OS EARLIEST DEADLINE FIRST

3. Write a C program to simulate Real-Time CPU Scheduling Algorithms:
b) Earliest-deadline First

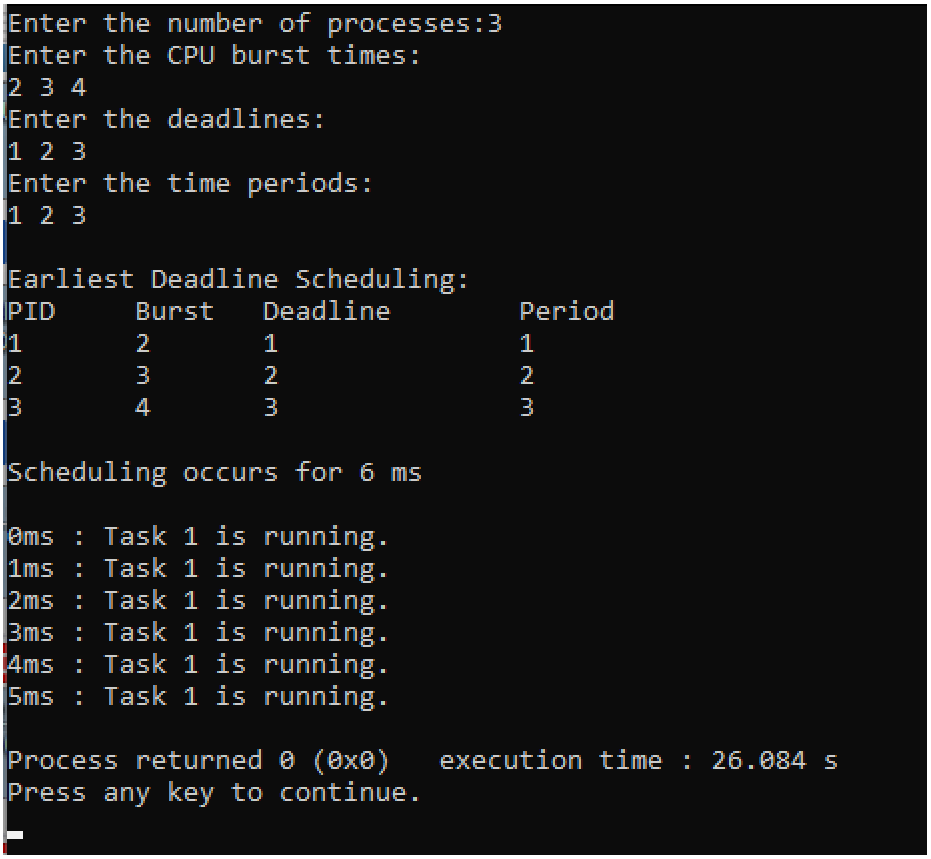
Code:

```
#include <stdio.h>
#include <limits.h>

#define MAX 10

struct Process {
    int pid;
    int burst;
    int deadline;
    int period;
    int remaining_time;
    int next_deadline;
};

int gcd(int a, int b) {
    return (b == 0) ? a : gcd(b, a % b);
}

int lcm(int a, int b) {
    return (a * b) / gcd(a, b);
}

int find_lcm(int arr[], int n) {
    int res = arr[0];
    for (int i = 1; i < n; i++) {
        res = lcm(res, arr[i]);
    }
    return res;
}

int main() {
    int n;
    struct Process p[MAX];
```

```
printf("Enter the number of processes:");
scanf("%d", &n);

printf("Enter the CPU burst times:\n");
for (int i = 0; i < n; i++) {
    scanf("%d", &p[i].burst);
    p[i].pid = i + 1;
    p[i].remaining_time = p[i].burst;
}

printf("Enter the deadlines:\n");
for (int i = 0; i < n; i++) {
    scanf("%d", &p[i].deadline);
    p[i].next_deadline = p[i].deadline;
}

printf("Enter the time periods:\n");
int periods[MAX];
for (int i = 0; i < n; i++) {
    scanf("%d", &p[i].period);
    periods[i] = p[i].period;
}

int lcm_val = find_lcm(periods, n);
printf("\nEarliest Deadline Scheduling:\n");
printf("PID\tBurst\tDeadline\tPeriod\n");
for (int i = 0; i < n; i++) {
    printf("%d\t%d\t%d\t\t%d\n", p[i].pid, p[i].burst, p[i].deadline, p[i].period);
}

printf("\nScheduling occurs for %d ms\n\n", lcm_val);

   for (int time = 0; time < lcm_val; time++) {
      for (int i = 0; i < n; i++) {
        if (time % p[i].period == 0) {
            p[i].remaining_time = p[i].burst;
            p[i].next_deadline = time + p[i].deadline;
        }
    }
```

```
        int min_deadline = INT_MAX;
        int current = -1;
        for (int i = 0; i < n; i++) {
            if (p[i].remaining_time > 0 && p[i].next_deadline < min_deadline) {
                min_deadline = p[i].next_deadline;
                current = i;
            }
        }

        if (current != -1) {
            printf("%dms : Task %d is running.\n", time, p[current].pid);
            p[current].remaining_time--;
        } else {
            printf("%dms : CPU is idle.\n", time);
        }
    }

    return 0;
}
```

Output:



```
Enter the number of processes:3
Enter the CPU burst times:
2 3 4
Enter the deadlines:
1 2 3
Enter the time periods:
1 2 3

Earliest Deadline Scheduling:
PID     Burst   Deadline        Period
1       2       1               1
2       3       2               2
3       4       3               3

Scheduling occurs for 6 ms

0ms : Task 1 is running.
1ms : Task 1 is running.
2ms : Task 1 is running.
3ms : Task 1 is running.
4ms : Task 1 is running.
5ms : Task 1 is running.

Process returned 0 (0x0)   execution time : 26.084 s
Press any key to continue.
```