Resolución de ecuación diferencial por Runge-Kutta de cuarto orden

1 Código

```
/* UNIVERSIDAD DE COSTA RICA
 * ESCUELA DE FISICA
 * CURSO: FS0733 TOPICOS DE METODOS MATEMATICOS DE LA FISICA
 * TEMA: FISICA COMPUTACIONAL Y PROGRAMACION EN C/C++
 * PROFESOR DAVID SOLANO SOLANO
 * ESTUDIANTE YEFRY LOPEZ NU EZ

 PROYECTO INTEGRACION DE ECUACIONES DIFERENCIALES DE 2DO ORDEN CON
    METODO DE RUNGE KUTTA */

 #include<stdio.h>
 #include<math.h> /* Use pow()*/
 #define ENE 250  /* Iterations */


 void kaa_values(void);
 double first_derivative( double t, double x, double u);
 double second_derivative( double t, double x, double u);


 static double deltat, deltax, deltau;
 static double x0, t0, u0, t, x, u, kaas[2][4];
 static double b,m;

 int main() {
  int i;
  double limsup;
  FILE *archivo;

  archivo = fopen("datac.csv", "w+");

  printf("\n\n\nINTEGRACION NUMERICA DE x'' = (b/m)(x')^2.\n\n\n");

  /*VALORES Constantes */
  b = 1.0;
  m = 1.0;
  /*VALORES INICIALES */

  t0 = 0;
  x0 = 0;
  u0 = 0;
```

```
41
42  printf("\n\n\nSe inicia desde el tiempo t0 = %lf .\nContinue
      Introduciendo el valor final tN: ", t0);
43  scanf("%lf", &limsup);
44
45  /* TAMANO DEL PASO */
46
47  deltat = (limsup-t0)/ENE;
48
49  /* INICIALIAZACION DEL BUCLE */
50
51  t = t0;
52  x = x0;
53  u = u0;
54
55  printf("\n\nSe resolver  el sistema con masa de %lf kg y una
      resistencia b de %lf\n", m,b);
56
57  /* IMPRIME LOS VALORES INICIALES EN LA PRIMERA LINEA DEL ARCHIVO
      */
58
59  fprintf(archivo, "%lf %lf %lf \n", t0, x0, u0);
60
61  /* BUCLE PROCESA EL CALCULO DE LOS t_i, y_i, y_i hasta i=N */
62  for(i=0;i<ENE;++i) {
63    kaa_values();
64    fprintf(archivo, "%lf %lf %lf \n", t0+(double)(i)*deltat, x+
      deltax, u+deltau);
65    t = t0+(double)(i)*deltat;
66    x = x+deltax;
67    u = u+deltau;
68  }
69
70  fclose(archivo);
71
72  return 0;
73 }
74
75 /*CALCULO DE LOS INDICES DE RUNGE-KUTTA K1,K2,K3,K4 */
76
77
78 void kaa_values( void ) {
79
80  /* Primer indice 0 calcula los k_0n = f1*/
81  kaas[0][0] = deltat*first_derivative(t, x, u);
82
83  /* Segundo indice 1 calcula los k_1n = f2*/
84  kaas[1][0] = deltat*second_derivative(t, x, u);
85
86
87  kaas[0][1] = deltat*first_derivative(t + deltat, x + 0.5*kaas
      [0][0], u + 0.5*kaas[1][0]);
88  kaas[1][1] = deltat*second_derivative(t + deltat, x + 0.5*kaas
      [0][0], u + 0.5*kaas[1][0]);
89
90  kaas[0][2] = deltat*first_derivative(t + deltat, x + 0.5*kaas
      [0][1], u + 0.5*kaas[1][1]);
```

```
 kaas[1][2] = deltat*second_derivative(t + deltat, x + 0.5*kaas
   [0][1], u + 0.5*kaas[1][1]);


 kaas[0][3] = deltat*first_derivative(t + deltat, x + kaas[0][2],
   u + kaas[1][2]);
 kaas[1][3] = deltat*second_derivative(t + deltat, x + kaas[0][2],
    u + kaas[1][2]);


 deltax = (1.0/6.0)*(kaas[0][0]+2.0*kaas[0][1]+2.0*kaas[0][2]+kaas
   [0][3]);
 deltau = (1.0/6.0)*(kaas[1][0]+2.0*kaas[1][1]+2.0*kaas[1][2]+kaas
   [1][3]);
}

/* ECUACION DIFERENCIAL x' = f1(t,x,u) = u */

double first_derivative( double t, double x, double u){
 return(u);
}

/* ECUACION DIFERENCIAL u' = f2(t,x,u) */

double second_derivative( double t, double x, double u){
 return((-b/m)*pow(u,2));
}
```

Listings