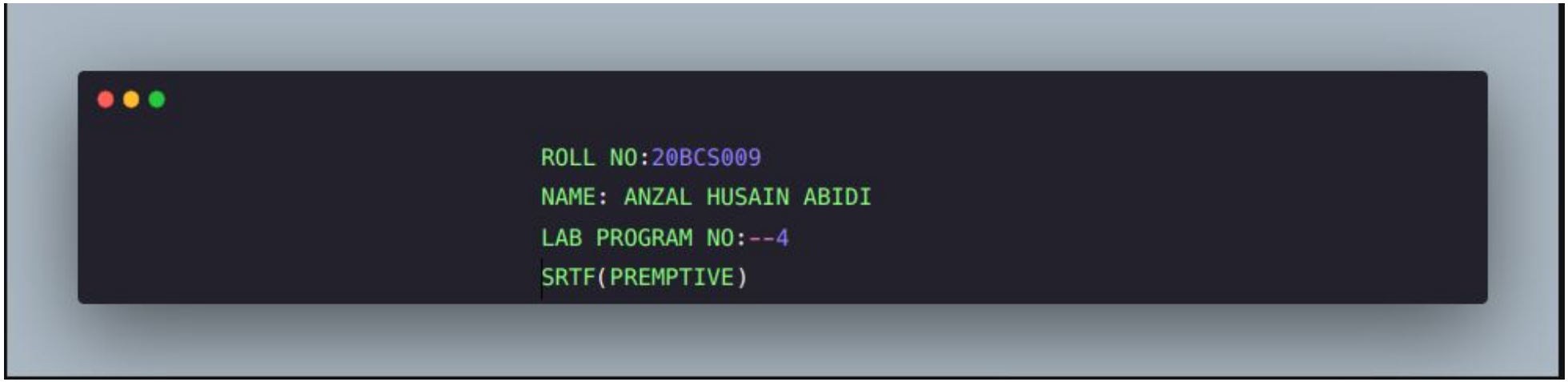
ROLL NO:20BCS009
NAME: ANZAL HUSAIN ABIDI
LAB PROGRAM NO:--4
SRTF(PREMPTIVE)

```cpp
#include <iostream>
#include <algorithm>
#include <iomanip>
#include <string.h>
#include<bits/stdc++.h>
using namespace std;
vector<int>com;
vector<int>ind;
vector<int>cotime;
vector<int>freq;
int bufftime=9999;

int f=0,sf=0,et=0;
struct process {
    int pid;
    int arrival_time;
    int burst_time;
    int start_time;
    int completion_time;
    int turnaround_time;
    int waiting_time;
    int response_time;
};
```

```
void print_gantt_chart()
{   int y=ind.size();
    cout<<"\n";
    int i, j;
    if(bufftime==9999)
    {
        bufftime=0;
    }
    if(bufftime!=0){
    //-----------------------------------------------
    printf(" ");
    // for(i=0; i<n; i++) {
        for(j=0; j<bufftime; j++) printf("--");
        printf(" ");
    // }
    // printf("\n|");
    //-----------------------------------------------
    }

    if(bufftime==0){
    printf(" ");
    }
    for(i=0; i<y; i++) {
        for(j=0; j<freq[i]; j++) printf("--");
        printf(" ");
    }
    printf("\n|");

    if(bufftime!=0){
    //-----------------------------------------------
    // for(i=0; i<n; i++) {
        for(j=0; j<bufftime - 1; j++) printf(" ");
        printf("I%d", bufftime);
        for(j=0; j<bufftime - 1; j++) printf(" ");
        printf("|");
    // }
    // printf("\n ");
    //-----------------------------------------------
    }
```

```
    for(i=0; i<y; i++) {
        for(j=0; j<freq[i] - 1; j++) printf(" ");
        printf("P%d", ind[i]);
        for(j=0; j<freq[i] - 1; j++) printf(" ");
        printf("|");
    }
    printf("\n ");

    if(bufftime!=0){
    //-----------------------------------------------
    // for(i=0; i<n; i++) {
        for(j=0; j<bufftime; j++) printf("--");
        printf(" ");
    // }
    // printf("\n");
    //-----------------------------------------------
    }

    for(i=0; i<y; i++) {
        for(j=0; j<freq[i]; j++) printf("--");
        printf(" ");
    }
    printf("\n");

    if(bufftime!=0){
    //-----------------------------------------------
    printf("0");
    // for(i=0; i<n; i++) {
        for(j=0; j<bufftime; j++) printf("  ");
        // if(bufftime > 9) printf("\b");
        // printf("%d", bufftime);

    // }
    // printf("\n");
    //-----------------------------------------------
    }

    printf("%d",bufftime);
    for(i=0; i<y; i++) {
        for(j=0; j<freq[i]; j++) printf("  ");
        if(cotime[i] > 9) printf("\b");
        printf("%d", cotime[i]);

    }
    printf("\n");

}
```

```
int main() {

    int x;
    struct process p[100];
    float avg_turnaround_time;
    float avg_waiting_time;
    float avg_response_time;
    float cpu_utilization;
    int total_turnaround_time = 0;
    int total_waiting_time = 0;
    int total_response_time = 0;
    int total_idle_time = 0;
    float throughput;
    int burst_remaining[100];
    int is_completed[100];
    memset(is_completed,0,sizeof(is_completed));

    cout << setprecision(2) << fixed;

    cout<<"Enter the number of processes: ";
    cin>>x;

        cout<<"Enter id of the process :\n";
    for(int i = 0; i < x; i++) {
        cin>>p[i].pid;
    }
        cout<<"Enter arrival time of the process :\n";
      for(int i = 0; i < x; i++) {
        cin>>p[i].arrival_time;
        bufftime=min(bufftime,p[i].arrival_time);
        }
        cout<<"Enter burst time of the process :\n";
      for(int i = 0; i < x; i++) {
        cin>>p[i].burst_time;
        burst_remaining[i] = p[i].burst_time;
    }

    int current_time = 0;
    int completed = 0;
    int prev = 0;
```

```
while(completed != x) {
    int idx = -1;
    int mn = 10000000;
    for(int i = 0; i < x; i++) {
        if(p[i].arrival_time <= current_time && is_completed[i] == 0) {
            if(burst_remaining[i] < mn) {
                mn = burst_remaining[i];
                idx = i;
            }
            if(burst_remaining[i] == mn) {
                if(p[i].arrival_time < p[idx].arrival_time) {
                    mn = burst_remaining[i];
                    idx = i;
                }
            }
        }
    }

    if(idx != -1) {
        if(burst_remaining[idx] == p[idx].burst_time) {
            p[idx].start_time = current_time;
            total_idle_time += p[idx].start_time - prev;
        }
        burst_remaining[idx] -= 1;
        current_time++;


        prev = current_time;
        // sf=p[idx].pid;
        if(p[idx].pid!=f)
        {
          ind.push_back(p[idx].pid);
        //    com.push_back(et);
        //    et=current_time;
          f=p[idx].pid;
        }
        com.push_back(p[idx].pid);
        // cout<<"--------";
        // cout<<"P"<<p[idx].pid<<" "<<"T"<<current_time<<" ";
        if(burst_remaining[idx] == 0) {
            p[idx].completion_time = current_time;
            p[idx].turnaround_time = p[idx].completion_time - p[idx].arrival_time;
            p[idx].waiting_time = p[idx].turnaround_time - p[idx].burst_time;
            p[idx].response_time = p[idx].start_time - p[idx].arrival_time;

            total_turnaround_time += p[idx].turnaround_time;
            total_waiting_time += p[idx].waiting_time;
            total_response_time += p[idx].response_time;
```

```
                total_turnaround_time += p[idx].turnaround_time;
                total_waiting_time += p[idx].waiting_time;
                total_response_time += p[idx].response_time;

                is_completed[idx] = 1;
                completed++;
            }
        }
        else {
            current_time++;
        }
    }

    int min_arrival_time = 10000000;
    int max_completion_time = -1;
    for(int i = 0; i < x; i++) {
        min_arrival_time = min(min_arrival_time,p[i].arrival_time);
        max_completion_time = max(max_completion_time,p[i].completion_time);
    }

    avg_turnaround_time = (float) total_turnaround_time / x;
    avg_waiting_time = (float) total_waiting_time / x;
    avg_response_time = (float) total_response_time / x;
    // cpu_utilization = ((max_completion_time - total_idle_time) / (float) max_completion_time )*100;
    // throughput = float(x) / (max_completion_time - min_arrival_time);

    cout<<endl<<endl;

    cout<<"Id\t"<<"AT\t"<<"BT\t"<<"RT\t"<<"CT\t"<<"TAT\t"<<"WT\t"<<"\n"<<endl;

    for(int i = 0; i < x; i++) {
        cout<<p[i].pid<<"\t"<<p[i].arrival_time<<"\t"<<p[i].burst_time<<"\t"<<p[i].start_time<<"\t"<<p[i].completion_time<<"\t"
        <<p[i].turnaround_time<<"\t"<<p[i].waiting_time<<"\t"<<"\n"<<endl;
    }
    cout<<"Average Turnaround Time = "<<avg_turnaround_time<<endl;
    cout<<"Average Waiting Time = "<<avg_waiting_time<<endl;
    cout<<"Average Response Time = "<<avg_response_time<<endl;
    // cout<<"CPU Utilization = "<<cpu_utilization<<"%"<<endl;
    // cout<<"Throughput = "<<throughput<<" process/unit time"<<endl;
```

```
    int ert=0;
    int j=0;
    int fre=0;
    for(int i=0;i<com.size();)
    {
            while(com[j]==com[i])
            {   j++;
                ert=ert+1;
                fre++;
            }
        i=j;
        cotime.push_back(ert);
        freq.push_back(fre);
        fre=0;

    }
    // for(int i=0;i<freq.size();i++)
    // {
    //     cout<<freq[i]<<" ";
    // }
    // cout<<'\n';
    // for(int i=0;i<cotime.size();i++)
    // {
    //     cout<<cotime[i]<<" ";
    // }
    print_gantt_chart();



}
```

```
anzal  anzal  ../4th sem/os lab/p4  cd "/home/anzal/Desktop/4th sem/os lab/p4/" && g++ strf.cpp -o strf && "/home/anzal/Desktop/4th sem/os lab/p4/"strf
Enter the number of processes: 6
Enter id of the process :
1
2
3
4
5
6
Enter arrival time of the process :
0
1
2
3
4
5
Enter burst time of the process :
8
4
2
1
3
2


Id      AT      BT      RT      CT      TAT     WT

1       0       8       0       20      20      12

2       1       4       1       10      9       5

3       2       2       2       4       2       0

4       3       1       4       5       2       1

5       4       3       10      13      9       6

6       5       2       5       7       2       0

Average Turnaround Time = 7.33
Average Waiting Time = 4.00
Average Response Time = 1.17

 -- -- ---- -- ---- ------ ------ --------------
|P1|P2| P3 |P4| P6 |  P2  |  P5  |      P1      |
 -- -- ---- -- ---- ------ ------ --------------
0  1  2    4  5    7     10     13             20
```