**Лабораторная работа №6**

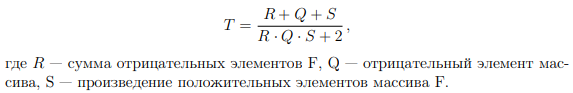
“Динамические массивы”

Задание 6.1

**Постановка задачи:**

Создать динамический одномерный массив целых чисел F размерности 12 и заполнить его положительными и отрицательными числами. T = R + Q + S R · Q · S + 2 , где R — сумма отрицательных элементов F, Q — отрицательный элемент массива, S — произведение положительных элементов массива F.

**Математическая модель:**

****

**Список идентификаторов:**

|  |  |  |
| --- | --- | --- |
| **Идентификатор** | **Семантика** | **Тип** |
| \*f | Указатель на массив | int |
| n | Количество элементов массива | int |
| t,r,q,s | Переменные, значение которых указано в математической модели | float |
| i | Переменная, позволяющая управлять элементами массива | int |

**Код программы:**

#include <malloc.h>

#include <math.h>

#include <stdio.h>

#include <stdlib.h>

int main(int argc, char const \*argv[]) {

int \*f; // указатель на массив

int i, n = 12;

float t, r = 0, q, s = 1;

// Выделение памяти

f = (int \*)malloc(n \* sizeof(int));

// Заполнение массива случайными числами

for (i = 0; i < n; i++) {

f[i] = rand() % 100 - 50;

if (f[i] <= 0) {

r += f[i];

q = f[i];

} else {

s \*= f[i];

}

}

// Вывод элементов массива

for (i = 0; i < n; i++)

printf("%d ", f[i]);

printf("\n");

t = (r + q + s) / (r \* q \* s + 2);

printf("t=%f\n", t);

free(f);

return 0;

}

**Результат выполненной работы:**

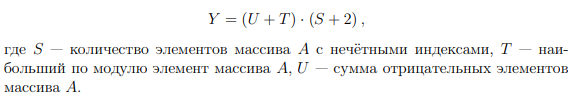
****

**Задание 6.2**

**Постановка задачи:**

Дан одномерный массив целых чисел A размера 12. Вычислить: Y = (U + T) · (S + 2), где S — количество элементов массива A с нечётными индексами, T — наибольший по модулю элемент массива A, U — сумма отрицательных элементов массива A.

**Математическая модель:**

****

При вычислении значения выражения надо будет сохранить знак переменной T.

**Список идентификаторов:**

|  |  |  |
| --- | --- | --- |
| **Идентификатор** | **Семантика** | **Тип** |
| \*a | Указатель на массив | int |
| n | Количество элементов массива | int |
| u,t,s,y | Переменные, значение которых указано в математической модели | int |
| k | Переменная, в которой сохраняется знак t | int |
| i | Переменная, позволяющая управлять элементами массива | int |

**Код программы:**

#include <malloc.h>

#include <math.h>

#include <stdio.h>

#include <stdlib.h>

int main(int argc, char const \*argv[]) {

int \*a; // указатель на массив

int i, n = 12, s = 0, t = 0, u = 0, y, k = 0;

// Выделение памяти

a = (int \*)malloc(n \* sizeof(int));

// Заполнение массива случайными числами

for (i = 0; i < n; i++) {

a[i] = rand() % 100 - 60;

if (i % 2 != 0)

s++;

if (abs(a[i]) > t) {

//Переменная k позволяет сохранить "знак" наибольшего по модулю числа

k = (a[i] < 0) ? 1 : 0;

t = abs(a[i]);

}

if (a[i] < 0)

u += a[i];

}

for (i = 0; i < n; i++)

printf("%d ", a[i]);

printf("\n");

if (k != 0)

t = -t;

printf("\n");

printf("s=%d\n", s);

printf("t=%d\n", t);

printf("u=%d\n", u);

printf("\n");

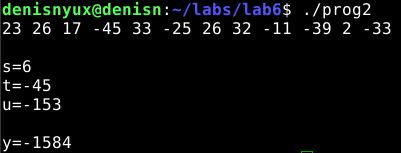
y = (u + t) \* (s + 2);

printf("y=%d\n", y);

return 0;

}

**Результат выполненной работы:**

****

**Задание 6.3**

**Постановка задачи:**

Напишите программу для вычисления пересечения двух конечных множеств (наборов) A и B целых чисел одинакового размера с использованием динамических массивов. В качестве множества A можно взять первые 12 чисел ряда Фибоначчи: 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144. В качестве множества B можно взять первые 12 чисел последовательности Падована: 1, 1, 1, 2, 2, 3, 4, 5, 7, 9, 12, 16. Дублирующиеся значения можно исключать.

**Математическая модель:**

Числа Фибоначчи можно представить в виде рекуррентного соотношения, у которого a(0)=1; a(1)=1; a(n)=a(n-1)+a(n-2);

Последовательность Падована можно также представить в виде соотношения b(0)=1; b(1)=1; b(2)=1; b(n)=b(n-2)+b(n-3);

**Список идентификаторов:**

|  |  |  |
| --- | --- | --- |
| **Идентификатор** | **Семантика** | **Тип** |
| \*a | Указатель на массив (Фибоначчи) | int |
| \*b | Указатель на массив  (Падован) | int |
| \*с | Указатель на массив, содержащий общие элементы последовательностей | int |
| n | Количество элементов массивов a и b | int |
| k | Переменная, позволяющая вычислить количество совпадающих элементов и количество элементов массива c | int |
| i | Переменная, позволяющая управлять элементами массива | int |

**Код программы:**

#include <malloc.h>

#include <math.h>

#include <stdio.h>

#include <stdlib.h>

int main(int argc, char const \*argv[]) {

/\*Объявление динамических массивов размерностью n\*/

int \*a, \*b, \*c;

int n = 12, i, k = 0, j;

;

a = (int \*)malloc(n \* sizeof(int));

b = (int \*)malloc(n \* sizeof(int));

c = (int \*)malloc(n \* sizeof(int));

/\*Заполнение массивов элементами последовательности\*/

a[0] = 1;

a[1] = 1;

b[0] = 1;

b[1] = 1;

b[2] = 1;

for (i = 2; i < n; i++)

a[i] = a[i - 1] + a[i - 2];

for (i = 3; i < n; i++)

b[i] = b[i - 2] + b[i - 3];

printf("\nПервое множество:\n");

for (i = 0; i < n; i++)

printf("%d ", a[i]);

printf("\nВторое множество:\n");

for (i = 0; i < n; i++)

printf("%d ", b[i]);

printf("\n");

/\*Находим совпадающие\*/

for (i = 0; i < n; i++) {

for (j = 0; j < n; j++) {

if (a[i] == b[j] & c[k - 1] != b[j]) {

c[k] = a[i];

k++;

}

}

}

printf("\nПересечение двух множеств:\n");

for (i = 0; i < k; i++)

printf("%d ", c[i]);

printf("\n");

free(a);

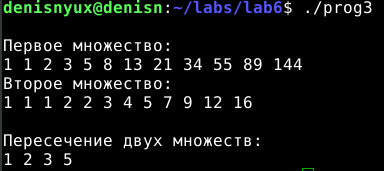
free(b);

free(c);

return 0;

}

**Результат выполненной работы:**

****

**Задание 6.4**

**Постановка задачи:**

Выделить динамически память под некоторую матрицу A размерности M × N и заполнить её произвольными числами. Сократить правильно размер этой матрицы, удалив из неё одну выбранную строку, освободив от неё также и память, используя указатели. После удаления строки в матрице AM×N должна быть возможность обхода всех элементов «новой» матрицы AM−1×N таким же способом, что и изначальной матрицы. Распечатать (используя циклы) матрицу до удаления строки и после удаления.

**Математическая модель:**

Удалением строки будем считать “обнуление” всех элементов строки.

**Список идентификаторов:**

|  |  |  |
| --- | --- | --- |
| **Идентификатор** | **Семантика** | **Тип** |
| \*\*a | Указатель на указатель на строку элементов | int |
| n, m | Количество строк и столбцов массива | int |
| k | Номер удаляемой строки | int |
| i,j | Переменные, позволяющая управлять элементами | int |

**Код программы:**

#include <malloc.h>

#include <math.h>

#include <stdio.h>

#include <stdlib.h>

int main(int argc, char const \*argv[]) {

int \*\*a;

int i, j, n, m, k;

printf("Введите количество строк и столбцов: ");

scanf("%d %d", &m, &n);

/\*Выделение памяти на строки\*/

a = (int \*\*)malloc(m \* sizeof(int \*));

for (i = 0; i < m; i++) {

/\*Выделение памяти на столбцы и заполнение массива случайными числами\*/

a[i] = (int \*)malloc(n \* sizeof(int));

for (j = 0; j < n; j++) {

a[i][j] = rand() % 50 - 25;

}

}

/\*Вывод массива\*/

for (i = 0; i < m; i++) {

for (j = 0; j < n; j++) {

printf("%d ", a[i][j]);

}

printf("\n");

}

printf("Введите номер строки, которую необходимо удалить:");

scanf("%d", &k);

for (i = 0; i < n; i++) {

a[k - 1][i] = 0;

}

for (i = 0; i < m; i++) {

for (j = 0; j < n; j++) {

printf("%d ", a[i][j]);

}

printf("\n");

}

/\*Очистка памяти\*/

for (i = 0; i < m; i++)

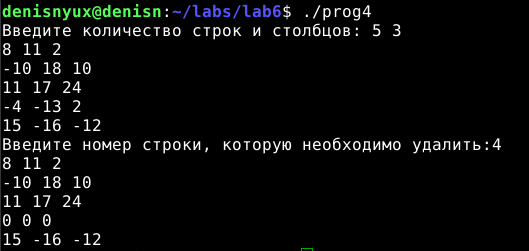
free(a[i]);

free(a);

return 0;

}

**Результат выполненной работы:**

****

**Задание 6.5**

**Постановка задачи:**

Написать программу, которая вычисляет некоторый вектор b, как результат умножения некоторой матрицы M на вектор a:

b = M × a .

**Математическая модель:**

При умножении матриц стоит обратить внимание на их размерность. Если заданная матрица M имеет n строк и m столбцов, то вектор А, на который она умножается должен иметь размерность na=n и m=1. Аналогичную размерность должен иметь полученный в итоге вектор B.

**Список идентификаторов:**

|  |  |  |
| --- | --- | --- |
| **Идентификатор** | **Семантика** | **Тип** |
| \*\*mas | Указатель на указатель на строку элементов в массиве mas | int |
| \*\*a | Указатель на указатель на строку элементов в массиве a | int |
| \*\*b | Указатель на указатель на строку элементов в массиве b | int |
| n, m | Количество строк и столбцов массива mas | int |
| na, ma | Количество строк и столбцов массива a | int |
| nb, mb | Количество строк и столбцов массива b | int |
| k | Вспомогательная переменная при умножении матриц | int |
| i,j | Переменные, позволяющая управлять элементами | int |

**Код программы:**

#include <malloc.h>

#include <math.h>

#include <stdio.h>

#include <stdlib.h>

int main(int argc, char const \*argv[]) {

int \*\*mas;

int i, j, n, m;

printf("Введите количество строк и столбцов матрицы M: ");

scanf("%d %d", &n, &m);

/\*Выделение памяти на строки\*/

mas = (int \*\*)malloc(n \* sizeof(int \*));

for (i = 0; i < n; i++) {

/\*Выделение памяти на столбцы и заполнение матрицы случайными числами\*/

mas[i] = (int \*)malloc(m \* sizeof(int));

for (j = 0; j < m; j++) {

mas[i][j] = rand() % 50 - 25;

}

}

int \*\*a;

int na = n, ma = 1;

/\*Выделение памяти на строки\*/

a = (int \*\*)malloc(na \* sizeof(int \*));

for (i = 0; i < na; i++) {

/\*Выделение памяти на столбцы и заполнение матрицы случайными числами\*/

a[i] = (int \*)malloc(ma \* sizeof(int));

for (j = 0; j < ma; j++) {

a[i][j] = rand() % 50 - 25;

}

}

printf("\nМатрица М:\n");

/\*Вывод массива M\*/

for (i = 0; i < n; i++) {

for (j = 0; j < m; j++) {

printf("%d ", mas[i][j]);

}

printf("\n");

}

printf("\n");

/\*Вывод вектора a\*/

printf("Вектор a:\n");

for (i = 0; i < na; i++) {

for (j = 0; j < ma; j++) {

printf("%d\n", a[i][j]);

}

}

/\*Умножение матрицы на вектор\*/

int \*\*b;

int nb = n, mb = ma, k;

b = (int \*\*)malloc(nb \* sizeof(int \*));

for (i = 0; i < nb; i++) {

b[i] = (int \*)malloc(mb \* sizeof(int));

for (j = 0; j < mb; j++) {

b[i][j] = 0;

for (k = 0; k < m; k++) {

b[i][j] += mas[i][j] \* a[i][j];

}

}

}

/\*Вывод вектора b\*/

printf("\nВектор b:\n");

for (i = 0; i < nb; i++) {

for (j = 0; j < mb; j++) {

printf("%d\n", b[i][j]);

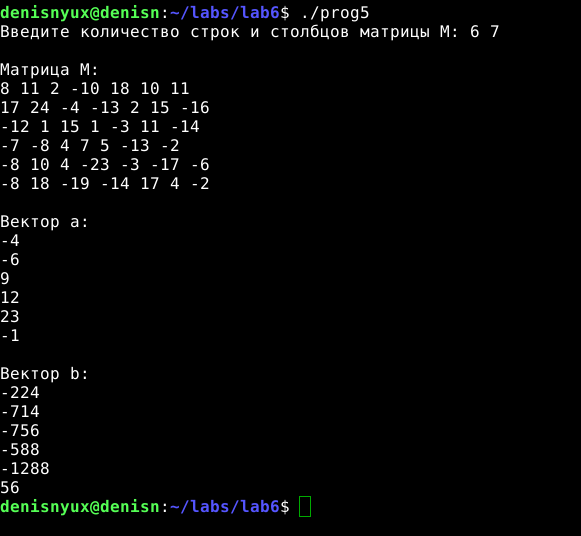
}

}

return 0;

}

**Результат выполненной работы:**

****

**Задание 6.6**

**Постановка задачи:**

Напишите программу, в которой создаётся квадратная матрица, заполненная нулями и единицами. Единичные значения у тех элементов, для которых сумма индексов является нечётным числом. Нулевые значения у тех элементов, для которых сумма индексов является чётным числом.

**Математическая модель:**

Квадратная матрица имеет одинаковое количество столбцов и строк. Сумма индексов - это i+j, где i - номер строки, j - номер столбца.

**Список идентификаторов:**

|  |  |  |
| --- | --- | --- |
| **Идентификатор** | **Семантика** | **Тип** |
| \*\*mas | Указатель на указатель на строку элементов | int |
| n | Количество строк и столбцов массива | int |
| i,j | Переменные, позволяющая управлять элементами | int |

**Код программы:**

#include <malloc.h>

#include <math.h>

#include <stdio.h>

#include <stdlib.h>

int main(int argc, char const \*argv[]) {

int \*\*mas;

int i, j, n;

printf("Введите количество строк и столбцов квадратной матрицы:");

scanf("%d", &n);

/\*Выделение памяти на строки\*/

mas = (int \*\*)malloc(n \* sizeof(int \*));

for (i = 0; i < n; i++) {

/\*Выделение памяти на столбцы и заполнение матрицы случайными числами\*/

mas[i] = (int \*)malloc(n \* sizeof(int));

for (j = 0; j < n; j++) {

mas[i][j] = ((i + j) % 2 == 0) ? 0 : 1;

}

}

/\*Вывод массива\*/

printf("\n");

for (i = 0; i < n; i++) {

for (j = 0; j < n; j++) {

printf("%d ", mas[i][j]);

}

printf("\n");

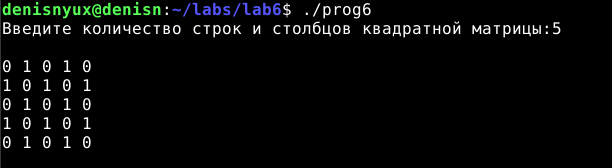
}

printf("\n");

return 0;

}

**Результат выполненной работы:**

****