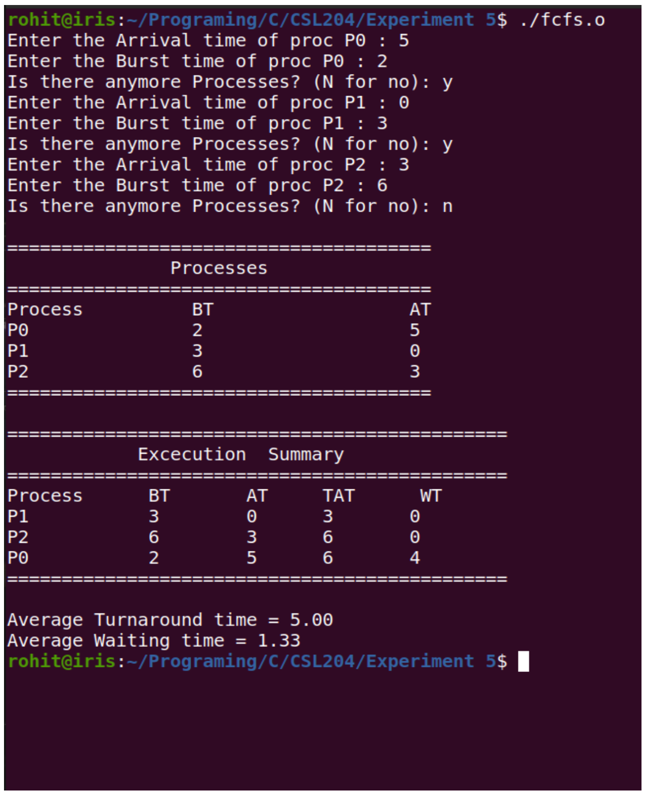
Experiment 5
Scheduling Algorithms

Done By: Rohit Karunakaran
Roll No: 58
Date Of Submission: 16-06-2021

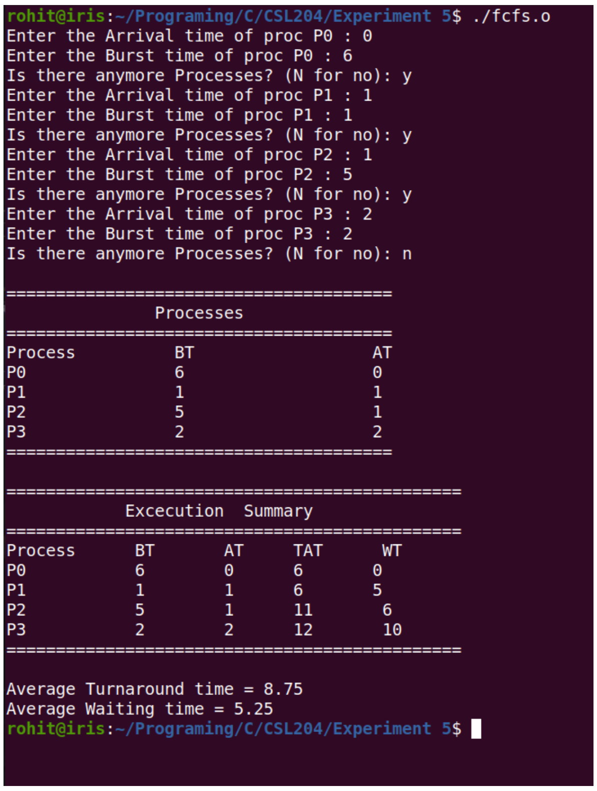
First Come First Serve Scheduling (FCFS)

```
#include<stdio.h>
#include<stdlib.h>
#define MAX_PROCESS 2

typedef struct {
    int arrival_time;
    int burst_time;
    int p_num;
    int comp;
} proc;

typedef struct{
    proc** list ;
    int size;
    int cap;
} proc_list;

typedef struct {
    int p_id;
    int tat;
    int wt;
} proc_summ;


proc_list *
create_process()
{
    proc_list* p_list = (proc_list *) malloc(sizeof(proc_list));
    p_list->size = 0;
    p_list->cap = MAX_PROCESS;

    p_list->list = (proc**)malloc(sizeof(proc)*p_list->cap);

    int RUN=1;
    int i=0;
    int c;

    while(RUN)
    {
        proc *p = (proc*)malloc(sizeof(proc));
        printf("Enter the Arrival time of proc P%d : ",i);
        scanf("%d",&p->arrival_time);
        printf("Enter the Burst time of proc P%d : ",i);
        scanf("%d",&p->burst_time);
        p->p_num = i;
        p->comp = 0;

        p_list->list[i] = p;

        i++;
        p_list->size = i;
        printf("Is there anymore Processes? (N for no): ");
        getc(stdin);
        c = getc(stdin);
        if(c == 'n' || c == 'N'){
            RUN=0;
            break;
        }

        if(! i < p_list -> cap){
            p_list -> cap = p_list->cap*2;
            p_list->list= (proc**) realloc(p_list->list, sizeof(proc)*p_list-
>cap);
        }
    }

    return p_list;
}

void
 show_process(proc_list *procs)
{
    printf("\n");
    printf("=======================================\n");
    printf("               Processes              \n");
    printf("=======================================\n");
    printf("Process         BT                 AT\n");
     for(unsigned int i=0;i<procs->size; i++){
    printf("P%d              %d                  %d\n",
            procs->list[i]->p_num,
            procs->list[i]->burst_time,
            procs->list[i]->arrival_time);
     }
    printf("=======================================\n");

}

void
 sort ( proc** list, int s) {
    for(int i=0 ; i<s; i++){
        for (int j=0 ;j<=s-i-1 ;j++){
            if(list[j]->arrival_time > list[j+1]->arrival_time){
                proc *temp = list[j];
                list[j] = list[j+1];
                list[j+1] = temp;
            }
        }
    }
}

void
show_result(proc_summ * psum, proc_list *procs)
 {
    if (psum == NULL) return;
    int n = procs->size;
    printf("\n");
    printf("==============================================\n");
    printf("            Excecution  Summary               \n");
    printf("==============================================\n");
    printf("Process      BT       AT     TAT      WT     \n");
    int wt_sum = 0;
    int tat_sum = 0;
    for(unsigned int i=0;i<procs->size; i++){
        printf("P%d           %d        %d       %d        %d      \n",
            procs->list[i]->p_num,
            procs->list[i]->burst_time,
            procs->list[i]->arrival_time,
            psum[i].tat,
            psum[i].wt);
        wt_sum += psum[i].wt;
        tat_sum += psum[i].tat;

    }
    printf("==============================================\n");

    float average_wt = (float)wt_sum / n;
    float average_tat = (float)tat_sum / n;

    printf("\nAverage Turnaround time = %.2f\nAverage Waiting time = %.2f\
n",average_tat,average_wt);
}


proc_summ *
fcfs(proc_list *procs)
 {
    if ( procs->size > 0 ){

        int n = procs->size;
        proc_summ *ps = (proc_summ*) malloc (sizeof (proc_summ)*n);

        int time = 0;

        //Sort the process using their arrival time.
        sort(procs->list, n-1);
        //show_process(procs);

        time = procs -> list[0]->arrival_time;

        for (int i = 0 ; i < n; ){
            if (time >= procs->list[i] -> arrival_time){
                time = time + procs->list[i]->burst_time;
                ps[i].p_id = procs->list[i] -> p_num;
                ps[i].tat = time - procs -> list[i] -> arrival_time;
                ps[i].wt = ps[i].tat - procs -> list[i] -> burst_time;
                i++;
            }
            else{
                time++;
            }
        }

        return ps;
    }
    return NULL;

}


void
main()
{
    proc_list *s = create_process();
    show_process(s);
    proc_summ *psum = fcfs(s);
    show_result(psum, s);
}
```

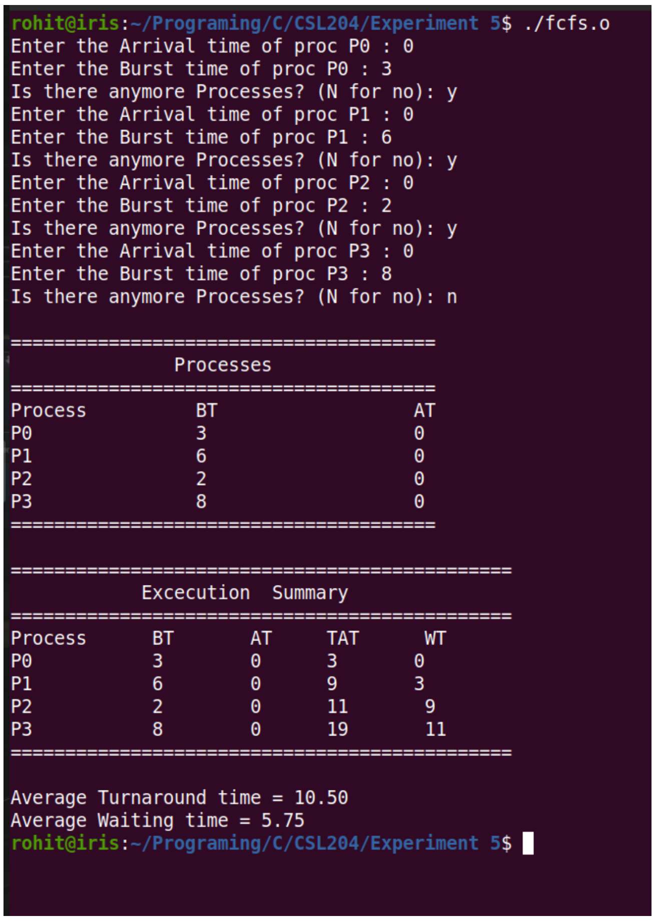
Screen Shots:



```
rohit@iris:~/Programing/C/CSL204/Experiment 5$ ./fcfs.o
Enter the Arrival time of proc P0 : 5
Enter the Burst time of proc P0 : 2
Is there anymore Processes? (N for no): y
Enter the Arrival time of proc P1 : 0
Enter the Burst time of proc P1 : 3
Is there anymore Processes? (N for no): y
Enter the Arrival time of proc P2 : 3
Enter the Burst time of proc P2 : 6
Is there anymore Processes? (N for no): n

======================================
               Processes
======================================
Process          BT                  AT
P0               2                   5
P1               3                   0
P2               6                   3
======================================

=============================================
            Excecution  Summary
=============================================
Process      BT        AT      TAT       WT
P1           3         0       3         0
P2           6         3       6         0
P0           2         5       6         4
=============================================

Average Turnaround time = 5.00
Average Waiting time = 1.33
rohit@iris:~/Programing/C/CSL204/Experiment 5$
```



```
rohit@iris:~/Programing/C/CSL204/Experiment 5$ ./fcfs.o
Enter the Arrival time of proc P0 : 0
Enter the Burst time of proc P0 : 6
Is there anymore Processes? (N for no): y
Enter the Arrival time of proc P1 : 1
Enter the Burst time of proc P1 : 1
Is there anymore Processes? (N for no): y
Enter the Arrival time of proc P2 : 1
Enter the Burst time of proc P2 : 5
Is there anymore Processes? (N for no): y
Enter the Arrival time of proc P3 : 2
Enter the Burst time of proc P3 : 2
Is there anymore Processes? (N for no): n

======================================
               Processes
======================================
Process          BT                  AT
P0               6                   0
P1               1                   1
P2               5                   1
P3               2                   2
======================================

=============================================
            Excecution  Summary
=============================================
Process      BT        AT      TAT       WT
P0           6         0       6         0
P1           1         1       6         5
P2           5         1       11         6
P3           2         2       12         10
=============================================

Average Turnaround time = 8.75
Average Waiting time = 5.25
rohit@iris:~/Programing/C/CSL204/Experiment 5$
```

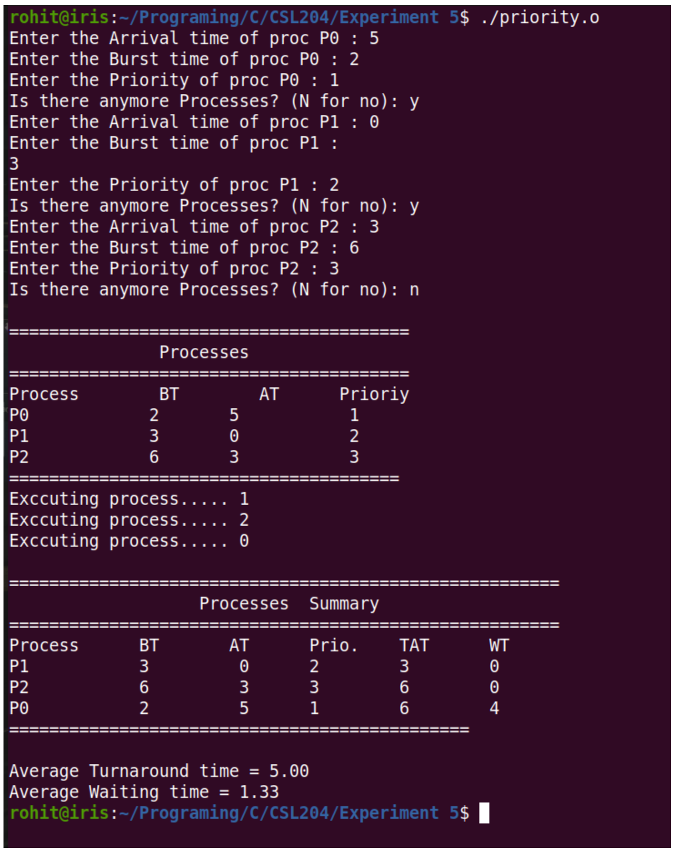


```
rohit@iris:~/Programing/C/CSL204/Experiment 5$ ./fcfs.o
Enter the Arrival time of proc P0 : 0
Enter the Burst time of proc P0 : 3
Is there anymore Processes? (N for no): y
Enter the Arrival time of proc P1 : 0
Enter the Burst time of proc P1 : 6
Is there anymore Processes? (N for no): y
Enter the Arrival time of proc P2 : 0
Enter the Burst time of proc P2 : 2
Is there anymore Processes? (N for no): y
Enter the Arrival time of proc P3 : 0
Enter the Burst time of proc P3 : 8
Is there anymore Processes? (N for no): n

======================================
               Processes
======================================
Process          BT                  AT
P0               3                   0
P1               6                   0
P2               2                   0
P3               8                   0
======================================

=============================================
            Excecution  Summary
=============================================
Process      BT        AT      TAT       WT
P0           3         0       3         0
P1           6         0       9         3
P2           2         0       11         9
P3           8         0       19         11
=============================================

Average Turnaround time = 10.50
Average Waiting time = 5.75
rohit@iris:~/Programing/C/CSL204/Experiment 5$
```

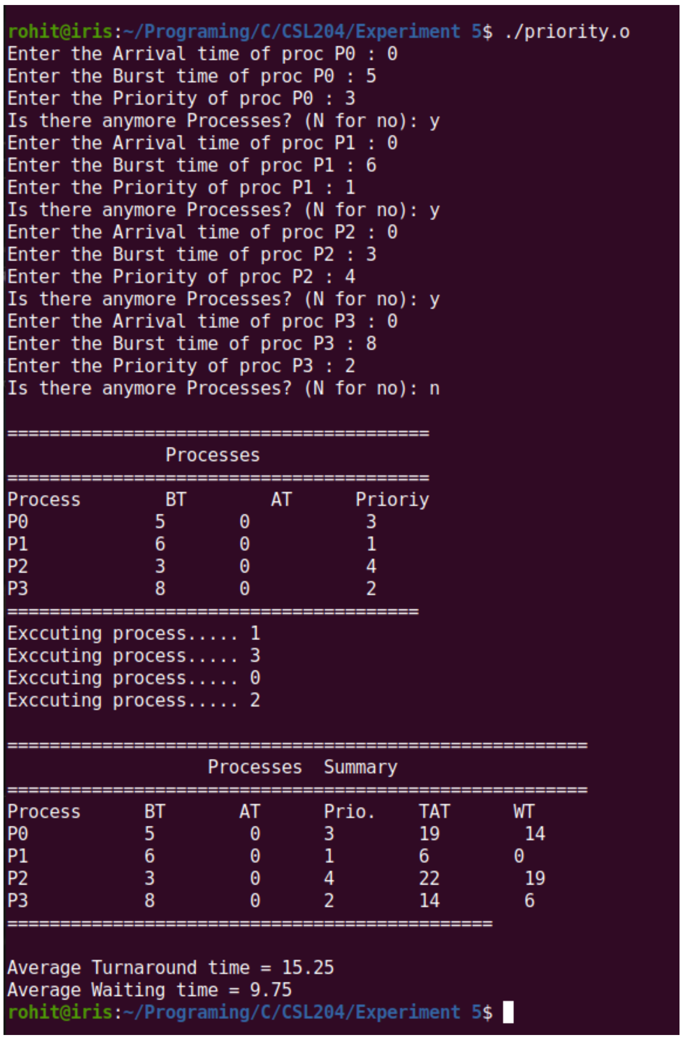
Priority Scheduling

```
#include<stdio.h>
#include<stdlib.h>
#define MAX_PROCESS 2

typedef struct {
    int arrival_time;
    int burst_time;
    int p_num;
    int pr;
    int comp;
} proc;

typedef struct{
    proc** list ;
    int size;
    int cap;
} proc_list;

typedef struct {
    int p_id;
    int tat;
    int wt;
} proc_summ;


proc_list *
 create_process()
{
    proc_list* p_list = (proc_list *) malloc(sizeof(proc_list));
    p_list->size = 0;
    p_list->cap = MAX_PROCESS;

    p_list->list = (proc**)malloc(sizeof(proc)*p_list->cap);

    int RUN=1;
    int i=0;
    int c;

    while(RUN)
    {
        proc *p = (proc*)malloc(sizeof(proc));
        printf("Enter the Arrival time of proc P%d : ",i);
        scanf("%d",&p->arrival_time);
        printf("Enter the Burst time of proc P%d : ",i);
        scanf("%d",&p->burst_time);
        printf("Enter the Priority of proc P%d : ",i);
        scanf("%d",&p->pr);
        p->p_num = i;
        p->comp = 0;

        p_list->list[i] = p;

        i++;
        p_list->size = i;

        printf("Is there anymore Processes? (N for no): ");
        getc(stdin);
        c = getc(stdin);
        if(c == 'n' || c == 'N'){
            RUN=0;
            break;
        }

        if(! i < p_list -> cap){
            p_list -> cap = p_list->cap*2;
            p_list->list= (proc**) realloc(p_list->list, sizeof(proc)*p_list-
>cap);
        }
    }

    return p_list;
}

void
 show_process(proc_list *procs)
{
    printf("\n");
    printf("========================================\n");
    printf("               Processes                \n");
    printf("========================================\n");
    printf("Process        BT        AT       Prioriy\n");
     for(unsigned int i=0;i<procs->size; i++){
    printf("P%d            %d        %d           %d\n",
            procs->list[i]->p_num,
            procs->list[i]->burst_time,
            procs->list[i]->arrival_time,
            procs->list[i]->pr);
     }
    printf("=======================================\n");

}

void
sort_at ( proc** list, int s)
 {
    for(int i=0 ; i<s; i++){
        for (int j=0 ;j<=s-i-1 ;j++){
            if(list[j]->arrival_time > list[j+1]->arrival_time){
                proc *temp = list[j];
                list[j] = list[j+1];
                list[j+1] = temp;
            }
        }
    }
}

int
get_process(proc_list *procs, int time)
 {
    int n = procs -> size;
    proc **list = procs -> list;

    int i=0;
    int smallest_pos = -1;
    while( i<n ){
        if(list[i] -> comp == 0 && list[i] -> arrival_time <= time)
            smallest_pos = i;
        i++;
    }

    if (smallest_pos == -1 ){
        printf("No Process to be excecuted at time %d\n", time);
        return -1;
    }

    for ( i=0; i < n; i++){

        if( time >= list[i] -> arrival_time && list[i]->comp != 1){
            if (smallest_pos >= 0 && list[i] -> pr < list[smallest_pos] -> pr){
                smallest_pos = i;
            }
        }
    }

    return smallest_pos;
}

void
show_result(proc_summ * psum, proc_list *procs)
 {
    int n = procs->size;
    printf("\n");
    printf("==================================================\n");
    printf("                   Processes  Summary                 \n");
    printf("==================================================\n");
    printf("Process      BT       AT      Prio.    TAT      WT     \n");
    int wt_sum = 0;
    int tat_sum = 0;
    for(unsigned int i=0;i<procs->size; i++){

        int pos = procs->list[i]->p_num;
    printf("P%d           %d         %d      %d        %d        %d  \n",
            procs->list[i]->p_num,
            procs->list[i]->burst_time,
            procs->list[i]->arrival_time,
            procs->list[i]->pr,
            psum[pos].tat,
            psum[pos].wt);
        wt_sum += psum[pos].wt;
        tat_sum += psum[pos].tat;

    }
    printf("==========================================\n");

    float average_wt = (float)wt_sum / n;
    float average_tat = (float)tat_sum / n;

    printf("\nAverage Turnaround time = %.2f\nAverage Waiting time = %.2f\
n",average_tat,average_wt);
}


proc_summ *
priority(proc_list *procs)
 {
    if ( procs->size > 0 ){

        int n = procs->size;
        proc_summ *ps = (proc_summ*) malloc (sizeof (proc_summ)*n);

        int time = 0;

        //Sort the process using their arrival time.
        sort_at(procs->list, n-1);
        //show_process(procs);

        //time = procs -> list[get_process(procs,time)] -> arrival_time;
        for (int i = 0 ; i < n; ){
            int p_index = get_process(procs, time);
            if( p_index != -1 ){
                printf("Exccuting process..... %d\n",procs->list[p_index] ->
p_num);

                int pos = procs->list[p_index]-> p_num;

                time += procs->list[p_index]->burst_time;

                ps[pos].p_id = pos;
                ps[pos].tat = time - procs -> list[p_index] -> arrival_time;
                ps[pos].wt = ps[pos].tat - procs -> list[p_index] -> burst_time;

                procs -> list[p_index] -> comp = 1;
                i++;
            }
            else{
                time++;
                if(time>1024){
                    printf("Time Limit Exceeded\n");
                    return NULL;
                }
            }

        }

        return ps;
    }
    return NULL;

}


void
main()
{
    proc_list *s = create_process();
    show_process(s);
    proc_summ *psum = priority(s);
    show_result(psum, s);
}
```

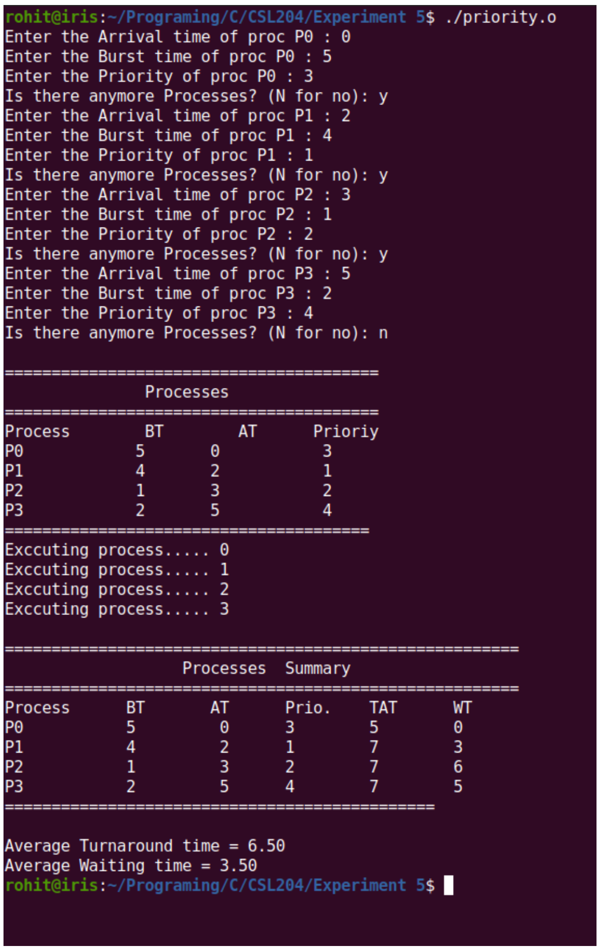
Screenshots:



```
rohit@iris:~/Programing/C/CSL204/Experiment 5$ ./priority.o
Enter the Arrival time of proc P0 : 5
Enter the Burst time of proc P0 : 2
Enter the Priority of proc P0 : 1
Is there anymore Processes? (N for no): y
Enter the Arrival time of proc P1 : 0
Enter the Burst time of proc P1 :
3
Enter the Priority of proc P1 : 2
Is there anymore Processes? (N for no): y
Enter the Arrival time of proc P2 : 3
Enter the Burst time of proc P2 : 6
Enter the Priority of proc P2 : 3
Is there anymore Processes? (N for no): n

========================================
               Processes
========================================
Process         BT        AT       Prioriy
P0              2       5           1
P1              3       0           2
P2              6       3           3
=======================================
Exccuting process..... 1
Exccuting process..... 2
Exccuting process..... 0

=======================================================
                   Processes  Summary
=======================================================
Process       BT        AT       Prio.     TAT       WT
P1            3         0        2         3         0
P2            6         3        3         6         0
P0            2         5        1         6         4
==============================================


Average Turnaround time = 5.00
Average Waiting time = 1.33
rohit@iris:~/Programing/C/CSL204/Experiment 5$
```



```
rohit@iris:~/Programing/C/CSL204/Experiment 5$ ./priority.o
Enter the Arrival time of proc P0 : 0
Enter the Burst time of proc P0 : 5
Enter the Priority of proc P0 : 3
Is there anymore Processes? (N for no): y
Enter the Arrival time of proc P1 : 0
Enter the Burst time of proc P1 : 6
Enter the Priority of proc P1 : 1
Is there anymore Processes? (N for no): y
Enter the Arrival time of proc P2 : 0
Enter the Burst time of proc P2 : 3
Enter the Priority of proc P2 : 4
Is there anymore Processes? (N for no): y
Enter the Arrival time of proc P3 : 0
Enter the Burst time of proc P3 : 8
Enter the Priority of proc P3 : 2
Is there anymore Processes? (N for no): n

========================================
               Processes
========================================
Process         BT        AT       Prioriy
P0              5       0           3
P1              6       0           1
P2              3       0           4
P3              8       0           2
=======================================
Exccuting process..... 1
Exccuting process..... 3
Exccuting process..... 0
Exccuting process..... 2

=======================================================
                   Processes  Summary
=======================================================
Process       BT        AT       Prio.     TAT       WT
P0            5         0        3         19         14
P1            6         0        1         6         0
P2            3         0        4         22         19
P3            8         0        2         14         6
==============================================

Average Turnaround time = 15.25
Average Waiting time = 9.75
rohit@iris:~/Programing/C/CSL204/Experiment 5$
```

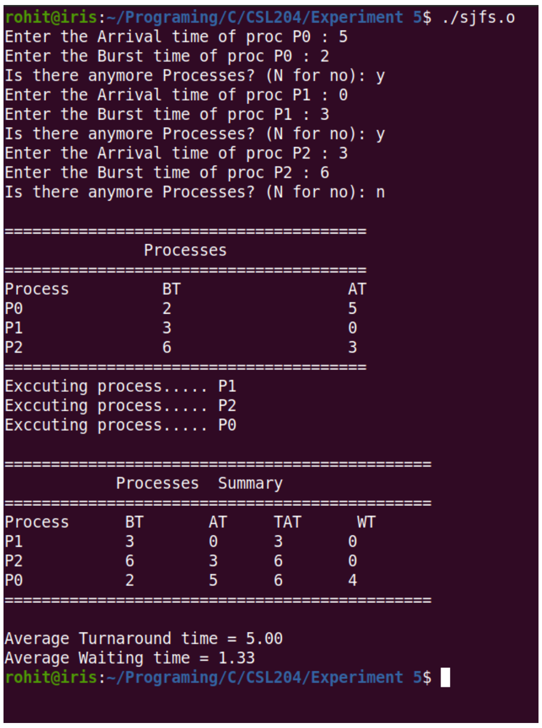


```
rohit@iris:~/Programing/C/CSL204/Experiment 5$ ./priority.o
Enter the Arrival time of proc P0 : 0
Enter the Burst time of proc P0 : 5
Enter the Priority of proc P0 : 3
Is there anymore Processes? (N for no): y
Enter the Arrival time of proc P1 : 2
Enter the Burst time of proc P1 : 4
Enter the Priority of proc P1 : 1
Is there anymore Processes? (N for no): y
Enter the Arrival time of proc P2 : 3
Enter the Burst time of proc P2 : 1
Enter the Priority of proc P2 : 2
Is there anymore Processes? (N for no): y
Enter the Arrival time of proc P3 : 5
Enter the Burst time of proc P3 : 2
Enter the Priority of proc P3 : 4
Is there anymore Processes? (N for no): n

========================================
               Processes
========================================
Process         BT        AT       Prioriy
P0              5       0           3
P1              4       2           1
P2              1       3           2
P3              2       5           4
=======================================
Excuting process..... 0
Excuting process..... 1
Excuting process..... 2
Excuting process..... 3

=======================================================
                   Processes  Summary
=======================================================
Process       BT        AT       Prio.     TAT       WT
P0            5         0        3         5         0
P1            4         2        1         7         3
P2            1         3        2         7         6
P3            2         5        4         7         5
==============================================

Average Turnaround time = 6.50
Average Waiting time = 3.50
rohit@iris:~/Programing/C/CSL204/Experiment 5$
```

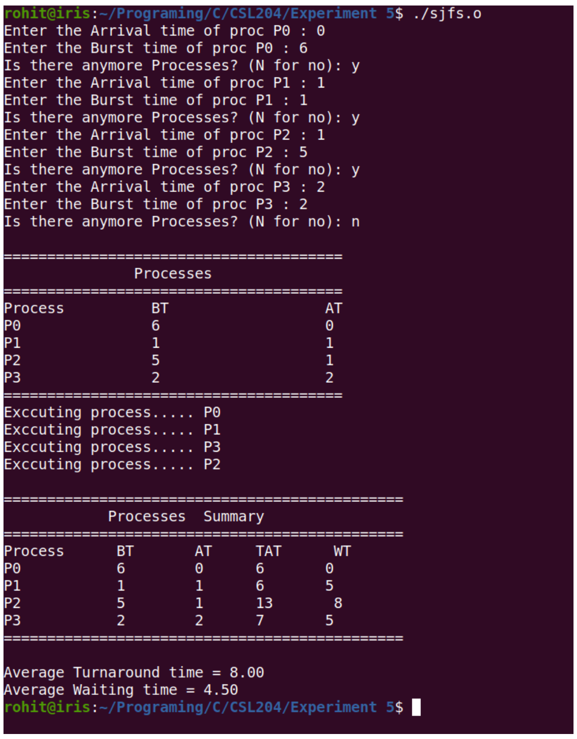
Shortest Job First Scheduling (SJFS)

```
#include<stdio.h>
#include<stdlib.h>
#define MAX_PROCESS 2

typedef struct {
    int arrival_time;
    int burst_time;
    int p_num;
    int comp;
} proc;

typedef struct{
    proc** list ;
    int size;
    int cap;
} proc_list;

typedef struct {
    int p_id;
    int tat;
    int wt;
} proc_summ;


proc_list *
 create_process()
{
    proc_list* p_list = (proc_list *) malloc(sizeof(proc_list));
    p_list->size = 0;
    p_list->cap = MAX_PROCESS;

    p_list->list = (proc**)malloc(sizeof(proc)*p_list->cap);

    int RUN=1;
    int i=0;
    int c;

    while(RUN)
    {
        proc *p = (proc*)malloc(sizeof(proc));
        printf("Enter the Arrival time of proc P%d : ",i);
        scanf("%d",&p->arrival_time);
        printf("Enter the Burst time of proc P%d : ",i);
        scanf("%d",&p->burst_time);
        p->p_num = i;
        p->comp = 0;

        p_list->list[i] = p;

        i++;
        p_list->size = i;

        printf("Is there anymore Processes? (N for no): ");
        getc(stdin);
        c = getc(stdin);
        if(c == 'n' || c == 'N'){
            RUN=0;
            break;
        }

        if(! i < p_list -> cap){
            p_list -> cap = p_list->cap*2;
            p_list->list= (proc**) realloc(p_list->list, sizeof(proc)*p_list-
>cap);
        }
    }

    return p_list;
}

void
 show_process(proc_list *procs)
{
    printf("\n");
    printf("=======================================\n");
    printf("               Processes              \n");
    printf("=======================================\n");
    printf("Process         BT                  AT\n");
     for(unsigned int i=0;i<procs->size; i++){
    printf("P%d              %d                   %d\n",
            procs->list[i]->p_num,
            procs->list[i]->burst_time,
            procs->list[i]->arrival_time);
     }
    printf("=======================================\n");

}

void
sort_at ( proc** list, int s)
 {
    for(int i=0 ; i<s; i++){
        for (int j=0 ;j<=s-i-1 ;j++){
            if(list[j]->arrival_time > list[j+1]->arrival_time){
                proc *temp = list[j];
                list[j] = list[j+1];
                list[j+1] = temp;
            }
        }
    }
}

int
get_process(proc_list *procs, int time)
 {
    int n = procs -> size;
    proc **list = procs -> list;

    int i=0;
    int smallest_pos = -1;
    while( i<n ){
        if(list[i] -> comp == 0 && list[i] -> arrival_time <= time)
            smallest_pos = i;
        i++;
    }

    if (smallest_pos == -1 ){
        printf("No Process to be excecuted at time %d\n", time);
        return -1;
    }

    for ( i=0; i < n; i++){

        if( time >= list[i] -> arrival_time && list[i]->comp != 1){
            if (smallest_pos >= 0 && list[i] -> burst_time < list[smallest_pos]
-> burst_time){
                smallest_pos = i;
            }
        }
    }

    return smallest_pos;
}

void
show_result(proc_summ * psum, proc_list *procs)
 {
    int n = procs->size;
    printf("\n");
    printf("==============================================\n");
    printf("            Processes  Summary                \n");
    printf("==============================================\n");
    printf("Process      BT       AT     TAT      WT     \n");
    int wt_sum = 0;
    int tat_sum = 0;
    for(unsigned int i=0;i<procs->size; i++){

        int pos = procs->list[i]->p_num;
        printf("P%d           %d        %d      %d       %d     \n",
            procs->list[i]->p_num,
            procs->list[i]->burst_time,
            procs->list[i]->arrival_time,
            psum[pos].tat,
            psum[pos].wt);
        wt_sum += psum[pos].wt;
        tat_sum += psum[pos].tat;

    }
    printf("==============================================\n");

    float average_wt = (float)wt_sum / n;
    float average_tat = (float)tat_sum / n;

    printf("\nAverage Turnaround time = %.2f\nAverage Waiting time = %.2f\
n",average_tat,average_wt);
}


proc_summ *
sjfs(proc_list *procs)
 {
    if ( procs->size > 0 ){

        int n = procs->size;
        proc_summ *ps = (proc_summ*) malloc (sizeof (proc_summ)*n);

        int time = 0;

        //Sort the process using their arrival time.
        sort_at(procs->list, n-1);
        //show_process(procs);

        //time = procs -> list[get_process(procs,time)] -> arrival_time;

        for (int i = 0 ; i < n; ){
            int p_index = get_process(procs, time);
            if( p_index != -1 ){
                int pos = procs->list[p_index]-> p_num;
                printf("Exccuting process..... P%d\n",pos);
                time += procs->list[p_index]->burst_time;

                ps[pos].p_id = pos;
                ps[pos].tat = time - procs -> list[p_index] -> arrival_time;
                ps[pos].wt = ps[pos].tat - procs -> list[p_index] -> burst_time;

                procs -> list[p_index] -> comp = 1;
                i++;
            }
            else{
                time++;
                if(time>1024){
                    printf("Time Limit Exceeded\n");
                    return NULL;
                }
            }

        }

        return ps;
    }
    return NULL;

}


void
main()
{
    proc_list *s = create_process();
    show_process(s);
    proc_summ *psum = sjfs(s);
    show_result(psum, s);
}
```

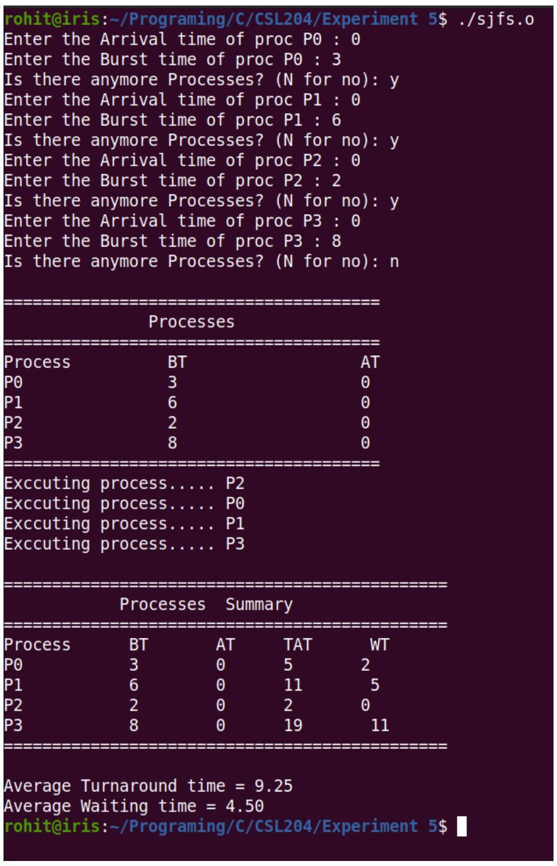
Screenshots



```
rohit@iris:~/Programing/C/CSL204/Experiment 5$ ./sjfs.o
Enter the Arrival time of proc P0 : 5
Enter the Burst time of proc P0 : 2
Is there anymore Processes? (N for no): y
Enter the Arrival time of proc P1 : 0
Enter the Burst time of proc P1 : 3
Is there anymore Processes? (N for no): y
Enter the Arrival time of proc P2 : 3
Enter the Burst time of proc P2 : 6
Is there anymore Processes? (N for no): n

=======================================
               Processes
=======================================
Process          BT                   AT
P0               2                    5
P1               3                    0
P2               6                    3
=======================================
Excciting process..... P1
Excciting process..... P2
Excciting process..... P0

==============================================
            Processes  Summary
==============================================
Process      BT       AT     TAT      WT
P1           3        0      3        0
P2           6        3      6        0
P0           2        5      6        4
==============================================

Average Turnaround time = 5.00
Average Waiting time = 1.33
rohit@iris:~/Programing/C/CSL204/Experiment 5$ 
```



```
rohit@iris:~/Programing/C/CSL204/Experiment 5$ ./sjfs.o
Enter the Arrival time of proc P0 : 0
Enter the Burst time of proc P0 : 6
Is there anymore Processes? (N for no): y
Enter the Arrival time of proc P1 : 1
Enter the Burst time of proc P1 : 1
Is there anymore Processes? (N for no): y
Enter the Arrival time of proc P2 : 1
Enter the Burst time of proc P2 : 5
Is there anymore Processes? (N for no): y
Enter the Arrival time of proc P3 : 2
Enter the Burst time of proc P3 : 2
Is there anymore Processes? (N for no): n

=======================================
               Processes
=======================================
Process          BT                   AT
P0               6                    0
P1               1                    1
P2               5                    1
P3               2                    2
=======================================
Excciting process..... P0
Excciting process..... P1
Excciting process..... P3
Excciting process..... P2

==============================================
            Processes  Summary
==============================================
Process      BT       AT     TAT      WT
P0           6        0      6        0
P1           1        1      6        5
P2           5        1      13        8
P3           2        2      7        5
==============================================

Average Turnaround time = 8.00
Average Waiting time = 4.50
rohit@iris:~/Programing/C/CSL204/Experiment 5$ 
```

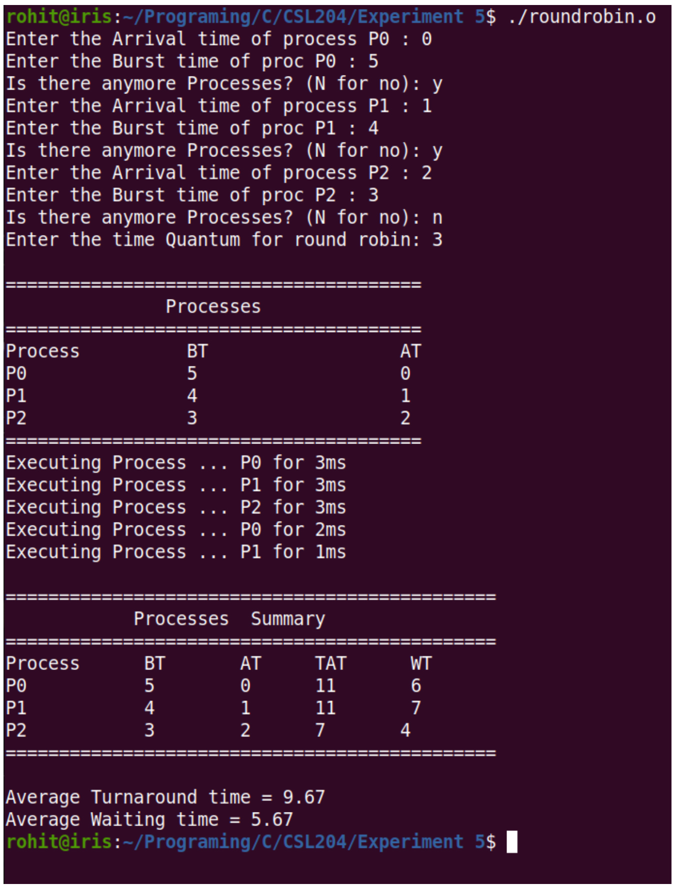


```
rohit@iris:~/Programing/C/CSL204/Experiment 5$ ./sjfs.o
Enter the Arrival time of proc P0 : 0
Enter the Burst time of proc P0 : 3
Is there anymore Processes? (N for no): y
Enter the Arrival time of proc P1 : 0
Enter the Burst time of proc P1 : 6
Is there anymore Processes? (N for no): y
Enter the Arrival time of proc P2 : 0
Enter the Burst time of proc P2 : 2
Is there anymore Processes? (N for no): y
Enter the Arrival time of proc P3 : 0
Enter the Burst time of proc P3 : 8
Is there anymore Processes? (N for no): n

=======================================
               Processes
=======================================
Process          BT                   AT
P0               3                    0
P1               6                    0
P2               2                    0
P3               8                    0
=======================================
Excciting process..... P2
Excciting process..... P0
Excciting process..... P1
Excciting process..... P3

==============================================
            Processes  Summary
==============================================
Process      BT       AT     TAT      WT
P0           3        0      5        2
P1           6        0      11        5
P2           2        0      2        0
P3           8        0      19        11
==============================================

Average Turnaround time = 9.25
Average Waiting time = 4.50
rohit@iris:~/Programing/C/CSL204/Experiment 5$ 
```

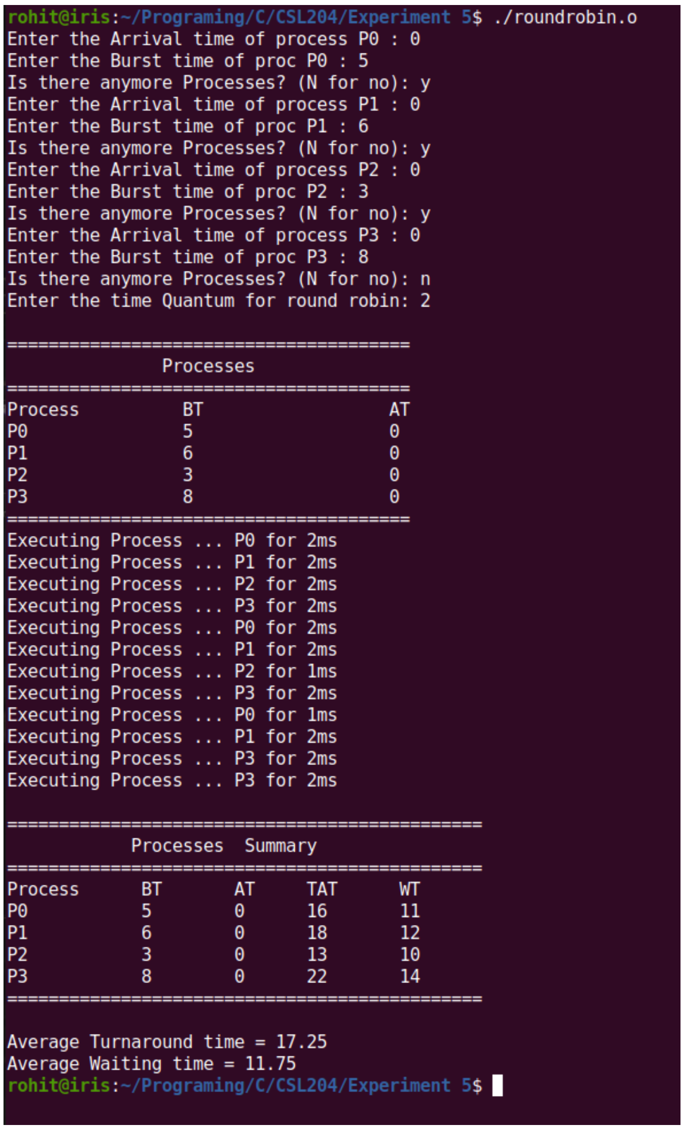
Round Robin Scheduling

```
#include<stdio.h>
#include<stdlib.h>
#define MAX_PROCESS 10

typedef struct {
    int arrival_time;
    int burst_time;
    int rem_time;
    int p_num;
    int comp;
} proc;

typedef struct{
    proc** list ;
    int size;
    int cap;
} proc_list;

typedef struct {
    int p_id;
    int tat;
    int wt;
} proc_summ;

proc_list *
 create_process()
{
    proc_list* p_list = (proc_list *) malloc(sizeof(proc_list));
    p_list->size = 0;
    p_list->cap = MAX_PROCESS;

    p_list->list = (proc**)malloc(sizeof(proc)*p_list->cap);

    int RUN=1;
    int i=0;
    int c;

    while(RUN)
    {
        proc *p = (proc*)malloc(sizeof(proc));
        printf("Enter the Arrival time of process P%d : ",i);
        scanf("%d",&p->arrival_time);
        printf("Enter the Burst time of proc P%d : ",i);
        scanf("%d",&p->burst_time);
        p->rem_time = p->burst_time;
        p->p_num = i;
        p->comp = 0;

        p_list->list[i] = p;

        i++;
        p_list->size = i;

        printf("Is there anymore Processes? (N for no): ");
        getc(stdin);
        c = getc(stdin);
        if(c == 'n' || c == 'N'){
            RUN=0;
            break;
        }
        if(! i < p_list -> cap){
            p_list -> cap = p_list->cap*2;
            p_list->list= (proc**) realloc(p_list->list, sizeof(proc)*p_list-
>cap);
        }
    }

    return p_list;
}

void
 show_process(proc_list *procs)
{
    printf("\n");
    printf("=======================================\n");
    printf("              Processes               \n");
    printf("=======================================\n");
    printf("Process         BT                  AT\n");
     for(unsigned int i=0;i<procs->size; i++){
    printf("P%d              %d                  %d\n",
            procs->list[i]->p_num,
            procs->list[i]->burst_time,
            procs->list[i]->arrival_time);
     }
    printf("=======================================\n");

}

void
sort_at ( proc** list, int s)
 {
    for(int i=0 ; i<s; i++){
        for (int j=0 ;j<=s-i-1 ;j++){
            if(list[j]->arrival_time > list[j+1]->arrival_time){
                proc *temp = list[j];
                list[j] = list[j+1];
                list[j+1] = temp;
            }
        }
    }
}

void
show_result(proc_summ * psum, proc_list *procs)
 {
    if (psum == NULL){
        printf("Process summary cant be null ERROR!!!!!\n");
        return;
    }

    int n = procs->size;
    printf("\n");
    printf("==============================================\n");
    printf("           Processes  Summary                \n");
    printf("==============================================\n");
    printf("Process      BT       AT     TAT      WT     \n");
    int wt_sum = 0;
    int tat_sum = 0;
    for(unsigned int i=0;i<procs->size; i++){

        int pos = procs->list[i]->p_num;
        printf("P%d           %d       %d      %d       %d     \n",
            procs->list[i]->p_num,
            procs->list[i]->burst_time,
            procs->list[i]->arrival_time,
            psum[pos].tat,
            psum[pos].wt);
        wt_sum += psum[pos].wt;
        tat_sum += psum[pos].tat;

    }
    printf("==============================================\n");

    float average_wt = (float)wt_sum / n;
    float average_tat = (float)tat_sum / n;

    printf("\nAverage Turnaround time = %.2f\nAverage Waiting time = %.2f\
n",average_tat,average_wt);
}


proc_summ *
round_robin(proc_list *procs, int time_q)
 {
    if ( procs->size > 0 ){

        int n = procs->size;
        int time = 0;        // Time in ms
        int proc_comp = 0;  // Number of process completed

        // Stores the summary of all the process.
        proc_summ *ps = (proc_summ*) malloc (sizeof (proc_summ)*n);

        //Sort the process using their arrival time.
        sort_at(procs->list, n-1);

        while ( proc_comp < n ) {
            //Execute this loop until all the process is completed

            int idle = 1;
            //Update the ready queue, ie, add any process that is not completed.
            for( int i =0 ; i< n; i++){
                proc *p = procs -> list[i];
                if ( p -> comp != 1 && time >= p-> arrival_time ){
                    // add the process to the ready queue
                    if (idle) idle = 0;
                    int ex_time = p->rem_time > time_q? time_q : p->rem_time ;
                    int index = p->p_num;
                    printf("Executing Process ... P%d for %dms\n", index,
ex_time);

                    time += ex_time;
                    p->rem_time -= ex_time;
                    if (p -> rem_time == 0){
                        p->comp = 1;
                        ps[ index ] . tat = time - p -> arrival_time;
                        ps[ index ] . wt = ps[ index ] . tat - p-> burst_time;
                        proc_comp += 1;
                    }
                }
            }
            if (idle){
                time++;
            }

            if(time > 1024){
                printf("Time Limit exceeded");
                return NULL;
            }
            // Execute the process in the ready queue
        }
        return ps;
    }
    return NULL;

}


void
main()
{
    proc_list *s = create_process();
    printf("Enter the time Quantum for round robin: ");
    int tq;
    scanf("%d", &tq);
    show_process(s);
    proc_summ *psum = round_robin(s,tq);
    show_result(psum, s);
}
```

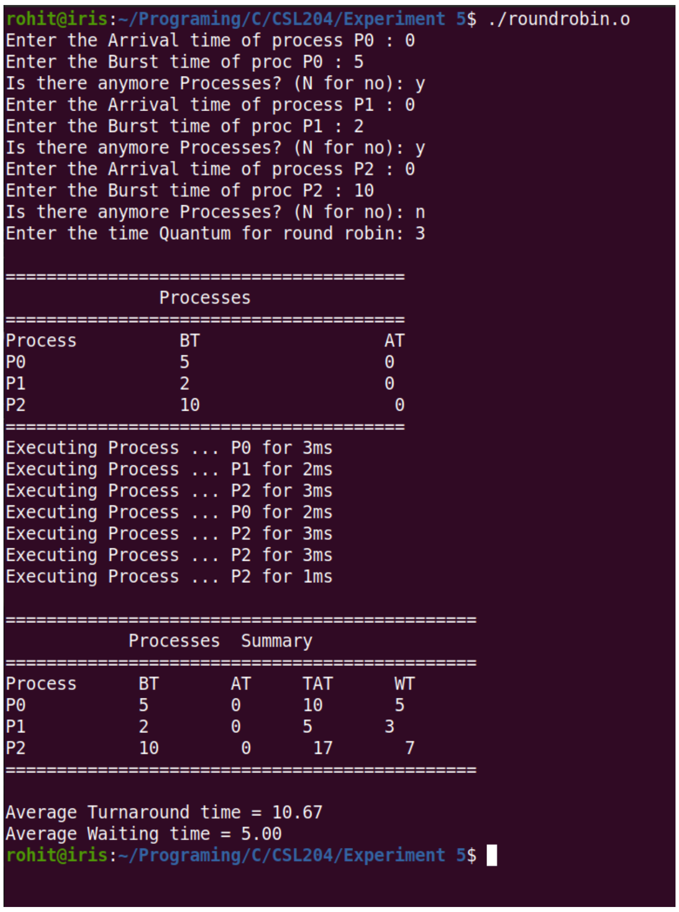
Screenshots:



```
rohit@iris:~/Programing/C/CSL204/Experiment 5$ ./roundrobin.o
Enter the Arrival time of process P0 : 0
Enter the Burst time of proc P0 : 5
Is there anymore Processes? (N for no): y
Enter the Arrival time of process P1 : 1
Enter the Burst time of proc P1 : 4
Is there anymore Processes? (N for no): y
Enter the Arrival time of process P2 : 2
Enter the Burst time of proc P2 : 3
Is there anymore Processes? (N for no): n
Enter the time Quantum for round robin: 3

========================================
               Processes
========================================
Process          BT                   AT
P0              5                    0
P1              4                    1
P2              3                    2
========================================
Executing Process ... P0 for 3ms
Executing Process ... P1 for 3ms
Executing Process ... P2 for 3ms
Executing Process ... P0 for 2ms
Executing Process ... P1 for 1ms

===============================================
            Processes  Summary
===============================================
Process      BT       AT      TAT      WT
P0           5        0       11       6
P1           4        1       11       7
P2           3        2       7       4
===============================================

Average Turnaround time = 9.67
Average Waiting time = 5.67
rohit@iris:~/Programing/C/CSL204/Experiment 5$
```



```
rohit@iris:~/Programing/C/CSL204/Experiment 5$ ./roundrobin.o
Enter the Arrival time of process P0 : 0
Enter the Burst time of proc P0 : 5
Is there anymore Processes? (N for no): y
Enter the Arrival time of process P1 : 0
Enter the Burst time of proc P1 : 6
Is there anymore Processes? (N for no): y
Enter the Arrival time of process P2 : 0
Enter the Burst time of proc P2 : 3
Is there anymore Processes? (N for no): y
Enter the Arrival time of process P3 : 0
Enter the Burst time of proc P3 : 8
Is there anymore Processes? (N for no): n
Enter the time Quantum for round robin: 2

========================================
               Processes
========================================
Process          BT                   AT
P0              5                    0
P1              6                    0
P2              3                    0
P3              8                    0
========================================
Executing Process ... P0 for 2ms
Executing Process ... P1 for 2ms
Executing Process ... P2 for 2ms
Executing Process ... P3 for 2ms
Executing Process ... P0 for 2ms
Executing Process ... P1 for 2ms
Executing Process ... P2 for 1ms
Executing Process ... P3 for 2ms
Executing Process ... P0 for 1ms
Executing Process ... P1 for 2ms
Executing Process ... P3 for 2ms
Executing Process ... P3 for 2ms

===============================================
            Processes  Summary
===============================================
Process      BT       AT      TAT      WT
P0           5        0       16       11
P1           6        0       18       12
P2           3        0       13       10
P3           8        0       22       14
===============================================

Average Turnaround time = 17.25
Average Waiting time = 11.75
rohit@iris:~/Programing/C/CSL204/Experiment 5$
```



```
rohit@iris:~/Programing/C/CSL204/Experiment 5$ ./roundrobin.o
Enter the Arrival time of process P0 : 0
Enter the Burst time of proc P0 : 5
Is there anymore Processes? (N for no): y
Enter the Arrival time of process P1 : 0
Enter the Burst time of proc P1 : 2
Is there anymore Processes? (N for no): y
Enter the Arrival time of process P2 : 0
Enter the Burst time of proc P2 : 10
Is there anymore Processes? (N for no): n
Enter the time Quantum for round robin: 3

========================================
               Processes
========================================
Process          BT                   AT
P0              5                    0
P1              2                    0
P2              10                    0
========================================
Executing Process ... P0 for 3ms
Executing Process ... P1 for 2ms
Executing Process ... P2 for 3ms
Executing Process ... P0 for 2ms
Executing Process ... P2 for 3ms
Executing Process ... P2 for 3ms
Executing Process ... P2 for 1ms

===============================================
            Processes  Summary
===============================================
Process      BT       AT      TAT      WT
P0           5        0       10       5
P1           2        0       5       3
P2           10        0       17       7
===============================================

Average Turnaround time = 10.67
Average Waiting time = 5.00
rohit@iris:~/Programing/C/CSL204/Experiment 5$
```