

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);

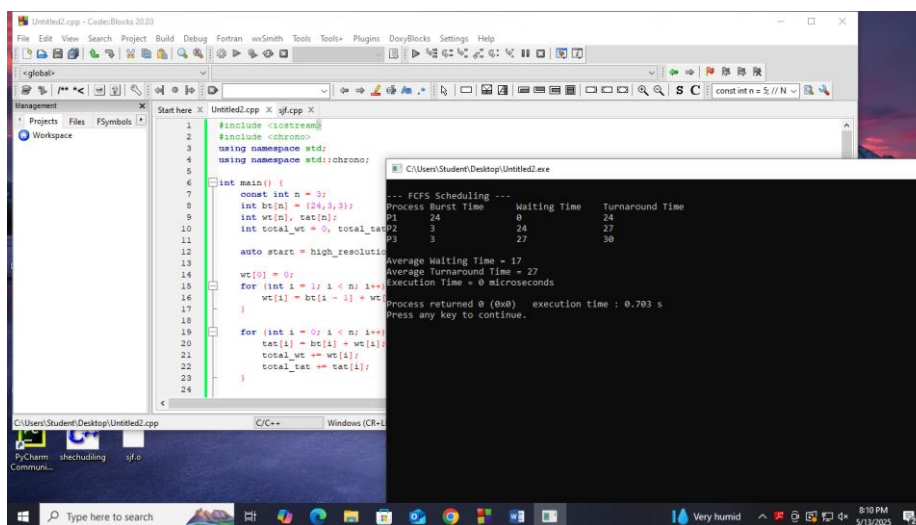
cout << "\nExecution Time = " << duration.count() << " microseconds" << endl;

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;

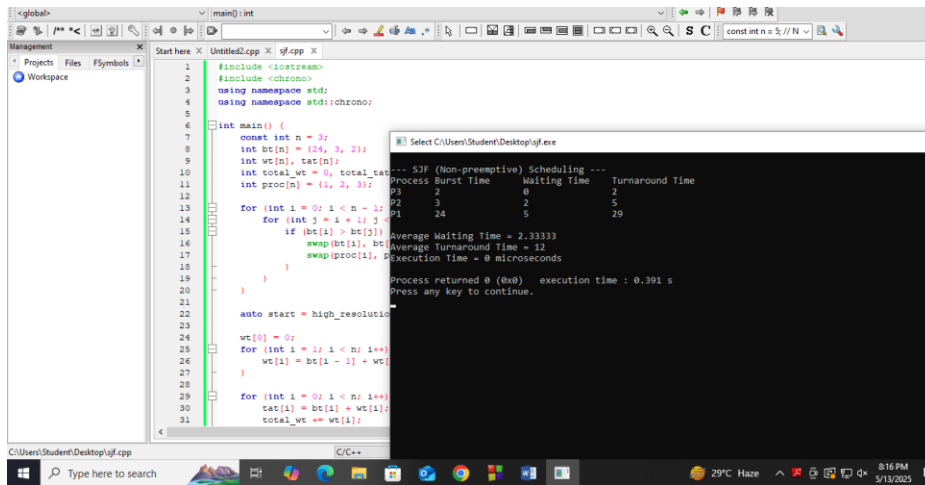
```

```

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

```
man01.cpp
Start here X: Untitled2.cpp X: qf.cpp X: mof.cpp X
1 #include <iostream>
2 #include <algorithm>
3 using namespace std;
4
5 int main() {
6     int n = 4;
7     int at[] = {0, 1, 2, 3};
8     int bt[] = {8, 4, 9, 5};
9     int rt[];
10
11     for (int i = 0; i < n; i++)
12         rt[i] = bt[i];
13
14     int complete = 0, t = 0, minn = INT_MAX;
15     int shortest = 0, finish_time;
16     bool check = false;
17     int wt[4] = {0}, tat[4] = {0};
18
19     while (complete != n) {
20         minn = INT_MAX;
21         check = false;
22
23         for (int j = 0; j < n; j++) {
24             if ((at[j] <= t) && (rt[j] < minn) && rt[j])
25                 minn = rt[j];
26             shortest = j;
27             check = true;
28         }
29
30         if (!check)
31             continue;
32
33         t = t + minn;
34         wt[shortest] = t - at[shortest];
35         tat[shortest] = t;
36         rt[shortest] = 0;
37         complete++;
38     }
39
40     cout << "Average Waiting Time = " << (double)wt[0] + wt[1] + wt[2] + wt[3] / 4 << endl;
41     cout << "Average Turnaround Time = " << (double)tat[0] + tat[1] + tat[2] + tat[3] / 4 << endl;
42     cout << "Process returned 0 (0x0)   execution time : 0.047 s\n";
43     cout << "Press any key to continue.\n";
44     getch();
45 }
```

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.