## EXPERIMENT 5

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

**Aim:-** Construct a scheduling program with C that selects the waiting process with the highest priority to execute next.

## Algorithm:-

* 1. Initialize the necessary data structures to store process information, including process ID, arrival time, burst time, and priority.
  2. Read the number of processes (N) from the user.
  3. For each process, read the following information:
  4. Process ID (PID)
  5. Arrival Time
  6. Burst Time (time required for execution)
  7. Priority (lower values indicate higher priority)
  8. Sort the processes based on their arrival time in ascending order.
  9. Initialize a variable current\_time to 0 (representing the current time in the simulation).
  10. Initialize a variable completed to 0 (to keep track of the number of completed processes).
  11. Create a priority queue or data structure to store processes based on their priority.
  12. Initialize a variable total\_waiting\_time to 0.
  13. While there are still processes to execute (i.e., completed < N), repeat the following:
      1. For each process that has arrived but has not been completed, add it to the priority queue.
      2. Pop the process with the highest priority from the queue.
      3. Calculate the waiting time for the process as current\_time - arrival time.
      4. Add the waiting time to total\_waiting\_time.
      5. Update current\_time by adding the process's burst time.
      6. Print the process ID, arrival time, burst time, waiting time, and turnaround time.
      7. Mark the process as completed.
  14. Calculate the average waiting time as total\_waiting\_time / N. 15. Print the average waiting time.

A screenshot of a computer

Description automatically generated