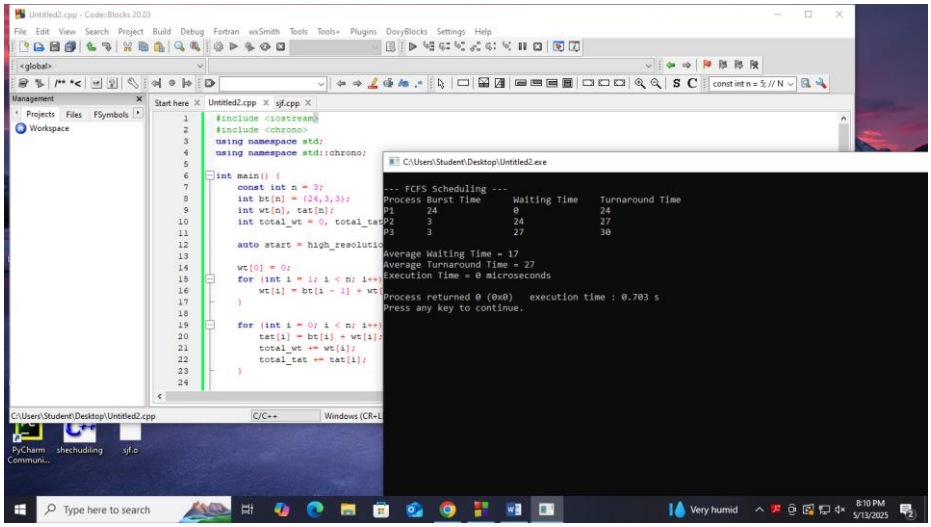
Source Code:

```
#include <iostream>
#include <chrono>
using namespace std;
using namespace std::chrono;

int main() {
    const int n = 3;
    int bt[n] = {24,3,2};
    int wt[n], tat[n];
    int total_wt = 0, total_tat = 0;
    auto start = high_resolution_clock::now();
    wt[0] = 0;
    for (int i = 1; i < n; i++) {
        wt[i] = bt[i - 1] + wt[i - 1];
    }
    for (int i = 0; i < n; i++) {
        tat[i] = bt[i] + wt[i];
        total_wt += wt[i];
        total_tat += tat[i];
    }
    auto stop = high_resolution_clock::now();
    cout << "\n--- FCFS Scheduling ---\n";
    cout << "Process\tBurst Time\tWaiting Time\tTurnaround Time\n";
    for (int i = 0; i < n; i++) {
        cout << "P" << i + 1 << "\t" << bt[i] << "\t\t" << wt[i] << "\t\t" << tat[i] << endl;
    }
```

```
    cout << "\nAverage Waiting Time = " << (float)total_wt / n;
    cout << "\nAverage Turnaround Time = " << (float)total_tat / n;
    auto duration = duration_cast<microseconds>(stop - start);
    cout << "\nExecution Time = " << duration.count() << " microseconds" << endl;
    return 0;
}
```

Screenshots of Output: for FCF



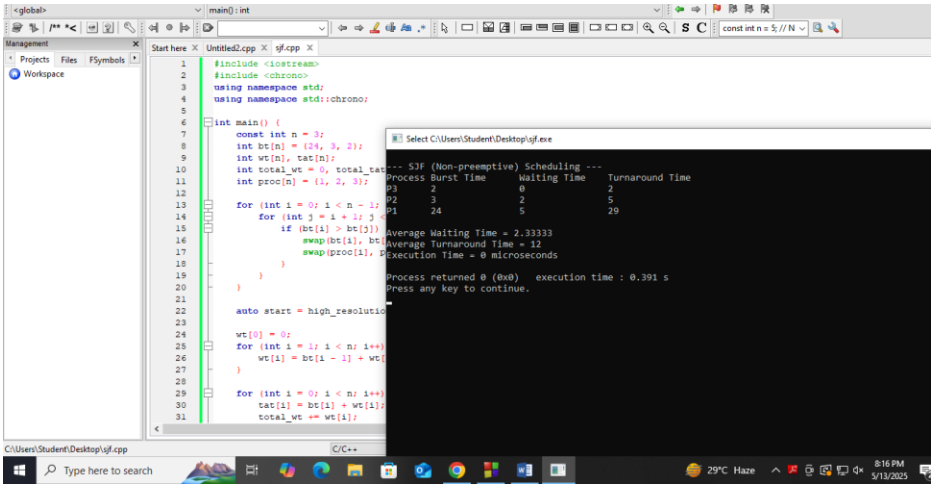
source code:for SJF

```
#include <iostream>
#include <chrono>
using namespace std;
using namespace std::chrono;

int main() {
    const int n = 3;
    int wt[n], tat[n];
    int total_wt = 0, total_tat = 0;
    int proc[n] = {1, 2, 3};
```

```
for (int i = 0; i < n - 1; i++) {
    for (int j = i + 1; j < n; j++) {
        if (bt[i] > bt[j]) {
            swap(bt[i], bt[j]);
            swap(proc[i], proc[j]);
        }
    }
}
auto start = high_resolution_clock::now();
wt[0] = 0;
for (int i = 1; i < n; i++) {
    wt[i] = bt[i - 1] + wt[i - 1];
}
for (int i = 0; i < n; i++) {
    tat[i] = bt[i] + wt[i];
    total_wt += wt[i];
    total_tat += tat[i];
}
auto stop = high_resolution_clock::now();

cout << "\n--- SJF (Non-preemptive) Scheduling ---\n";
cout << "Process\tBurst Time\tWaiting Time\tTurnaround Time\n";
for (int i = 0; i < n; i++) {
    cout << "P" << proc[i] << "\t" << bt[i] << "\t\t" << wt[i] << "\t\t" << tat[i] << endl;
}
cout << "\nAverage Waiting Time = " << (float)total_wt / n;
cout << "\nAverage Turnaround Time = " << (float)total_tat / n;
auto duration = duration_cast<microseconds>(stop - start);
cout << "\nExecution Time = " << duration.count() << " microseconds" << endl;
```

```
    return 0;
}
```

Screenshots of output:for SJF



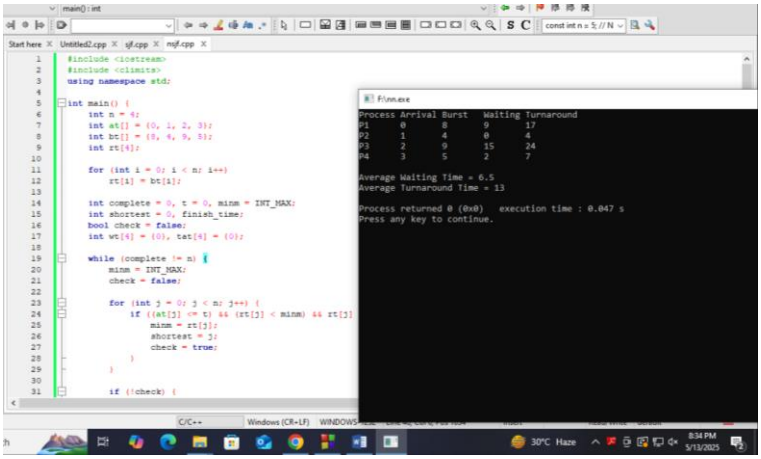
Source code : for no preemption SJF

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

int main() {
    int n = 4;
    int at[] = {0, 1, 2, 3};
    int bt[] = {8, 4, 9, 5};
    int rt[4];

    for (int i = 0; i < n; i++)
        rt[i] = bt[i];
```

```
int complete = 0, t = 0, minm = INT_MAX;
int shortest = 0, finish_time;
bool check = false;
int wt[4] = {0}, tat[4] = {0};

while (complete != n) {
   minm = INT_MAX;
   check = false;

   for (int j = 0; j < n; j++) {
      if ((at[j] <= t) && (rt[j] < minm) && rt[j] > 0) {
         minm = rt[j];
         shortest = j;
         check = true;
      }
   }

   if (!check) {
      t++;
      continue;
   }

   rt[shortest]--;

   if (rt[shortest] == 0) {
      complete++;
      finish_time = t + 1;
      wt[shortest] = finish_time - bt[shortest] - at[shortest];
      if (wt[shortest] < 0) wt[shortest] = 0;
```

```
        }

        t++;
    }

    for (int i = 0; i < n; i++) {
        tat[i] = bt[i] + wt[i];
    }

    cout << "Process\tArrival\tBurst\tWaiting\tTurnaround\n";
    int total_wt = 0, total_tat = 0;
    for (int i = 0; i < n; i++) {
        total_wt += wt[i];
        total_tat += tat[i];
        cout << "P" << i + 1 << "\t" << at[i] << "\t" << bt[i]
             << "\t" << wt[i] << "\t" << tat[i] << endl;
    }

    cout << "\nAverage Waiting Time = " << (float)total_wt / n;
    cout << "\nAverage Turnaround Time = " << (float)total_tat / n << endl;

    return 0;
}
```

Screenshots of Output: for no preemption of SJF


#include <iostream>
#include <climits>
using namespace std;
int main() {
int n = 4;
int at[] = {0, 1, 2, 3};
int bt[] = {8, 4, 9, 5};
int rt[4];
for (int i = 0; i < n; i++)
rt[i] = bt[i];
int complete = 0, t = 0, minm = INT_MAX;
int shortest = 0, finish_time;
bool check = false;
int wt[4] = {0}, tat[4] = {0};
while (complete != n) {
minm = INT_MAX;
check = false;
for (int j = 0; j < n; j++) {
if ((at[j] <= t) && (rt[j] < minm) && rt[j]
minm = rt[j];
shortest = j;
check = true;
}
}
if (!check) {
F:\nn.exe
Process Arrival Burst Waiting Turnaround
P1 0 8 9 17
P2 1 4 0 4
P3 2 9 15 24
P4 3 5 2 7
Average Waiting Time = 6.5
Average Turnaround Time = 13
Process returned 0 (0x0) execution time : 0.047 s
Press any key to continue.