5. Construct a scheduling program with C that selects the waiting process with the highest priority to execute next.

PROGRAM :

```
#include<stdio.h>
struct priority_scheduling {
char process_name;
int burst_time;
int waiting_time;
int turn_around_time;
int priority;
};
int main() {
int number_of_process;
int total = 0;
struct priority_scheduling temp_process;
int ASCII_number = 65;
int position;
float average_waiting_time;
float average_turnaround_time;
printf("Enter the total number of Processes: ");
scanf("%d", &number_of_process);
struct priority_scheduling process[number_of_process];
printf("\nPlease Enter the Burst Time and Priority of each process:\n");
for (int i = 0; i < number_of_process; i++)
{
process[i].process_name = (char) ASCII_number;
printf("\nEnter the  details of  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);
 ASCII_number++;
}
for (int i = 0; i < number_of_process; i++) {
position = i;
for (int j = i + 1; j < number_of_process; j++) {
if (process[j].priority > process[position].priority)
position = j;
}
temp_process = process[i];
 process[i] = process[position];
 process[position] = temp_process;
 }
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++) {
process[i].waiting_time += process[j].burst_time;
}
total += process[i].waiting_time;
}

average_waiting_time = (float) total / (float) number_of_process;
total = 0;
printf("\n\nProcess_name \t Burst Time \t Waiting Time \t  Time\n");
printf("_____________________________________________________________\n");
for (int i = 0; i < number_of_process; i++)
{
process[i].turn_around_time = process[i].burst_time + process[i].waiting_time;
total+=process[i].turn_around_time;
```

```
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");
}
average_turnaround_time = (float) total / (float) number_of_process;
printf("\n\n Average Waiting Time : %f", average_waiting_time);
printf("\n Average Turnaround Time: %f\n", average_turnaround_time);
return 0;
}
```

OUTPUT :

```
Enter the  details of  the process A
Enter the burst time: 2
Enter the priority: 1

Enter the  details of  the process B
Enter the burst time: 10
Enter the priority: 3

Enter the  details of  the process C
Enter the burst time: 6
Enter the priority: 2


Process_name     Burst Time      Waiting Time     Time
-------------------------------------------------------
         B               10              0               10
 -------------------------------------------------------
         C               6               10              16
 -------------------------------------------------------
         A               2               16              18
 -------------------------------------------------------


 Average Waiting Time : 8.666667
 Average Turnaround Time: 14.666667

--------------------------------
Process exited after 25.81 seconds with return value 0
Press any key to continue . . .
```