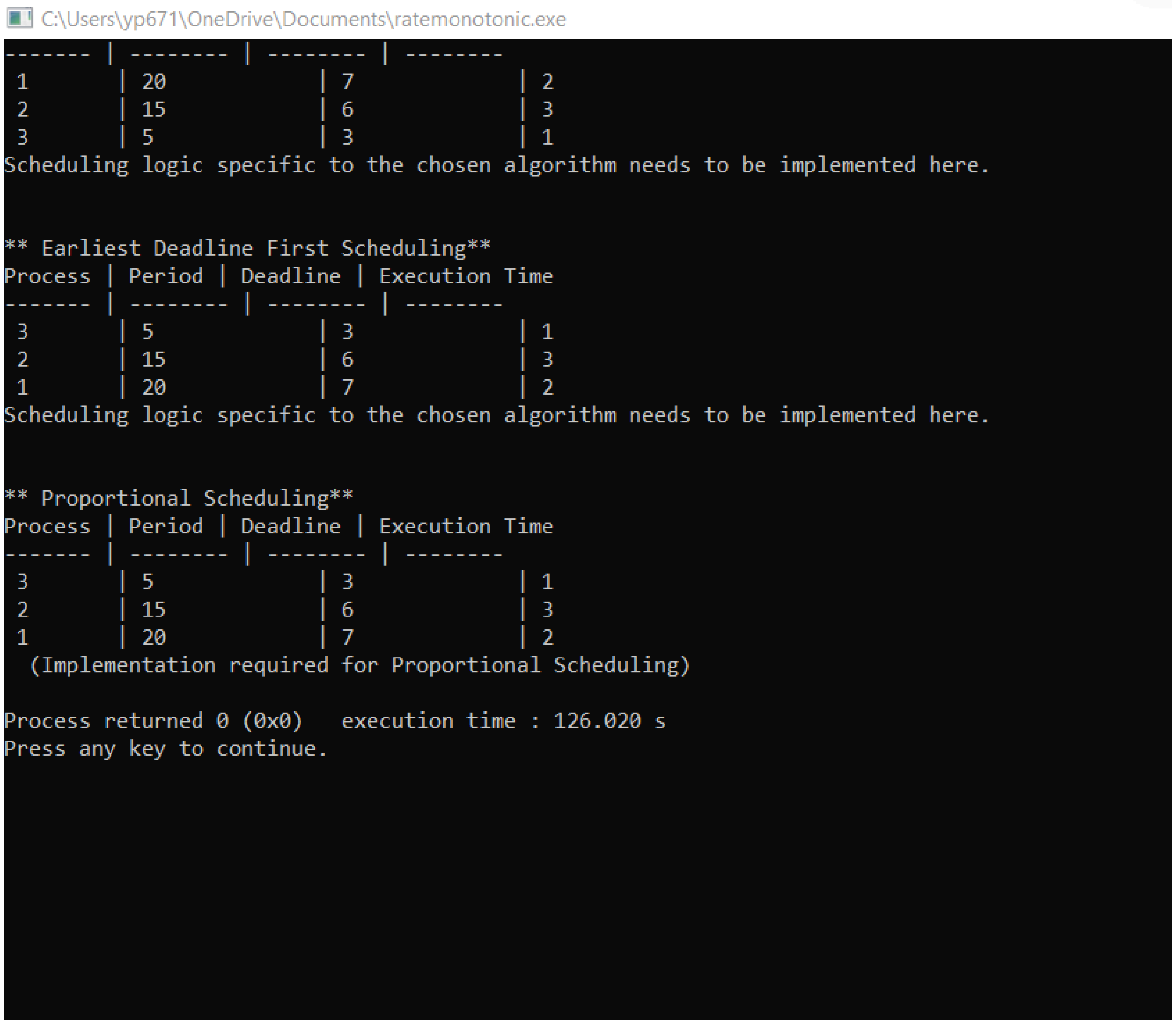
4.program to find rate monotonic,earliest deadline first scheduling,proportional scheduling

```
#include <stdio.h>
#define MAX_PROCESS 100
struct process {
  int pid;
  int period;
  int deadline;
  int execution_time;
void swap(struct process* a, struct process* b) {
  struct process temp = *a;
  *a = *b;
  *b = temp;
}
void sort_by_period(struct process proc[], int n) {
  for (int i = 0; i < n - 1; i++) {
    for (int j = 0; j < n - i - 1; j++) {
      if (proc[j].period > proc[j + 1].period) {
        swap(&proc[j], &proc[j + 1]);
      }
    }
  }
}
void sort_by_deadline(struct process proc[], int n) {
  for (int i = 0; i < n - 1; i++) {
    for (int j = 0; j < n - i - 1; j++) {
      if (proc[j].deadline > proc[j + 1].deadline) {
        swap(&proc[j], &proc[j + 1]);
      }
    }
  }
}
void schedule(struct process proc[], int n) {
  printf("Scheduling logic specific to the chosen algorithm needs to be implemented here.\n");
}
void print_table_header() {
  printf("Process | Period | Deadline | Execution Time\n");
  printf("------- | -------- | -------- | --------\n");
}
void print_process_info(struct process proc) {
  printf(" %d \t | %d \t\t | %d \t\t | %d \n", proc.pid, proc.period, proc.deadline,
proc.execution_time);
}
```

```
int main() {
  int n, i;
  struct process proc[MAX_PROCESS];
  printf("Enter the number of processes: ");
  scanf("%d", &n);
  printf("Enter details of processes:\n");
  for (i = 0; i < n; i++) {
    printf("Process ID: ");
    scanf("%d", &proc[i].pid);
    printf("Period: ");
    scanf("%d", &proc[i].period);
    printf("Deadline: ");
    scanf("%d", &proc[i].deadline);
    printf("Execution Time: ");
    scanf("%d", &proc[i].execution_time);
  }
  printf("\nScheduling Results:\n");
  printf("\n** Rate Monotonic Scheduling**\n");
  print_table_header();
  for (i = 0; i < n; i++) {
    print_process_info(proc[i]);
  }
  sort_by_period(proc, n);
  schedule(proc, n);
  printf("\n\n** Earliest Deadline First Scheduling**\n");
  print_table_header();
  for (i = 0; i < n; i++) {
    print_process_info(proc[i]);
  }
  sort_by_deadline(proc, n);
  schedule(proc, n);
  printf("\n\n** Proportional Scheduling**\n");
  print_table_header();
  for (i = 0; i < n; i++) {
    print_process_info(proc[i]);
  }
  printf("  (Implementation required for Proportional Scheduling)\n");
  return 0;
}
```



C:\Users\yp671\OneDrive\Documents\ratemonotonic.exe

```
------- | -------- | -------- | --------
 1       | 20              | 7               | 2
 2       | 15              | 6               | 3
 3       | 5               | 3               | 1
Scheduling logic specific to the chosen algorithm needs to be implemented here.


** Earliest Deadline First Scheduling**
Process | Period | Deadline | Execution Time
------- | -------- | -------- | --------
 3       | 5               | 3               | 1
 2       | 15              | 6               | 3
 1       | 20              | 7               | 2
Scheduling logic specific to the chosen algorithm needs to be implemented here.


** Proportional Scheduling**
Process | Period | Deadline | Execution Time
------- | -------- | -------- | --------
 3       | 5               | 3               | 1
 2       | 15              | 6               | 3
 1       | 20              | 7               | 2
  (Implementation required for Proportional Scheduling)

Process returned 0 (0x0)   execution time : 126.020 s
Press any key to continue.
```