```
1       /*
          2.
          Соседями элемента A [i, j] в матрице назовем элементы A [k, l] с
            i - 1 <= k <= i + 1,
            j - 1 <= l <= j + 1,
            (k, l) != (i, j).

          Операция сглаживания матрицы дает новую матрицу того же размера,
          каждый элемент которой получается как среднее арифметическое
          имеющихся соседей соответствующего элемента исходной матрицы.

          Построить результат сглаживания заданной вещественной матрицы
          размером 10 на 10. Выполнить задание, используя динамическое
          выделение памяти.

          MSP,
          Home,
          18.10.09, 20:44-21:48;
          Work,
          19.10.09, 11:32-14:16.
        */

19      # include <stdio.h>
20      # include <conio.h>
21      # include <stdlib.h>
22      # include <time.h>

23      int main ()
24      {
25        char
26          i, j, // индексы матрицы
27          N = 10; // размер матрицы 10 x 10

28        float
29          K, // число соседей
30          S, // сумма соседей
31          * R, // адрес "сглаженной" матрицы
32          * A; // адрес исходной матрицы

33        clrscr ();

34        A = (float *) malloc (N * N * sizeof (float));
35        R = (float *) malloc (N * N * sizeof (float));

36        // включить генератор случайных чисел
37        randomize ();

38        // заполним матрицу случайным образом
39        K = 1;
40        for (i = 0; i < N; ++ i)
```

```
41            for (j = 0; j < N; ++ j)
42              //* (A + i * N + j) = K ++;
43              * (A + i * N + j) =
44                (random (100) - 50) / (random (10) + 4.2);

45          // печать матрицы
46          printf ("Исходная матрица:\n");
47          for (i = 0; i < N; ++ i)
48          {
49            for (j = 0; j < N; ++ j)
50              printf ("%6.1f ", * (A + i * N + j));
51            printf ("\n");
52          }

53          // произведем операцию сглаживания матрицы
54          // заполним матрицу случайным образом
55          for (i = 0; i < N; ++ i)
56            for (j = 0; j < N; ++ j)
57            {
58              // просмотрим соседей по часовой стрелке
59              K = 0;
60              S = 0.0;
61              // 1. сосед сверху, если он есть
62              if (i - 1 >= 0)
63                S += * (A + (i - 1) * N + j), K ++;

64              // 2. сосед в правом верхнем углу квадрата соседей
65              if (
66                   i - 1 >= 0
67                   &&
68                   j + 1 <= N - 1
69                 )
70                S += * (A + (i - 1) * N + j + 1), K ++;

71              // 3. сосед справа
72              if (j + 1 <= N - 1)
73                S += * (A + i * N + j + 1), K ++;

74              // 4. сосед в правом нижнем углу квадрата соседей
75              if (
76                   i + 1 <= N - 1
77                   &&
78                   j + 1 <= N - 1
79                 )
80                S += * (A + (i + 1) * N + j + 1), K ++;

81              // 5. сосед внизу
82              if (i + 1 <= N - 1)
83                S += * (A + (i + 1) * N + j), K ++;

84              // 6. сосед в левом нижнем углу квадрата соседей
```

```
85              if (
86                   i + 1 <= N - 1
87                   &&
88                   j - 1 >= 0
89                 )
90                S += * (A + (i + 1) * N + j - 1), K ++;

91              // 7. сосед слева
92              if (j - 1 >= 0)
93                S += * (A + i * N + j - 1), K ++;

94              // 8. сосед в левом верхнем углу квадрата соседей
95              if (
96                   i - 1 >= 0
97                   &&
98                   j - 1 >= 0
99                 )
100               S += * (A + (i - 1) * N + j - 1), K ++;

101             * (R + i * N + j) = S / K;
102           }

103         // печать матрицы-результата
104         printf ("\nМатрица-результат:\n");
105         for (i = 0; i < N; ++ i)
106         {
107           for (j = 0; j < N; ++ j)
108             printf ("%6.1f ", * (R + i * N + j));
109           printf ("\n");
110         }

111         // вернем занятую память OS
112         free (A);
113         free (R);

114         getch ();

115         return 0;
116       }
```

Listing данной задачи опубликован в сети Internet по адресу
http://www.Best-Listing.ru/color-2-task-139.html

Sergey Mitrofanov, 26.08.14, 21:36
E-mail: infostar @ mail.ru