

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 double *dvector(long i, long j);          /* ベクトル領域の確保 */
5 void free_dvector(double *a, long i);     /* 領域の解放 */
6 void FUNC(double x, double *y, double *f); /* 関数の定義 */
7 /* ルンゲ・クッタ法(N 変数版) */
8 void rk4_m( double *y, double *f, int N,
9             double a, double b, int n, void (*F)(double, double *, double *) );
10
11 int main(void)
12 {
13     double *y, *f, a=0.0, b=3.1415926535897932;
14     int n, N=2; /* N 変数 */
15
16     y = dvector( 1, N ); f = dvector( 1, N ); /* 領域の確保 */
17     y[1]=1.0;y[2]=0.0; /* 初期値の設定 */
18
19     printf("分割数を入力してください--->");
20     scanf("%d",&n);
21
22     rk4_m( y, f, N, a, b, n, FUNC ); /* ルンゲ・クッタ法 */
23
24     return 0;
25 }
26
27 /* ルンゲ・クッタ法(N 変数版) */
28 void rk4_m( double *y, double *f, int N,
29             double a, double b, int n, void (*F)(double, double *, double *) )
30 {
31     double *k1, *k2, *k3, *k4, h, x, *tmp;
32     int i, j;
33
34     k1 = dvector( 1, N ); k2 = dvector( 1, N );
35     k3 = dvector( 1, N ); k4 = dvector( 1, N );
36     tmp = dvector( 1, N );
37
38     /* 初期値の設定・表示 */
39     h = (b-a)/n; /* 刻み幅 */
40     x = a;
41     printf("x=%f \t y1=%f \t y2=%f \n", x, y[1], y[2]);
42
43     /* ルンゲ・クッタ法(N 変数版) */
44     for ( i = 0 ; i < n ; i++)
45     { /* k1 の計算 */
46         (*FUNC)( x, y, f );
47         for( j = 1; j <= N; j++ ) k1[j] = f[j];
48         for( j = 1; j <= N; j++ ) tmp[j] = y[j] + h*k1[j]/2.0;
49         /* k2 の計算 */
50         (*FUNC)( x+h/2.0, tmp, f );
51         for( j = 1; j <= N; j++ ) k2[j] = f[j];
52         for( j = 1; j <= N; j++ ) tmp[j] = y[j] + h*k2[j]/2.0;
53         /* k3 の計算 */
54         (*FUNC)( x+h/2.0, tmp, f );
55         for( j = 1; j <= N; j++ ) k3[j] = f[j];
56         for( j = 1; j <= N; j++ ) tmp[j] = y[j] + h*k3[j];
57         /* k4 の計算 */
58         (*FUNC)( x+h, tmp, f );
59         for( j = 1; j <= N; j++ ) k4[j] = f[j];
60         for( j = 1; j <= N; j++ )
61             y[j] = y[j] + h/6.0*( k1[j] + 2.0*k2[j] + 2.0*k3[j] + k4[j] );
62         x += h;
63         printf("x=%f \t y1=%f \t y2=%f \n", x, y[1], y[2]);
64     }
65
66     /* 領域の解放 */
67     free_dvector( k1, 1 ); free_dvector( k2, 1 );
68     free_dvector( k3, 1 ); free_dvector( k4, 1 );
69     free_dvector( tmp, 1 );
70 }
71
72 /* 関数の定義 */
73 void FUNC(double x, double *y, double *f)
74 {
75     f[1] = y[2];
76     f[2] = -y[1];
77 }
78
79 double *dvector(long i, long j) /* a[i]~a[j] の領域を確保 */
80 {
81     double *a;
82
83     if ((a = malloc(((j - i + 1) * sizeof(double)))) == NULL)
84     {
85         printf("メモリが確保できません (from dvector) \n");
86         exit(1);
87     }
88
89     return (a - i);
90 }
91
92 void free_dvector(double *a, long i)
93 {
94     free((void *) (a + i)); /* (void *) 型へのキャストが必要 */
95 }

```