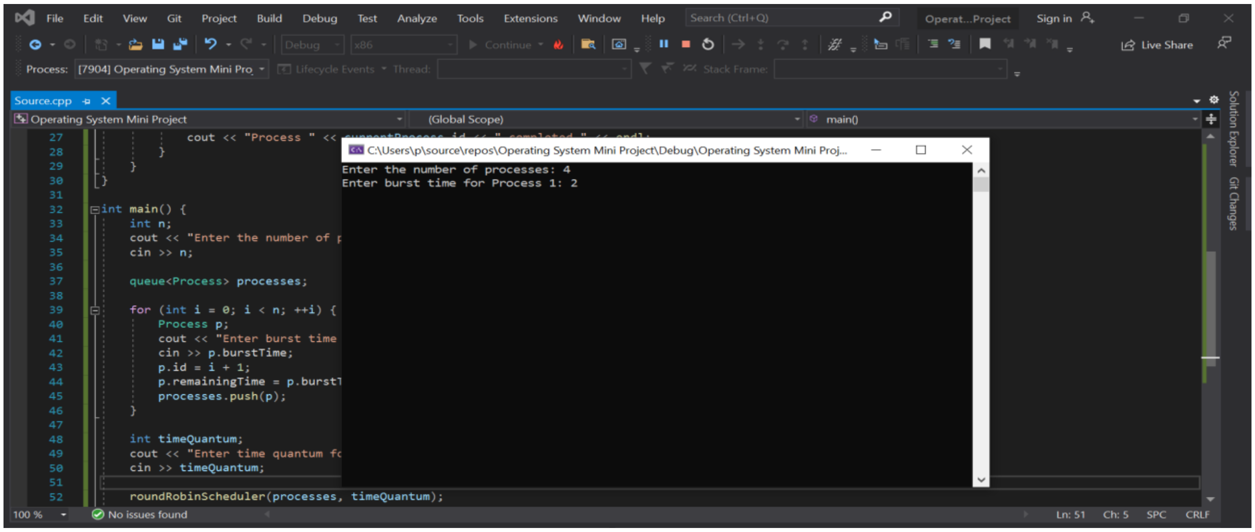
## Sample Output:



```
            cout << "Process " <<
        }
    }
}

int main() {
    int n;
    cout << "Enter the number of p
    cin >> n;

    queue<Process> processes;

    for (int i = 0; i < n; ++i) {
        Process p;
        cout << "Enter burst time
        cin >> p.burstTime;
        p.id = i + 1;
        p.remainingTime = p.burstT
        processes.push(p);
    }

    int timeQuantum;
    cout << "Enter time quantum fo
    cin >> timeQuantum;

    roundRobinScheduler(processes, timeQuantum);
```

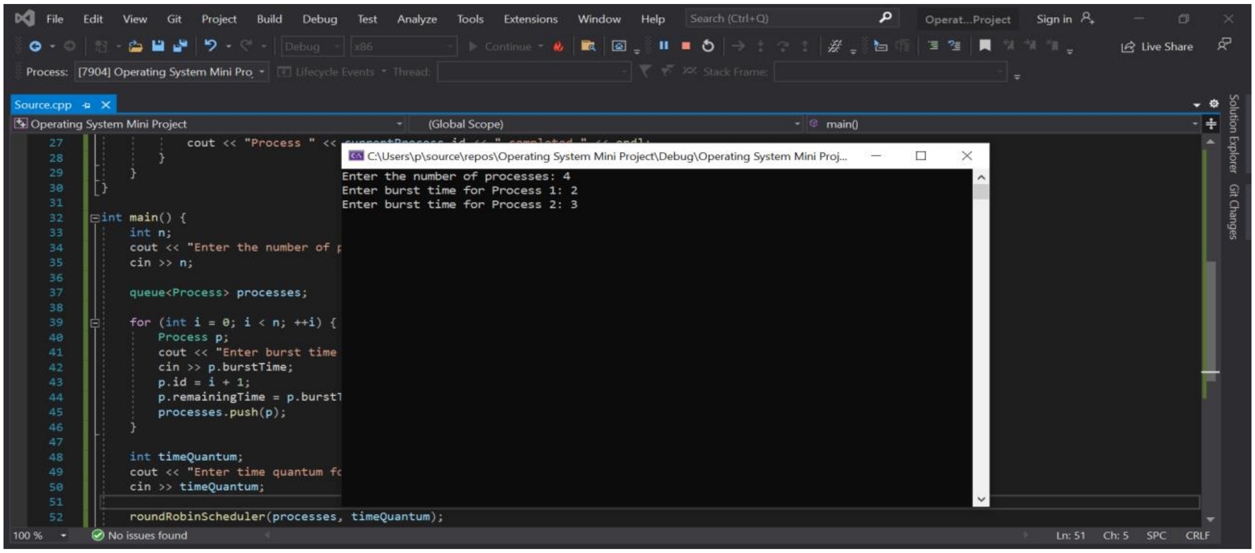
C:\Users\p\source\repos\Operating System Mini Project\Debug\Operating System Mini Proj...

```
Enter the number of processes: 4
Enter burst time for Process 1: _
```


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search (Ctrl+Q)
Operat...Project
Sign in
Live Share
Process: [7904] Operating System Mini Pro
Source.cpp
Operating System Mini Project
(Global Scope)
main()
C:\Users\p\source\repos\Operating System Mini Project\Debug\Operating System Mini Proj...
100 %
No issues found
Ln: 51 Ch: 5 SPC CRLF
Solution Explorer
Git Changes

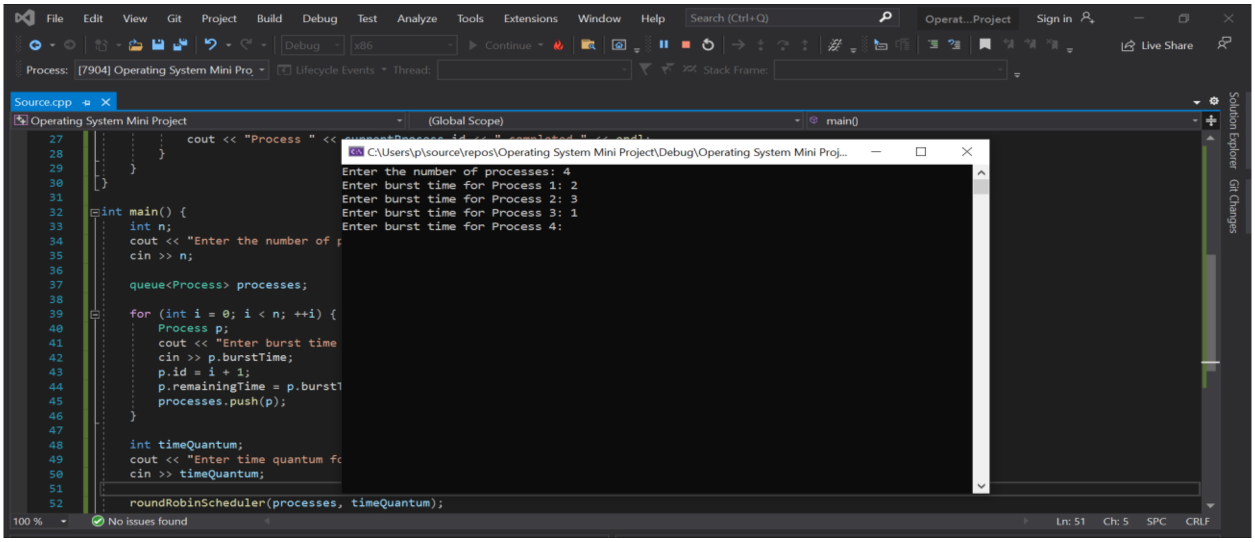
```cpp
                cout << "Process " <<
            }
        }
}

int main() {
    int n;
    cout << "Enter the number of p
    cin >> n;

    queue<Process> processes;

    for (int i = 0; i < n; ++i) {
        Process p;
        cout << "Enter burst time
        cin >> p.burstTime;
        p.id = i + 1;
        p.remainingTime = p.burstT
        processes.push(p);
    }

    int timeQuantum;
    cout << "Enter time quantum fo
    cin >> timeQuantum;

    roundRobinScheduler(processes, timeQuantum);
```

```
Enter the number of processes: 4
Enter burst time for Process 1: 2
```


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search (Ctrl+Q)
Operat...Project
Sign in
Live Share
Process: [7904] Operating System Mini Pro
Source.cpp
Operating System Mini Project
(Global Scope)
main()
C:\Users\p\source\repos\Operating System Mini Project\Debug\Operating System Mini Proj...
100 %
No issues found
Ln: 51 Ch: 5 SPC CRLF
Solution Explorer
Git Changes

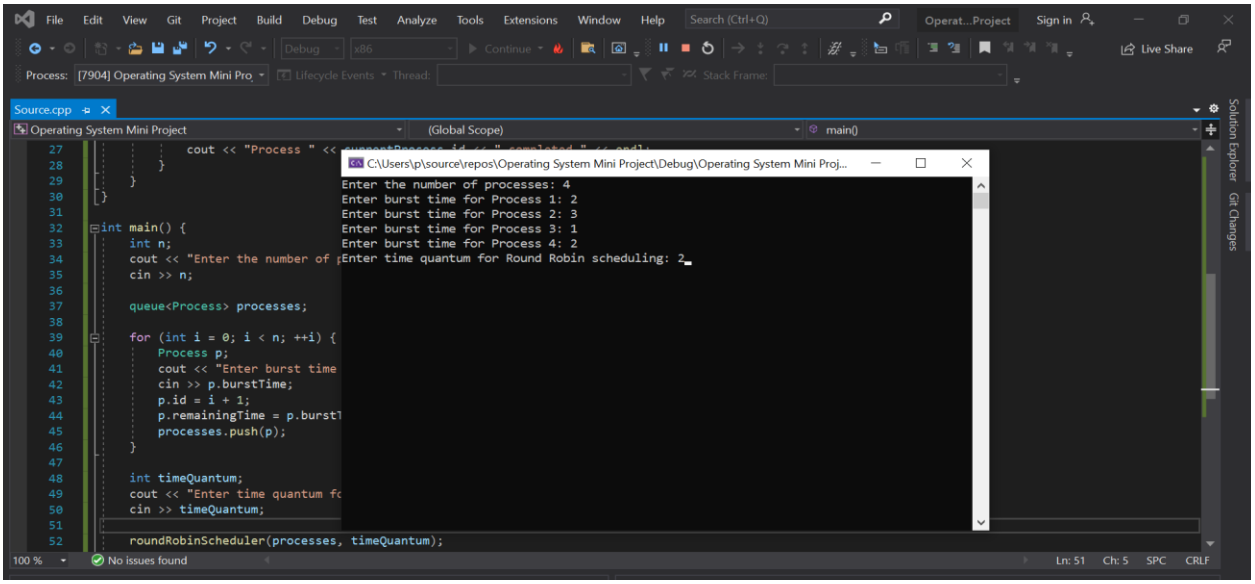
```cpp
                cout << "Process " <<
            }
        }
}

int main() {
    int n;
    cout << "Enter the number of p
    cin >> n;

    queue<Process> processes;

    for (int i = 0; i < n; ++i) {
        Process p;
        cout << "Enter burst time
        cin >> p.burstTime;
        p.id = i + 1;
        p.remainingTime = p.burstT
        processes.push(p);
    }

    int timeQuantum;
    cout << "Enter time quantum fo
    cin >> timeQuantum;

    roundRobinScheduler(processes, timeQuantum);
```

```
Enter the number of processes: 4
Enter burst time for Process 1: 2
Enter burst time for Process 2: 3
```


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search (Ctrl+Q)
Operat...Project
Sign in
Live Share
Process: [7904] Operating System Mini Pro
Source.cpp
Operating System Mini Project
(Global Scope)
main()
C:\Users\p\source\repos\Operating System Mini Project\Debug\Operating System Mini Proj...
100 %
No issues found
Ln: 51 Ch: 5 SPC CRLF
Solution Explorer
Git Changes

```cpp
                cout << "Process " <<
            }
        }
}

int main() {
    int n;
    cout << "Enter the number of p
    cin >> n;

    queue<Process> processes;

    for (int i = 0; i < n; ++i) {
        Process p;
        cout << "Enter burst time
        cin >> p.burstTime;
        p.id = i + 1;
        p.remainingTime = p.burstT
        processes.push(p);
    }

    int timeQuantum;
    cout << "Enter time quantum fo
    cin >> timeQuantum;

    roundRobinScheduler(processes, timeQuantum);
```

```
Enter the number of processes: 4
Enter burst time for Process 1: 2
Enter burst time for Process 2: 3
Enter burst time for Process 3: 1
Enter burst time for Process 4:
```



Process: [7904] Operating System Mini Pro

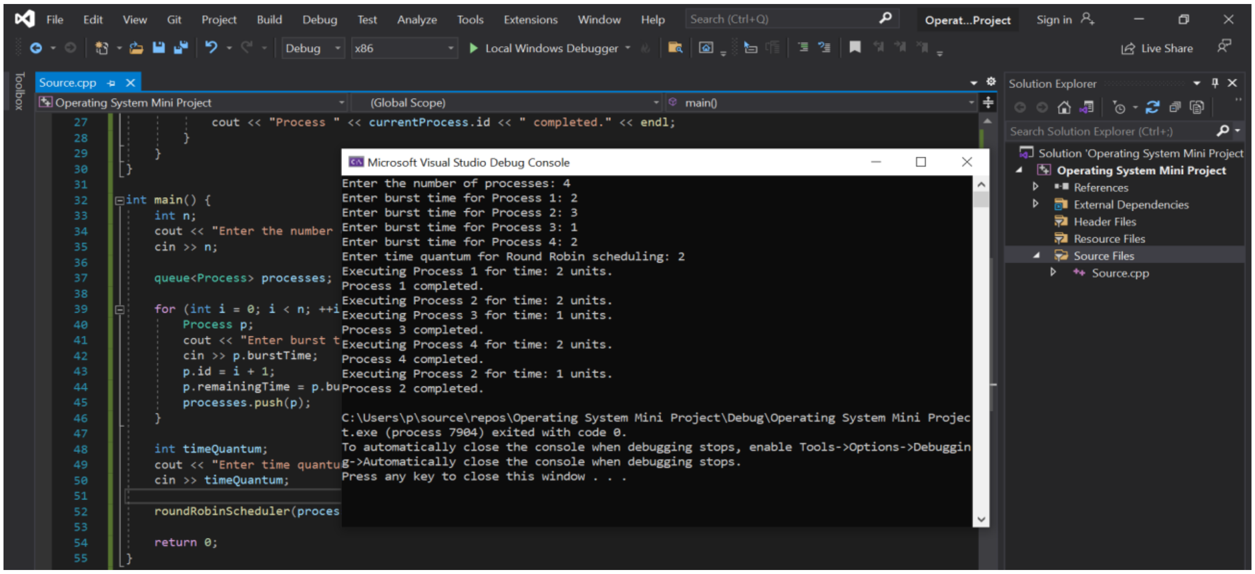
Debug | x86

Operating System Mini Project | (Global Scope) | main()

```
            cout << "Process " << currentProcess.id << " completed." << endl;
        }
    }
}

int main() {
    int n;
    cout << "Enter the number of p
    cin >> n;

    queue<Process> processes;

    for (int i = 0; i < n; ++i) {
        Process p;
        cout << "Enter burst time
        cin >> p.burstTime;
        p.id = i + 1;
        p.remainingTime = p.burstT
        processes.push(p);
    }

    int timeQuantum;
    cout << "Enter time quantum fc
    cin >> timeQuantum;

    roundRobinScheduler(processes, timeQuantum);
```

C:\Users\p\source\repos\Operating System Mini Project\Debug\Operating System Mini Proj...

```
Enter the number of processes: 4
Enter burst time for Process 1: 2
Enter burst time for Process 2: 3
Enter burst time for Process 3: 1
Enter burst time for Process 4: 2
Enter time quantum for Round Robin scheduling: 2
```



Debug | x86

Operating System Mini Project | (Global Scope) | main()

```
            cout << "Process " << currentProcess.id << " completed." << endl;
        }
    }
}

int main() {
    int n;
    cout << "Enter the number
    cin >> n;

    queue<Process> processes;

    for (int i = 0; i < n; ++i
        Process p;
        cout << "Enter burst t
        cin >> p.burstTime;
        p.id = i + 1;
        p.remainingTime = p.bu
        processes.push(p);
    }

    int timeQuantum;
    cout << "Enter time quantu
    cin >> timeQuantum;

    roundRobinScheduler(proces

    return 0;
}
```

Microsoft Visual Studio Debug Console

```
Enter the number of processes: 4
Enter burst time for Process 1: 2
Enter burst time for Process 2: 3
Enter burst time for Process 3: 1
Enter burst time for Process 4: 2
Enter time quantum for Round Robin scheduling: 2
Executing Process 1 for time: 2 units.
Process 1 completed.
Executing Process 2 for time: 2 units.
Executing Process 3 for time: 1 units.
Process 3 completed.
Executing Process 4 for time: 2 units.
Process 4 completed.
Executing Process 2 for time: 1 units.
Process 2 completed.

C:\Users\p\source\repos\Operating System Mini Project\Debug\Operating System Mini Projec
t.exe (process 7904) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debuggin
g->Automatically close the console when debugging stops.
Press any key to close this window . . .
```