NAME: Divyesh Mali
ROLL NO: 27
PRACTICAL NO1

```
#include<stdio.h>
//structure representing a structure struct
priority_scheduling{

   //name of the process
char process_name;

   //time required for execution

int burst_time;     //waiting time

of process     int waiting_time;

   // total time of execution
int turn_around_time;

   // priority of the process

int priority; };     int main()

{

   // total number of processes
int number_of_process;

   // total waiting and turnaround time
int total = 0;

   // temporary structure for swapping
struct priority_scheduling temp_process;

   // ASCII numbers are used to represent the name of the process
int ASCII_number = 65;

   // swapping position
int position;

   // average waiting time of the process
float average_waiting_time;

   // average turnaround time of the process
float average_turnaround_time;     printf("Enter
the total number of Processes:");     // get the
total number of the process as input
scanf("%d", & number_of_process);
```

```
   // initializing the structure array
   struct priority_scheduling process[number_of_process];

printf("\nPlease Enter the Burst Time and Priority of each process:\n");

   // get burst time and priority of all process
for(int i=0;i<number_of_process;i++){

   //assign names consecutively using ASCII number
process[i].process_name = (char)ASCII_number;

   printf("\nenter the detailof the process%c\n",process[i].process_name);
printf("enter the burst time;");     scanf("%d", & process[i].burst_time);

   printf("enter the priority:");
scanf("%d",&process[i].priority);

   // increment the ASCII number to get the next alphabet
ASCII_number++;
}
   //swap process according to high priority

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

position = i;     for(int j= i+1; j<

number_of_process; j++) {

   // check if priority is higher for swapping     if
(process[j].priority > process[position].priority)
position = j;
   }
   // swapping of lower priority process with the higher priority process
temp_process = process[i]; process[i] = process[position];
process[position] = temp_process;
}
// First process will not have to wait and hence has a waiting time of 0
process[0].waiting_time = 0;

for (int i = 1; i<number_of_process; i++) { process[i].waiting_time
= 0;

for (int j = 0; j<i; j++) {     // calculate waiting time
process[i].waiting_time += process[j].burst_time; }

   // calculate total waiting time
total += process[i].waiting_time;
```

```
}

// calculate average waiting time
average_waiting_time = (float) total / (float) number_of_process;

// assigning total as 0 for next calculations total
= 0;

printf("\n\nProcess_name \t Burst Time \t Waiting Time \t Turnaround Time");

printf("\n------------------------------------------------------------\n"); for (int i = 0; i<

number_of_process; i++) {

// calculating the turnaround time of the processes
process[i].turn_around_time = process[i].burst_time + process[i].waiting_time;

// calculating the total turnaround time. total
+= process[i].turn_around_time;

// printing all the values

printf("\t %c \t\t%d\t\t%d\t\t%d",process[i].process_name, process[i].burst_time,
process[i].waiting_time, process[i].turn_around_time);

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

// calculating the average turn_around time
average_turnaround_time = (float) total / (float) number_of_process;

// average waiting time printf("\n\n Average Waiting Time:%f",

average_waiting_time);

// average turnaround time
printf("\n Average Turnaround Time: %f", average_turnaround_time);

return 0;
}
```

OUTPUT:

Enter the total number of Processes:5

Please Enter the Burst Time and Priority of each process:

enter the detailof the processA
enter the burst time;1 enter the
priority:2

enter the detailof the processB
enter the burst time;2 enter the
priority:3