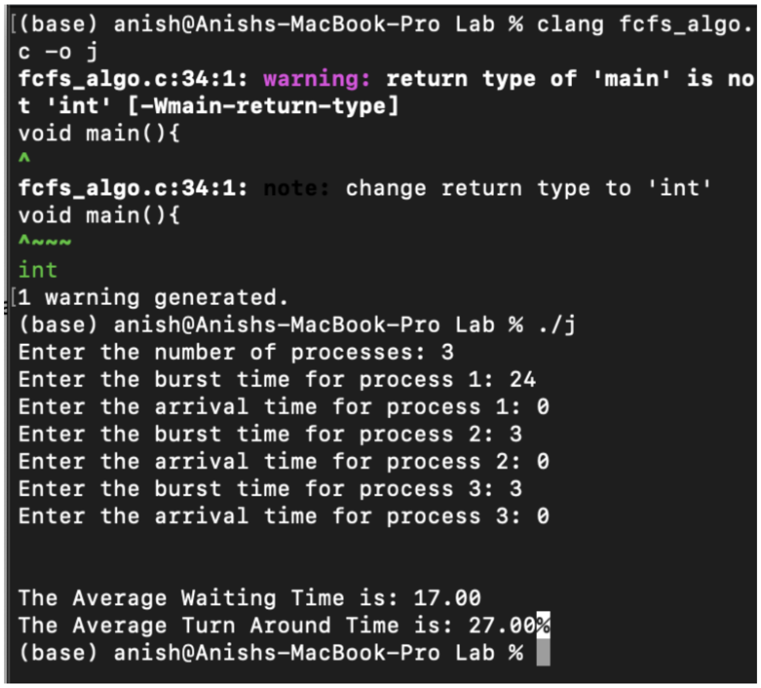
OPERATING SYSTEM LAB 3

Roll No.: K041	Name: Anish Sudhan Nair
Batch No.: A2/K2	Date: 03/01/2021

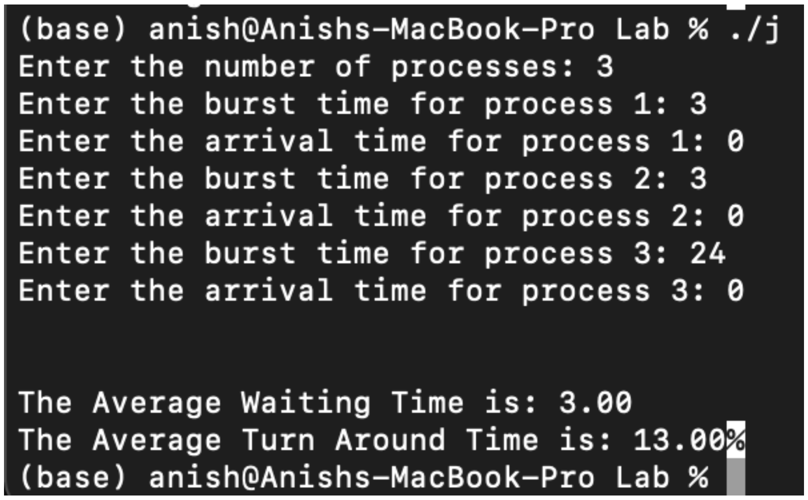
Aim: To familiarise and implement the first come first serve scheduling algorithm.

1. P1=24, P2=3, P3=3 in that order



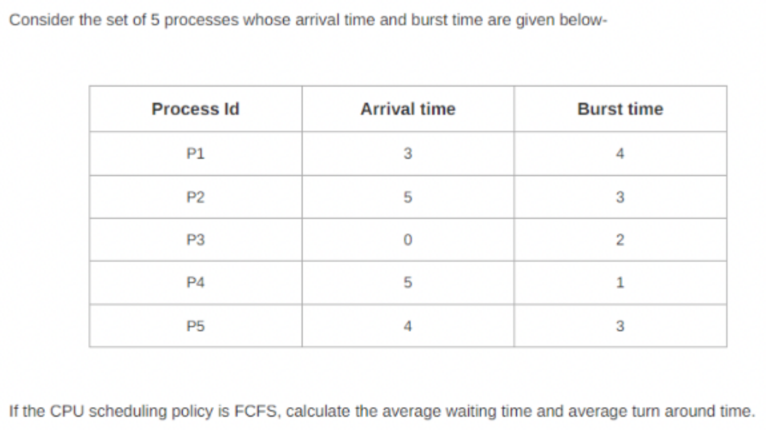
```
(base) anish@Anishs-MacBook-Pro Lab % clang fcfs_algo.
c -o j
fcfs_algo.c:34:1: warning: return type of 'main' is no
t 'int' [-Wmain-return-type]
void main(){
^
fcfs_algo.c:34:1: note: change return type to 'int'
void main(){
^~~~
int
1 warning generated.
(base) anish@Anishs-MacBook-Pro Lab % ./j
Enter the number of processes: 3
Enter the burst time for process 1: 24
Enter the arrival time for process 1: 0
Enter the burst time for process 2: 3
Enter the arrival time for process 2: 0
Enter the burst time for process 3: 3
Enter the arrival time for process 3: 0


The Average Waiting Time is: 17.00
The Average Turn Around Time is: 27.00%
(base) anish@Anishs-MacBook-Pro Lab %
```

2. Processes from part 1 in order – P2, P3, P1



```
(base) anish@Anishs-MacBook-Pro Lab % ./j
Enter the number of processes: 3
Enter the burst time for process 1: 3
Enter the arrival time for process 1: 0
Enter the burst time for process 2: 3
Enter the arrival time for process 2: 0
Enter the burst time for process 3: 24
Enter the arrival time for process 3: 0


The Average Waiting Time is: 3.00
The Average Turn Around Time is: 13.00%
(base) anish@Anishs-MacBook-Pro Lab %
```

3. Refer to this question

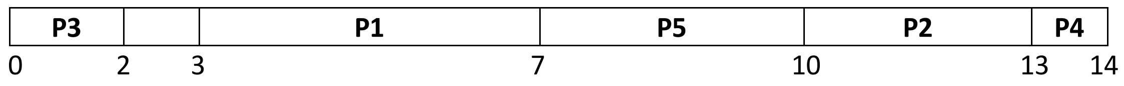


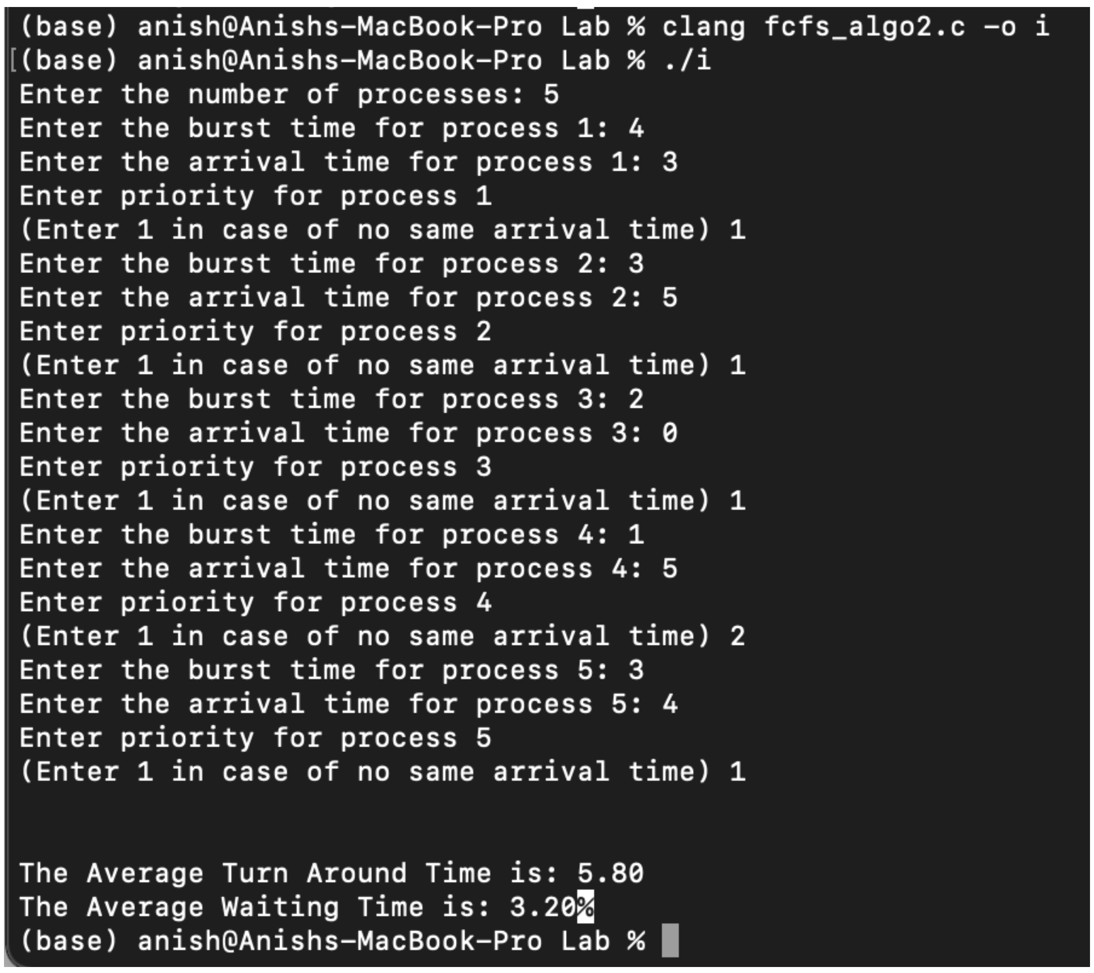
Consider the set of 5 processes whose arrival time and burst time are given below-

Process Id	Arrival time	Burst time
P1	3	4
P2	5	3
P3	0	2
P4	5	1
P5	4	3

If the CPU scheduling policy is FCFS, calculate the average waiting time and average turn around time.

Gantt chart

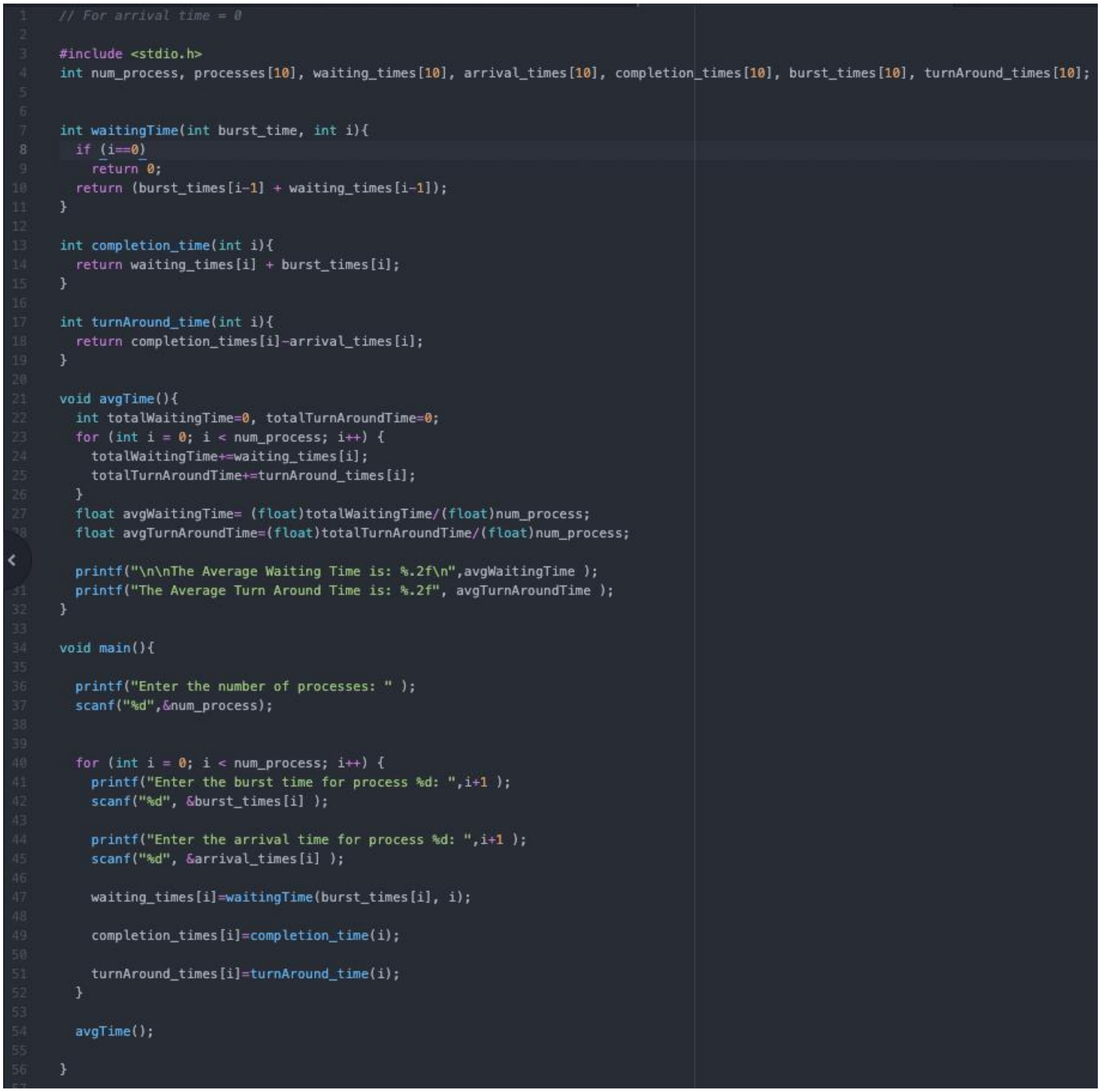




```
(base) anish@Anishs-MacBook-Pro Lab % clang fcfs_algo2.c -o i
[(base) anish@Anishs-MacBook-Pro Lab % ./i
Enter the number of processes: 5
Enter the burst time for process 1: 4
Enter the arrival time for process 1: 3
Enter priority for process 1
(Enter 1 in case of no same arrival time) 1
Enter the burst time for process 2: 3
Enter the arrival time for process 2: 5
Enter priority for process 2
(Enter 1 in case of no same arrival time) 1
Enter the burst time for process 3: 2
Enter the arrival time for process 3: 0
Enter priority for process 3
(Enter 1 in case of no same arrival time) 1
Enter the burst time for process 4: 1
Enter the arrival time for process 4: 5
Enter priority for process 4
(Enter 1 in case of no same arrival time) 2
Enter the burst time for process 5: 3
Enter the arrival time for process 5: 4
Enter priority for process 5
(Enter 1 in case of no same arrival time) 1


The Average Turn Around Time is: 5.80
The Average Waiting Time is: 3.20%
(base) anish@Anishs-MacBook-Pro Lab %
```

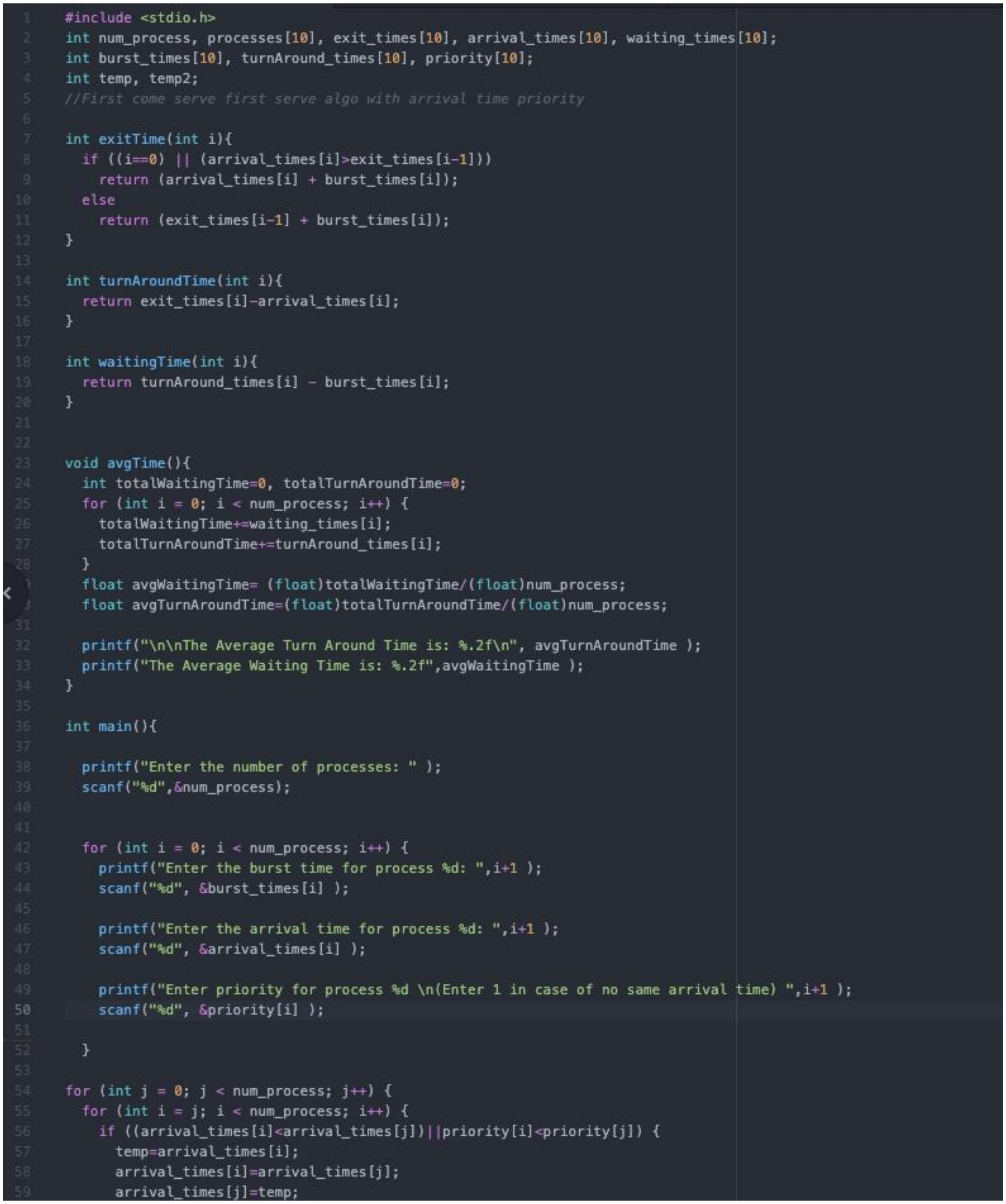
CODES:

For parts 1,2:

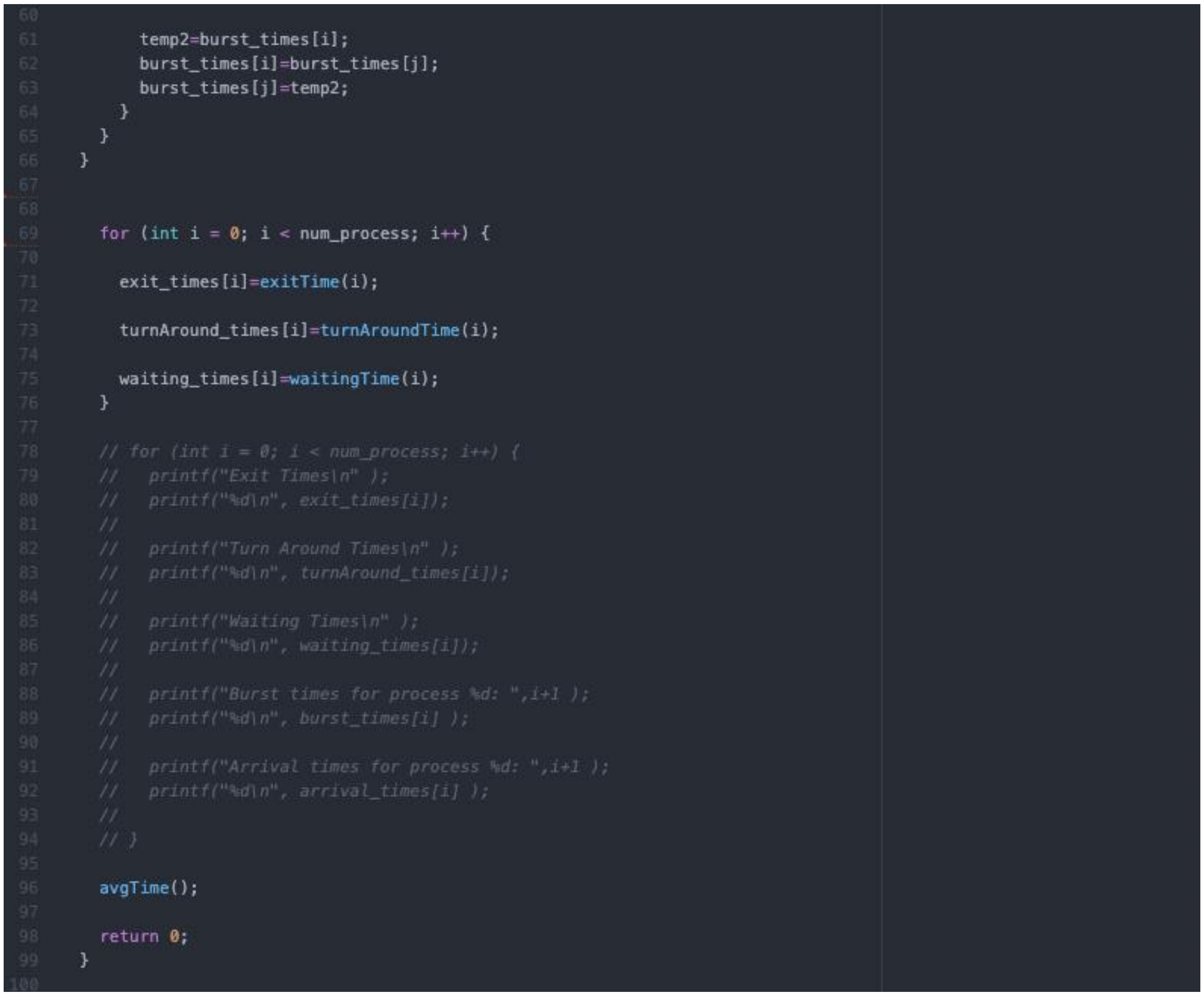


```c
// For arrival time = 0

#include <stdio.h>
int num_process, processes[10], waiting_times[10], arrival_times[10], completion_times[10], burst_times[10], turnAround_times[10];


int waitingTime(int burst_time, int i){
  if (i==0)
    return 0;
  return (burst_times[i-1] + waiting_times[i-1]);
}

int completion_time(int i){
  return waiting_times[i] + burst_times[i];
}

int turnAround_time(int i){
  return completion_times[i]-arrival_times[i];
}

void avgTime(){
  int totalWaitingTime=0, totalTurnAroundTime=0;
  for (int i = 0; i < num_process; i++) {
    totalWaitingTime+=waiting_times[i];
    totalTurnAroundTime+=turnAround_times[i];
  }
  float avgWaitingTime= (float)totalWaitingTime/(float)num_process;
  float avgTurnAroundTime=(float)totalTurnAroundTime/(float)num_process;

  printf("\n\nThe Average Waiting Time is: %.2f\n",avgWaitingTime );
  printf("The Average Turn Around Time is: %.2f", avgTurnAroundTime );
}

void main(){

  printf("Enter the number of processes: " );
  scanf("%d",&num_process);


  for (int i = 0; i < num_process; i++) {
    printf("Enter the burst time for process %d: ",i+1 );
    scanf("%d", &burst_times[i] );

    printf("Enter the arrival time for process %d: ",i+1 );
    scanf("%d", &arrival_times[i] );

    waiting_times[i]=waitingTime(burst_times[i], i);

    completion_times[i]=completion_time(i);

    turnAround_times[i]=turnAround_time(i);
  }

  avgTime();

}
```

For part 3:



```c
#include <stdio.h>
int num_process, processes[10], exit_times[10], arrival_times[10], waiting_times[10];
int burst_times[10], turnAround_times[10], priority[10];
int temp, temp2;
//First come serve first serve algo with arrival time priority

int exitTime(int i){
  if ((i==0) || (arrival_times[i]>exit_times[i-1]))
    return (arrival_times[i] + burst_times[i]);
  else
    return (exit_times[i-1] + burst_times[i]);
}

int turnAroundTime(int i){
  return exit_times[i]-arrival_times[i];
}

int waitingTime(int i){
  return turnAround_times[i] - burst_times[i];
}


void avgTime(){
  int totalWaitingTime=0, totalTurnAroundTime=0;
  for (int i = 0; i < num_process; i++) {
    totalWaitingTime+=waiting_times[i];
    totalTurnAroundTime+=turnAround_times[i];
  }
  float avgWaitingTime= (float)totalWaitingTime/(float)num_process;
  float avgTurnAroundTime=(float)totalTurnAroundTime/(float)num_process;

  printf("\n\nThe Average Turn Around Time is: %.2f\n", avgTurnAroundTime );
  printf("The Average Waiting Time is: %.2f",avgWaitingTime );
}

int main(){

  printf("Enter the number of processes: " );
  scanf("%d",&num_process);


  for (int i = 0; i < num_process; i++) {
    printf("Enter the burst time for process %d: ",i+1 );
    scanf("%d", &burst_times[i] );

    printf("Enter the arrival time for process %d: ",i+1 );
    scanf("%d", &arrival_times[i] );

    printf("Enter priority for process %d \n(Enter 1 in case of no same arrival time) ",i+1 );
    scanf("%d", &priority[i] );

  }

for (int j = 0; j < num_process; j++) {
  for (int i = j; i < num_process; i++) {
    if ((arrival_times[i]<arrival_times[j])||priority[i]<priority[j]) {
      temp=arrival_times[i];
      arrival_times[i]=arrival_times[j];
      arrival_times[j]=temp;
```



```
          temp2=burst_times[i];
          burst_times[i]=burst_times[j];
          burst_times[j]=temp2;
        }
      }
    }


    for (int i = 0; i < num_process; i++) {

      exit_times[i]=exitTime(i);

      turnAround_times[i]=turnAroundTime(i);

      waiting_times[i]=waitingTime(i);
    }

    // for (int i = 0; i < num_process; i++) {
    //   printf("Exit Times\n" );
    //   printf("%d\n", exit_times[i]);
    //
    //   printf("Turn Around Times\n" );
    //   printf("%d\n", turnAround_times[i]);
    //
    //   printf("Waiting Times\n" );
    //   printf("%d\n", waiting_times[i]);
    //
    //   printf("Burst times for process %d: ",i+1 );
    //   printf("%d\n", burst_times[i] );
    //
    //   printf("Arrival times for process %d: ",i+1 );
    //   printf("%d\n", arrival_times[i] );
    //
    // }

    avgTime();

    return 0;
  }
```

CONCLUSION:

In this lab, we were to implement and demonstrate the working of the first come first serve algorithm. By actually coding the algorithm, it helped to reinforce the working of this process and the manner in which it schedules the processes in a CPU.