**Problem 3:** Write a program to implement the shortest job first non-preemptive scheduling algorithm and find the average turnaround time, waiting time, completion time and response time for overall process. Also Print Gantt chart for it.

## Solution:

Source Code:

```
#include<stdio.h>

#include<string.h>

#include<stdlib.h>


typedef struct

{

   char process_name[3];

   int arrival_time;

   int burst_time;

   int complete_time;

   int turn_around_time;

   int wait_time;

   int response_time;

} process;
```

```
void print_process_table(process arr[],int n){

   int i;

   puts(" _________________ _________________ _____________ _________________ __
_____________________ _____________ _____________________");

   puts("| Process Name | Arrival Time  | Burst Time | Complete Time | Turn Aroun
d Time | Wait Time | Response Time |");

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

      puts("|_________________|_________________|_____________|_________________
_|_____________________|_____________|_____________________|");

      printf("|     %3s     |     %3d     |     %3d     |     %3d     |      %4d      |    %3d    |
   %3d      |\n",

      arr[i].process_name,arr[i].arrival_time,arr[i].burst_time,arr[i].complete_time,ar
r[i].turn_around_time,arr[i].wait_time,arr[i].response_time);

   }

   puts("|_________________|_________________|_____________|_________________|
_____________________|_____________|_____________________|");



}



void get_average(process arr[], int n){

   double tat=0,wt=0,rt=0;

   int i;

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

      tat += (double)arr[i].turn_around_time;
```

```
        wt += (double)arr[i].wait_time;

        rt += (double)arr[i].response_time;

    }

    printf("Total time to Complete = %3d        Average Time to Complete = %.3f\n",arr
[n-1].complete_time,(double)arr[n-1].complete_time/(double)n);

    printf("Total Turn Around Time = %.3f     Average Turn Around Time = %.3f\n",tat,
tat/(double)n);

    printf("Total Waiting Time = %.3f         Average Waiting Time = %.3f\n",wt,wt/(dou
ble)n);

    printf("Total Response Time = %.3f        Average Response Time = %.3f\n",rt,rt/(do
uble)n);

}



void gantt(process arr[],int n){

    int i,j;

    // upper row

    printf(" ");

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

        for(j=0;j<arr[i].burst_time+1;j++) printf("__");

        printf(" ");

    }

    printf("\n|");

    // middle row
```

```
for(i=0;i<n;i++){
    for(j=0;j<arr[i].burst_time-1;j++){
        printf(" ");
    }
    printf("%3s",arr[i].process_name);
    for(j=0;j<arr[i].burst_time;j++){
        printf(" ");
    }
    printf("|");
}
printf("\n|");
// lower row
for(i=0; i<n;i++){
    for(j=0;j<arr[i].burst_time+1;j++) printf("__");
    printf("|");
}
printf("\n");
printf("0");
for(i=0; i<n; i++) {
    for(j=0; j<arr[i].burst_time+1; j++) printf("  ");
    if(arr[i].turn_around_time > 9) printf("\b");
```

```
        printf("%d", arr[i].complete_time);


    }
    printf("\n");
}

void swap(process arr[],int ind1, int ind2){
    process temp = arr[ind1];
    arr[ind1] = arr[ind2];
    arr[ind2] = temp;


}

void main()
{
    int n =0,i,ct=0, mt=0,j, temp;
    printf("Enter the number of processes\t");
    scanf("%d",&n);
    process arr[n];
    printf("Enter PROCESS_NAME ARRIVAL_TIME and BURST_TIME\n");
    for(i=0; i<n;i++)
```

```
{
    scanf("%s %d %d",arr[i].process_name,&arr[i].arrival_time,&arr[i].burst_time);
}

 // calculating completion time
for(j=0;j<n;j++){
    mt=arr[j].burst_time;
    for(i=j+1;i<n;i++){
        if(arr[i].arrival_time<=ct && arr[i].burst_time<mt){
            swap(arr,j,i);
        }
        if(ct<arr[i].arrival_time){
            break;
        }
    }
    if(j==0){
        temp=0;
    }else{
        temp = arr[j-1].complete_time;
    }
    arr[j].complete_time=arr[j].burst_time+temp;
```

```
        arr[j].turn_around_time = arr[j].complete_time -  arr[j].arrival_time;
        arr[j].wait_time = arr[j].turn_around_time - arr[j].burst_time;
        arr[j].response_time = arr[j].wait_time;
        ct=arr[j].complete_time;
    }
    print_process_table(arr,n);
    get_average(arr, n);
    puts("---------------------- GANTT CHART -------------------");
    gantt(arr,n);
}
```

# Output:

```
Enter the number of processes   4
Enter PROCESS_NAME ARRIVAL_TIME and BURST_TIME
p01 0 6
p02 2 9
p03 3 3
p04 4 2
 _____________ _______________ _____________ _______________ _________________ _____________ _________________
| Process Name | Arrival Time  | Burst Time  | Complete Time | Turn Around Time | Wait Time   | Response Time |
|_____________|_______________|_____________|_______________|_________________|_____________|_________________|
|      p01    |        0      |       6     |        6      |         6       |      0      |        0      |
|_____________|_______________|_____________|_______________|_________________|_____________|_________________|
|      p04    |        4      |       2     |        8      |         4       |      2      |        2      |
|_____________|_______________|_____________|_______________|_________________|_____________|_________________|
|      p03    |        3      |       3     |       11      |         8       |      5      |        5      |
|_____________|_______________|_____________|_______________|_________________|_____________|_________________|
|      p02    |        2      |       9     |       20      |        18       |      9      |        9      |
|_____________|_______________|_____________|_______________|_________________|_____________|_________________|
Total time to Complete =  20        Average Time to Complete = 5.000
Total Turn Around Time = 36.000     Average Turn Around Time = 9.000
Total Waiting Time = 16.000         Average Waiting Time = 4.000
Total Response Time = 16.000        Average Response Time = 4.000
---------------------- GANTT CHART -------------------
 _____________ _____ _______ ____________________
|     p01     | p04 |  p03  |        p02         |
|_____________|_____|_______|____________________|
0             6     8       11                   20
```