```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
void sort (int pid[], int d[], int b[], int pt[], int n)
{
  int temp = 0;
  for (int i = 0; i < n; i++)
        {
          for (int j = i; j < n; j++)
                {
                  if (d[j] < d[i])
                        {
                          temp = d[j];
                          d[j] = d[i];
                          d[i] = temp;
                          temp = pt[i];
                          pt[i] = pt[j];
                          pt[j] = temp;
                          temp = b[j];
                          b[j] = b[i];
                          b[i] = temp;
                          temp = pid[i];
                          pid[i] = pid[j];
                          pid[j] = temp;
                        }
                }
        }
}

int gcd (int a, int b)
{
  int r;
  while (b > 0)
        {
          r = a % b;
          a = b;
          b = r;
        }
  return a;
}

int lcm1 (int p[], int n)
{
  int lcm = p[0];
  for (int i = 1; i < n; i++)
        {
          lcm = (lcm * p[i]) / gcd (lcm, p[i]);
        }
  return lcm;
}
```

```
void main ()
{
  int n;
  printf ("Enter the number of processes:");
  scanf ("%d", &n);
  int pid[n], b[n], pt[n], d[n], rem[n];
  printf("Enter the PID,CPU burst time,deadline and time period");
  for(int i=0;i<n;i++)
  {
        scanf("%d%d%d",&pid[i],&b[i],&d[i],&pt[i]);
        rem[i]=b[i]
  }
  sort (pid, d, b, pt, n);

  int l = lcm1 (pt, n);

  printf ("\nEarliest Deadline Scheduling:\n");
  printf ("PID\t BT\tDeadline\tPeriod\n");
  for (int i = 0; i < n; i++)
        printf ("%d\t\t%d\t\t%d\t\t%d\n", pid[i], b[i], d[i], pt[i]);

  printf ("Scheduling occurs for %d ms\n\n", l);


  int time = 0, prev = 0, x = 0;
  int nextDeadlines[n];
  for (int i = 0; i < n; i++)
        {
          nextDeadlines[i] = d[i];
          rem[i] = b[i];
        }
  while (time < l)
        {
          for (int i = 0; i < n; i++)
                {
                  if (time % pt[i] == 0 && time != 0)
                        {
                          nextDeadlines[i] = time + d[i];
                          rem[i] = b[i];
                        }
                }
          int minDeadline = l + 1;
          int taskToExecute = -1;
          for (int i = 0; i < n; i++)
                {
                  if (rem[i] > 0 && nextDeadlines[i] < minDeadline)
                        {
                          minDeadline = nextDeadlines[i];
                          taskToExecute = i;
                        }
                }
          if (taskToExecute != -1)
                {
                  printf ("%dms : Task %d is running.\n", time,
pid[taskToExecute]);
                  rem[taskToExecute]--;
                }
          else
                {
                  printf ("%dms: CPU is idle.\n", time);
                }

          time++;
        }
}
```