

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <math.h> /* コンパイル時に-lmオプションが必要 */
4
5 #define N 6 /* 要素の数 */
6
7 double *dvector(int i, int j); /* ベクトル領域の確保 */
8 void free_dvector(double *a, int i); /* 領域の解放 */
9
10 /* 1 ノルムの計算 a[m...n] */
11 double vector_norm1(double *a, int m, int n);
12 /* 2 ノルムの計算 a[m...n] */
13 double vector_norm2(double *a, int m, int n);
14 /* 比較関数 */
15 int double_comp(const void *s1, const void *s2);
16 /* 最大値ノルムの計算 a[m...n] */
17 double vector_norm_max(double *a, int m, int n);
18
19 int main(void)
20 {
21     int i; double *a;
22
23     /* ベクトルの定義, 配列 a の添字は 1~N */
24     a = dvector(1,N);
25     for(i = 1; i <= N; i++) a[i] = (double)(10 - i) / 20.0 * pow(-1.0,i);
26     for(i = 1; i <= N; i++) printf("a[%d]=%f\n", i, a[i]);
27
28     printf("ベクトル a の 1 ノルムは%f です\n", vector_norm1(a, 1, N));
29     printf("ベクトル a の 2 ノルムは%f です\n", vector_norm2(a, 1, N));
30     printf("ベクトル a の最大値ノルムは%f です\n", vector_norm_max(a, 1, N));
31     free_dvector(a,1); /* 領域の解放 */
32     return 0;
33 }
34
35 double *dvector(int i, int j) /* a[i]~a[j] の領域を確保 */
36 {
37     double *a;
38
39     if ((a = malloc(((j - i + 1) * sizeof(double)))) == NULL)
40     {
41         printf("メモリが確保できません (from dvector) \n");
42         exit(1);
43     }
44
45     return (a - i);
46 }
47
48 void free_dvector(double *a, int i)
49 {
50     free((void *) (a + i)); /* (void *) 型へのキャストが必要 */
51 }
52
53 /* 1 ノルムの計算 a[m...n] */
54 double vector_norm1(double *a, int m, int n)
55 {
56     int i;
57     double norm = 0.0;
58     for(i = m; i <= n; i++){
59         norm += fabs(a[i]);
60     }
61     return norm;
62 }
63
64 /* 2 ノルムの計算 a[m...n] */
65 double vector_norm2(double *a, int m, int n)
66 {
67     int i;
68     double norm = 0.0;
69     for(i = m; i <= n; i++){
70         norm += a[i] * a[i];
71     }
72     norm = sqrt(norm);
73     return norm;
74 }
75
76 /* 比較関数 (昇順) */
77 int double_comp(const void *s1, const void *s2)
78 {
79     const double a1 = *((double *)s1); /* (double *) へキャスト */
80     const double a2 = *((double *)s2); /* (double *) へキャスト */
81     if (a1 < a2)
82     {
83         return -1;
84     }
85     else if (a1 == a2)
86     {
87         return 0;
88     }
89     else
90     {
91         return 1;
92     }
93 }
94
95 /* 最大値ノルムの計算 a[m...n] */
96 double vector_norm_max(double *a, int m, int n)
97 {
98     int i, tmp;
99     tmp = n - m + 1; /* 全要素数の計算 */
100     for(i = m; i <= n; i++) a[i] = fabs(a[i]);
101     /* 並べ換え:先頭アドレスが m だけずれていることに注意 */
102     qsort(a + m, tmp, sizeof(a[0]), double_comp);
103     return a[n];
104 }

```