122CS0067

Assignment-VII

Name: **Amiya Chowdhury** Roll: **122CS0067** Date: **28/10/2024**

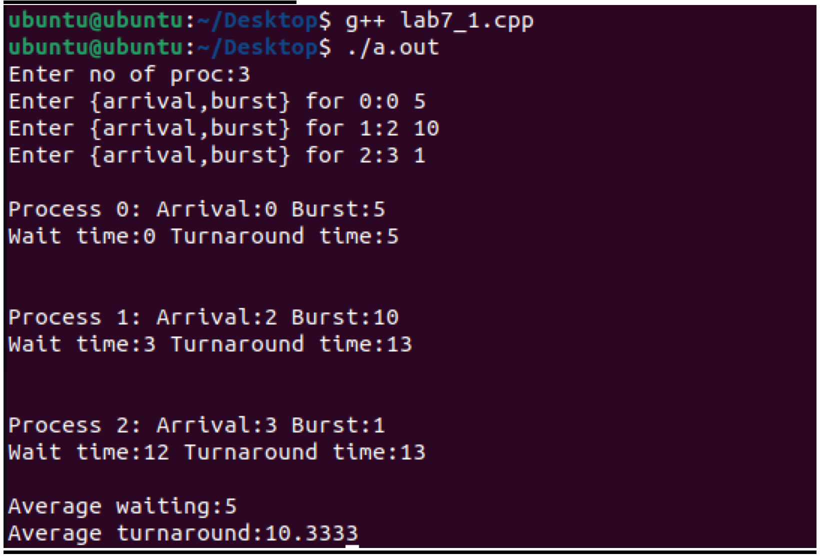
1. FCFS SCHEDULING

lab7_1.cpp

```
#include <bits/stdc++.h>
using namespace std;


void fcfs_sim(queue<pair<int,int>> proc){
    int global_time=0;
    int wait_g=0;
    int turn_g=0;
    int num=proc.size();
    for(int i=0;i<num;i++){
        if(i==0)global_time=proc.front().first;
        int arr=proc.front().first;
        int burst=proc.front().second;
        proc.pop();
        global_time+=burst;
        cout<<endl;
        cout<<"Process "<<i<<": Arrival:"<<arr<<" Burst:"<<burst<<endl;
        int wait=global_time-arr-burst;
        int turn=wait+burst;
        wait_g+=wait;
        turn_g+=turn;
        cout<<"Wait time:"<<wait<<" Turnaround time:"<<turn<<endl;
        cout<<endl;

    }
    cout<<"Average waiting:"<<(float)wait_g/num<<endl;
    cout<<"Average turnaround:"<<(float)turn_g/num<<endl;

}

int main(){
    queue<pair<int,int>> proc;
    int n;
    cout<<"Enter no of proc:";
    cin>>n;
    for(int i=0;i<n;i++){
        pair<int,int> temp;
        cout<<"Enter {arrival,burst} for "<<i<<":";
        cin>>temp.first;
        cin>>temp.second;
        proc.push(temp);
    }
    fcfs_sim(proc);
    return 0;
}
```

122CS0067

Terminal Screenshot:



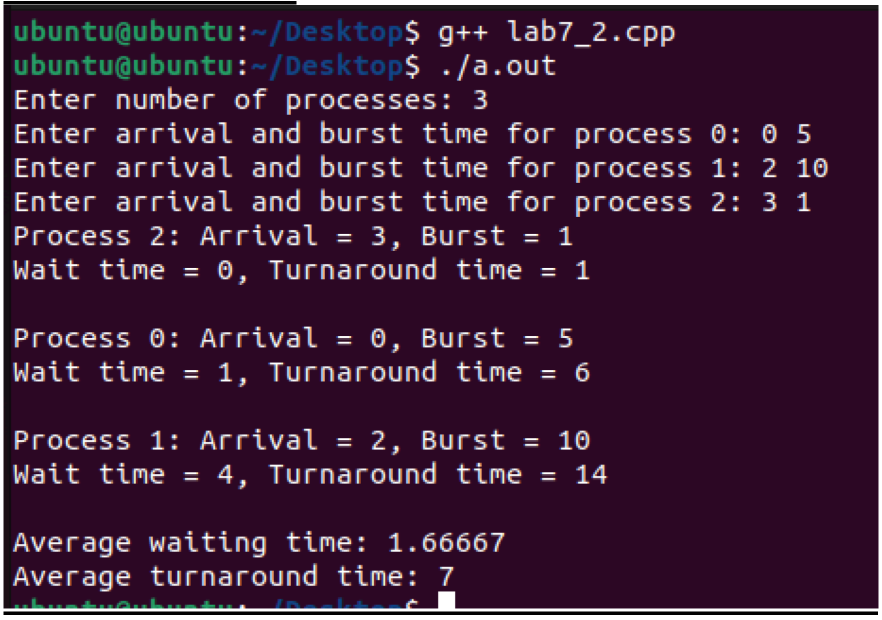
```
ubuntu@ubuntu:~/Desktop$ g++ lab7_1.cpp
ubuntu@ubuntu:~/Desktop$ ./a.out
Enter no of proc:3
Enter {arrival,burst} for 0:0 5
Enter {arrival,burst} for 1:2 10
Enter {arrival,burst} for 2:3 1

Process 0: Arrival:0 Burst:5
Wait time:0 Turnaround time:5


Process 1: Arrival:2 Burst:10
Wait time:3 Turnaround time:13


Process 2: Arrival:3 Burst:1
Wait time:12 Turnaround time:13

Average waiting:5
Average turnaround:10.3333
```

2. SRTF SCHEDULING

lab7_2.cpp

```cpp
#include <bits/stdc++.h>
using namespace std;

void srtf_sim(queue<vector<int>> &proc) {
    int global_time = 0;
    int wait_g = 0;
    int turn_g = 0;
    int num = proc.size();

    vector<int> remaining_burst(num);
    vector<int> wait(num, 0);
    vector<int> turn(num, 0);
    vector<int> arrival(num);
    vector<int> burst(num);


    for (int i = 0; i < num; i++) {
        vector<int> cur = proc.front();
        proc.pop();
        arrival[i] = cur[1];
        burst[i] = cur[2];
        remaining_burst[i] = cur[2];
        proc.push(cur);
    }

    int completed = 0;
    vector<bool> is_completed(num, false);


    while (completed < num) {
        int shortest = -1;
        int min_burst = INT_MAX;
        for (int i = 0; i < num; i++) {
            if (arrival[i] <= global_time && !is_completed[i] && remaining_burst[i] <
min_burst) {
                min_burst = remaining_burst[i];
                shortest = i;
            }
        }

        if (shortest == -1) {

            global_time++;
            continue;
        }


        remaining_burst[shortest]--;
        global_time++;


        if (remaining_burst[shortest] == 0) {
            is_completed[shortest] = true;
            completed++;
            int finish_time = global_time;


            turn[shortest] = finish_time - arrival[shortest];
            wait[shortest] = turn[shortest] - burst[shortest];

            wait_g += wait[shortest];
            turn_g += turn[shortest];


            cout << "Process " << shortest << ": Arrival = " << arrival[shortest]
                 << ", Burst = " << burst[shortest] << endl;
            cout << "Wait time = " << wait[shortest]
                 << ", Turnaround time = " << turn[shortest] << endl << endl;
        }
    }


    cout << "Average waiting time: " << (float)wait_g / num << endl;
    cout << "Average turnaround time: " << (float)turn_g / num << endl;
}

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

    for (int i = 0; i < n; i++) {
        vector<int> temp(3);
        cout << "Enter arrival and burst time for process " << i << ": ";
        cin >> temp[1];
        cin >> temp[2];
        temp[0] = 0;
        proc.push(temp);
    }

    srtf_sim(proc);
    return 0;
}
```

Terminal Screenshot:



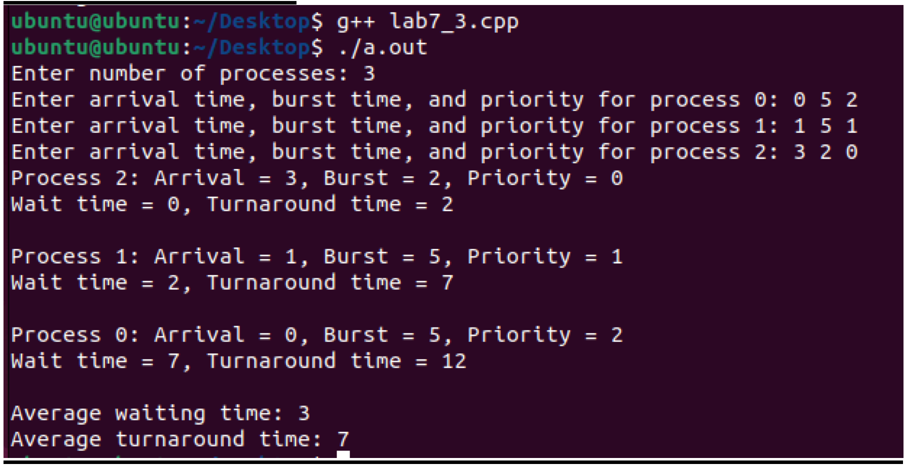
```
ubuntu@ubuntu:~/Desktop$ g++ lab7_2.cpp
ubuntu@ubuntu:~/Desktop$ ./a.out
Enter number of processes: 3
Enter arrival and burst time for process 0: 0 5
Enter arrival and burst time for process 1: 2 10
Enter arrival and burst time for process 2: 3 1
Process 2: Arrival = 3, Burst = 1
Wait time = 0, Turnaround time = 1

Process 0: Arrival = 0, Burst = 5
Wait time = 1, Turnaround time = 6

Process 1: Arrival = 2, Burst = 10
Wait time = 4, Turnaround time = 14

Average waiting time: 1.66667
Average turnaround time: 7
```

3. PRIORITY SCHEDULING

lab7_3.cpp

```
#include <bits/stdc++.h>
using namespace std;

void priority_scheduling(queue<vector<int>> &proc) {
    int global_time = 0;
    int wait_g = 0;
    int turn_g = 0;
    int num = proc.size();

    vector<int> remaining_burst(num);
    vector<int> wait(num, 0);
    vector<int> turn(num, 0);
    vector<int> arrival(num);
    vector<int> burst(num);
    vector<int> priority(num);
    for (int i = 0; i < num; i++) {
        vector<int> cur = proc.front();
        proc.pop();
        arrival[i] = cur[1];
        burst[i] = cur[2];
        priority[i] = cur[3];
        remaining_burst[i] = cur[2];
        proc.push(cur);
    }

    int completed = 0;
    vector<bool> is_completed(num, false);

    while (completed < num) {
        int highest_priority = -1;
        int min_priority = INT_MAX;


        for (int i = 0; i < num; i++) {
            if (arrival[i] <= global_time && !is_completed[i] && priority[i] <
min_priority) {
                min_priority = priority[i];
                highest_priority = i;
            }
        }

        if (highest_priority == -1) {
            global_time++;
            continue;
        }

        remaining_burst[highest_priority]--;
        global_time++;

        if (remaining_burst[highest_priority] == 0) {
            is_completed[highest_priority] = true;
            completed++;
            int finish_time = global_time;

            turn[highest_priority] = finish_time - arrival[highest_priority];
            wait[highest_priority] = turn[highest_priority] - burst[highest_priority];

            wait_g += wait[highest_priority];
            turn_g += turn[highest_priority];

            cout << "Process " << highest_priority << ": Arrival = " <<
arrival[highest_priority]
                 << ", Burst = " << burst[highest_priority] << ", Priority = " <<
priority[highest_priority] << endl;
            cout << "Wait time = " << wait[highest_priority]
                 << ", Turnaround time = " << turn[highest_priority] << endl << endl;
        }
    }
    cout << "Average waiting time: " << (float)wait_g / num << endl;
    cout << "Average turnaround time: " << (float)turn_g / num << endl;
}

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

    for (int i = 0; i < n; i++) {
        vector<int> temp(4);
        cout << "Enter arrival time, burst time, and priority for process " << i << ":
";
        cin >> temp[1];
        cin >> temp[2];
        cin >> temp[3];
        temp[0] = 0;
        proc.push(temp);
    }

    priority_scheduling(proc);
    return 0;
}
```

Terminal Screenshot:



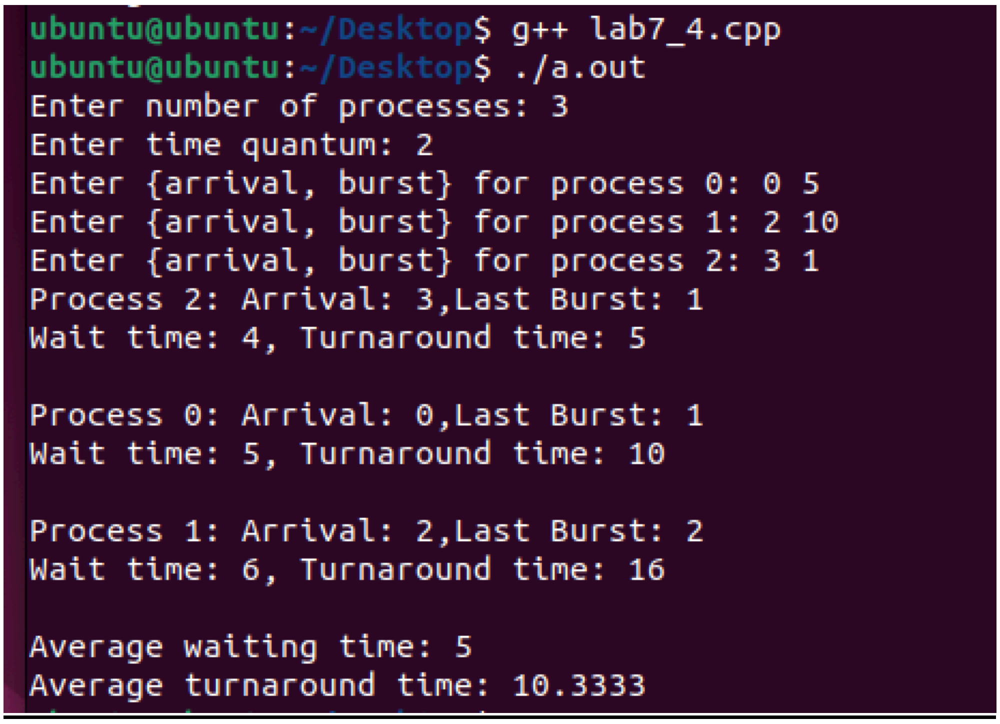
```
ubuntu@ubuntu:~/Desktop$ g++ lab7_3.cpp
ubuntu@ubuntu:~/Desktop$ ./a.out
Enter number of processes: 3
Enter arrival time, burst time, and priority for process 0: 0 5 2
Enter arrival time, burst time, and priority for process 1: 1 5 1
Enter arrival time, burst time, and priority for process 2: 3 2 0
Process 2: Arrival = 3, Burst = 2, Priority = 0
Wait time = 0, Turnaround time = 2

Process 1: Arrival = 1, Burst = 5, Priority = 1
Wait time = 2, Turnaround time = 7

Process 0: Arrival = 0, Burst = 5, Priority = 2
Wait time = 7, Turnaround time = 12

Average waiting time: 3
Average turnaround time: 7
```

4. ROUND ROBIN SCHEDULING

lab7_4.cpp

```
#include <bits/stdc++.h>
using namespace std;

void rr_sim(queue<vector<int>> &proc, int q) {
    int global_time = 0;
    int wait_g = 0;
    int turn_g = 0;
    int num = proc.size();
    vector<int> remaining_burst(num);
    vector<int> wait(num, 0);
    vector<int> turn(num, 0);
    vector<int> arrival(num);


    for (int i = 0; i < num; i++) {
        vector<int> cur = proc.front(); proc.pop();
        remaining_burst[i] = cur[2];
        arrival[i] = cur[1];
        proc.push(cur);
    }


    while (true) {
        bool all_done = true;

        for (int i = 0; i < num; i++) {
            if (remaining_burst[i] > 0) {
                all_done = false;
                if (remaining_burst[i] > q) {
                    global_time += q;
                    remaining_burst[i] -= q;
                    for(int j=0;j<num;j++){
                        if(j!=i && remaining_burst[j]!=0)wait[j]+=q;
                        if(remaining_burst[j]!=0)turn[j]+=q;
                    }
                } else {

                    global_time += remaining_burst[i];
                    int burst = remaining_burst[i];

                    for(int j=0;j<num;j++){
                        if(j!=i && remaining_burst[j]!=0)wait[j]+=burst;
                        if(remaining_burst[j]!=0)turn[j]+=burst;
                    }

                    remaining_burst[i] = 0;


                    int wait_time = wait[i];
                    int turn_time = turn[i];

                    wait_g += wait_time;
                    turn_g += turn_time;


                    cout << "Process " << i << ": Arrival: " << arrival[i]
                         << ",Last Burst: " << burst << endl;
                    cout << "Wait time: " << wait_time
                         << ", Turnaround time: " << turn_time << endl << endl;
                }
            }
        }

        if (all_done) break;
    }


    cout << "Average waiting time: " << (float)wait_g / num << endl;
    cout << "Average turnaround time: " << (float)turn_g / num << endl;
}

int main() {
    queue<vector<int>> proc;
    int n, q;
    cout << "Enter number of processes: ";
    cin >> n;
    cout << "Enter time quantum: ";
    cin >> q;

    for (int i = 0; i < n; i++) {
        vector<int> temp(3);
        cout << "Enter {arrival, burst} for process " << i << ": ";
        cin >> temp[1];
        cin >> temp[2];
        temp[0] = 0;
        proc.push(temp);
    }

    rr_sim(proc, q);
    return 0;
}
```

Terminal Screenshot:



```
ubuntu@ubuntu:~/Desktop$ g++ lab7_4.cpp
ubuntu@ubuntu:~/Desktop$ ./a.out
Enter number of processes: 3
Enter time quantum: 2
Enter {arrival, burst} for process 0: 0 5
Enter {arrival, burst} for process 1: 2 10
Enter {arrival, burst} for process 2: 3 1
Process 2: Arrival: 3,Last Burst: 1
Wait time: 4, Turnaround time: 5

Process 0: Arrival: 0,Last Burst: 1
Wait time: 5, Turnaround time: 10

Process 1: Arrival: 2,Last Burst: 2
Wait time: 6, Turnaround time: 16

Average waiting time: 5
Average turnaround time: 10.3333
```