**Лабораторная работа 8.**

**Функции.**

1. Даны три одномерных массива вещественных чисел A[1 . . . 6],

A[1 . . . 8] и C[1 . . . 7]. Найти общую сумму положительных элементов в массивах. Нахождение суммы элементов в массиве оформить функцией. Список идентификаторов:

Математическая модель: sum = sum + x[i], x[i]>0.

|  |  |  |
| --- | --- | --- |
| Имя | Значение | Тип |
| a | Массив из 6 элементов | int |
| b | Массив из 8 элементов | int |
| c | Массив из 7 элементов | int |
| sum | Сумма не отрицательных элементов | int |
| i | Счетчик цикла, индекс массива | int |
| \*arr | Массив передаваемый в функцию | int |
| num | Количество элементов массива передаваемое в функцию | int |

Код программы:

#include <stdio.h>

int get\_unsum(int \*arr, int num){

int i, sum=0;

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

if(arr[i]>0)

sum += arr[i];

return sum;

}

int main(){

int a[6] = {1, 5, -4, 2, -6, 6},

b[8] = {0, 0, -3, -1, 10, 3, 2, 0},

c[7] = {4, -5, -8, 11, -7, -6, 3};

printf("%d\n",get\_unsum(a, 6));

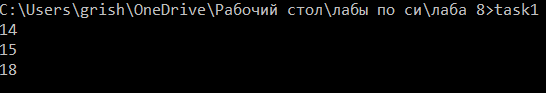
printf("%d\n",get\_unsum(b, 8));

printf("%d\n",get\_unsum(c, 7));

return 0;

}

Результат выполненной работы:



2. Даны два одномерных массива целых чисел A[1 . . . 8] и B[1 . . . 8]. Найти сумму их максимальных элементов. Для нахождения максимального элемента в массиве использовать функцию.

Математическая модель: Xmax ≥ x[i], i = 0..7; sum = Amax + Bmax.

Список идентификаторов:

|  |  |  |
| --- | --- | --- |
| Имя | Значение | Тип |
| a | Массив А | int |
| b | Массив В | int |
| sum | Сумма максимальных элементов | int |
| \*arr | Передаваемый в с | int |
| max | Максимальный элемент массива | int |
| i | Счетчик цикла, индекс массива | int |

Код программы:

#include <stdio.h>

int num\_max(int \*arr){

int i, max;

num = arr[0];

for(i = 0; i<8; i++)

if(arr[i]>max)

max = arr[i];

return max;

}

int main(){

int sum=0;

int a[8] = {1, -5, 4, 2, -4, -5, -3, 6},

b[8] = {0, -8, 11, -7, -6, 3, 2, 0};

sum += num\_max(a);

sum += num\_max(b);

printf("%d\n", sum);

return 0;

}

Результат выполненной работы:



3. Даны три одномерных массива вещественных чисел A[1 . . . 6], V [1 . . . 8] и C[1 . . . 7]. Найти среднее геометрическое значение положительных элементов для каждого. Вычисление среднего геометрического оформить в виде функции.

Математическая модель: .

Список идентификаторов:

|  |  |  |
| --- | --- | --- |
| Имя | Значение | Тип |
| a | Массив из 6 элементов | int |
| v | Массив из 8 элементов | int |
| c | Массив из 7 элементов | int |
| \*arr | Передаваемый в функцию массив | int |
| num | Степень корня | float |
| i | Счетчик цикла, индекс массива | int |
| avg | Среднее геометрическое | float |

Код программы:

#include <stdio.h>

#include <math.h>

float get\_geomavg(int \*arr, float num){

int i;

float avg = 1;

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

avg =avg \* arr[i];

num = 1/num;

avg = pow(avg, num);

return avg;

}

int main(){

int a[6] = {1, 5, -4, 2, -6, 6},

v[8] = {0, 0, -3, -1, 10, 3, 2, 0},

c[7] = {4, -5, -8, 11, -7, -6, 3};

printf("%g\n",get\_geomavg(a, 6));

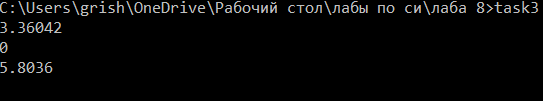
printf("%g\n",get\_geomavg(v, 8));

printf("%g\n",get\_geomavg(c, 7));

return 0;

}

Результат выполненной работы:



4. Даны две матрицы целых чисел S[1 . . . 3, 0 . . . 2], K[1 . . . 3, 0 . . . 2], в каждой из которых имеется по два одинаковых числа. Распечатать их значения.

Список идентификаторов:

|  |  |  |
| --- | --- | --- |
| Имя | Значение | Тип |
| s | Матрица S | int |
| k | Матрица K | int |
| arr | Передаваемая в функцию матрица | int |
| i | счетчик цикла, индекс строки первого массива | int |
| j | счетчик цикла, индекс столбца первого массива | int |
| m | счетчик цикла, индекс строки второго массива | int |
| n | счетчик цикла, индекс столбца второго массива | int |

Код программы:

#include <stdio.h>

void two\_num(int (\*arr)[3]){

int i, j, m, n;

for(i=0; i < 3; i++){

for(j=0; j < 3; j++){

n = j;

for(m=i; m<3; m++){

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

if((arr[i][j]==arr[m][n]) && !(i == m && j == n))

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

n = 0;

}

}

}

}

int main(){

int s[3][3] = {{1, -5, 1},

{3, -8, -5},

{-3, 4, 5}},

k[3][3] = {{0, 7, 1},

{7, 9, -6},

{-3, 2, 0}};

two\_num(s);

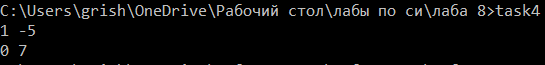
printf("\n");

two\_num(k);

return 0;

}

Результат выполненной работы:



5. Дана матрица целых чисел D[1 . . . 6, 1 . . . 5]. Найти наименьшую из сумм неотрицательных элементов строк матрицы. Для вычисления суммы использовать подпрограмму (функцию).

Математическая модель: X = min(, i = 0..m).

Список идентификаторов:

|  |  |  |
| --- | --- | --- |
| Имя | Значение | Тип |
| arr | Матрица 6x5 | int |
| i | счетчик цикла, индекс строки массива | int |
| j | счетчик цикла, индекс столбца массива | int |
| min | Наименьшая из сумм строк матрицы. | int |
| gl | Сумма строки матрицы | int |
| parr | Передаваемый в функцию массив | int |
| s | Сумма строки матрицы | int |

Код программы:

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

int sum(int\*\* parr, int i){

int s =0;

for(int j=0; j <5; j++)

if(parr[i][j]>0)

s += parr[i][j];

return s;

}

int main(int argc, char\*\* argv){

int i, j, min = 51, gl;

int\*\* arr = (int\*\*)malloc(6\*sizeof(int\*));

for(i=0; i<6; i++)

arr[i] = (int\*)malloc(5\*sizeof(int));

srand(time(NULL));

for(i=0; i<6; i++){

for(j=0; j<5; j++){

arr[i][j] = rand()%10;

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

}

printf("\n");

}

for(i=0; i<6; i++){

gl = sum(arr, i);

if (min > gl)

min = gl;

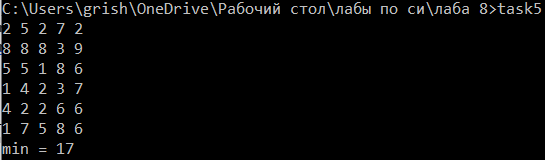
}

printf("min = %d", min);

return 0;

}

Результат выполненной работы:



6. Дана матрица целых чисел D[1 . . . 3, 1 . . . 5]. Используя функцию, найти среднее геометрическое значение для каждого столбца матрицы.

Математическая модель: .

Список идентификаторов:

|  |  |  |
| --- | --- | --- |
| Имя | Значение | Тип |
| d | Матрица D | int |
| arr | Матрица, переданная в функцию | int |
| i | счетчик цикла, индекс строки массива | int |
| j | счетчик цикла, индекс столбца массива | int |
| mul | Произведение элементов в столбце | float |
| avg | Среднее геометрическое. | float |

Код программы:

#include <stdio.h>

#include <math.h>

void colavg(int (\*arr)[5]){

int i, j;

float mul=1, avg;

for(i=0; i < 5; i++){

for(j=0; j < 3; j++){

mul \*= arr[j][i];

}

avg = pow(mul, (float)1/3);

printf("%g ", avg);

mul = 1;

}

}

int main(){

int d[3][5] = {{1, 5, 1, 4, 3},

{3, 8, 5, 6, 9},

{3, 4, 5, 1, 2}};

colavg(d);

return 0;

}

Результат выполненной работы:



7. Дана матрица целых чисел F[1 . . . 4, 1 . . . 5]. Найти наименьшие значения элементов в каждой из строк матрицы с помощью функции. Математическая модель: N - угольник.

Математическая модель: X[i]min ≤ x[i][j], j = 0..5.

Список идентификаторов:

|  |  |  |
| --- | --- | --- |
| Имя | Значение | Тип |
| f | Матрица F | int |
| arr | Матрица, переданная в функцию | int |
| i | счетчик цикла, индекс строки массива | int |
| str | Номер строки | int |
| j | счетчик цикла, индекс столбца массива | int |
| min | Минимальный элемент строки. | int |

Код программы:

#include <stdio.h>

#include <math.h>

int min\_instr(int (\*arr)[5], int str){

int j, min = arr[str][0];

for (j = 1; j < 5; j++)

{

if (min> arr[str][j])

min = arr[str][j];

}

return min;

}

int main(){

int i;

int f[4][5] = { {1, 5, -4, 2, 6},

{0, 0, -3, 2, 0},

{4, -8, -7, -6, 3},

{1, -9, 2, 6, 3}};

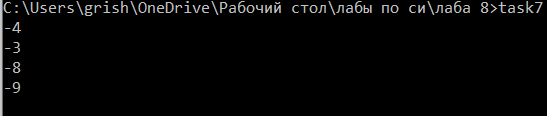
for (i = 0; i < 4; i++)

printf("%d\n", min\_instr(f, i));

return 0;

}

Результат выполненной работы:



8. Ввести число N и определить, простое оно или нет. Использовать функцию, которая отвечает на этот вопрос.

Математическая модель: простое число делится только на себя и на единицу.

Список идентификаторов:

|  |  |  |
| --- | --- | --- |
| Имя | Значение | Тип |
| n | Проверяемое число | int |
| i | Счетчик цикла, делитель числа | int |

Код программы:

#include <stdio.h>

int simp(int n){

int i;

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

if (n % i == 0){

printf("NO");

return 0;

}

}

printf("YES");

return 0;

}

int main(){

int n;

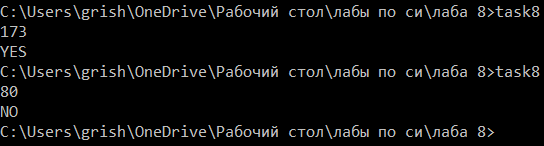
scanf("%d",&n);

simp(n);

return 0;

}

Результат выполненной работы:



9. Реализовать выделение и освобождение памяти для произвольной матрицы с помощью функций malloc и free в своих собственных отдельных функциях. С помощью этих новых функций реализовать код в функции main, выполняющий выделение и освобождение памяти под матрицу с заполнением значениями элементов матрицы и распечаткой этой матрицы на экране терминала. Список идентификаторов:

|  |  |  |
| --- | --- | --- |
| Имя | Значение | Тип |
| matrix | Созданная матрица | int |
| i | Счетчик цикла, индекс строки матрицы | int |
| j | Счетчик цикла, индекс столбца матрицы |  |
| size\_n | Количество строк матрицы |  |
| size\_m | Количество столбцов матрицы |  |

Код программы:

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

int\*\* create\_matrix(int \*\*matrix, int size\_n, int size\_m)

{

matrix = (int\*\*)malloc(size\_n \* sizeof(int));

for (int i = 0; i < size\_n; i++)

matrix[i] = (int\*)malloc(size\_m\* sizeof(int));

return matrix;

}

void set\_matrix(int\*\* matrix, int size\_n, int size\_m)

{

for (int i = 0; i < size\_n; i++)

for (int j = 0; j < size\_m; j++)

matrix[i][j] = rand() % 10;

}

void show\_matrix(int\*\* matrix, int size\_n, int size\_m)

{

for (int i = 0; i < size\_n; i++) {

for (int j = 0; j < size\_m; j++) {

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

}

printf("\n");

}

}

void free\_mem(int\*\* matrix, int size\_n)

{

for (int i = 0; i < size\_n; i++)

free(matrix[i]);

free(matrix);

}

int main()

{

srand(time(NULL));

int size\_n, size\_m;

printf("Enter matrix size: ");

scanf("%d%d", &size\_n, &size\_m);

int\*\* matrix;

matrix = create\_matrix(matrix, size\_n, size\_m);

set\_matrix(matrix, size\_n, size\_m);

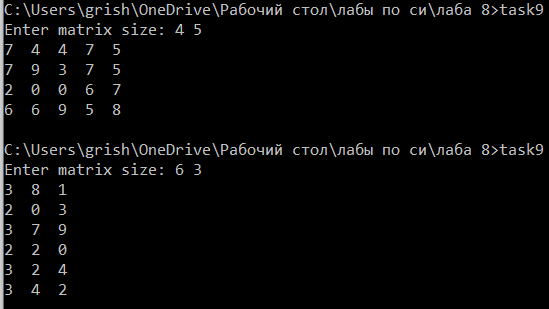
show\_matrix(matrix, size\_n, size\_m);

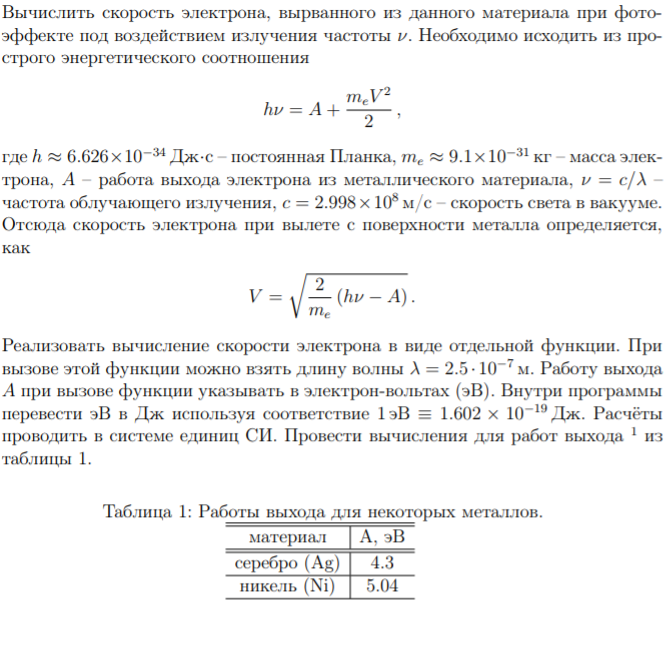
free\_mem(matrix, size\_n);

return 0;

}

Результат выполненной работы:



10. 

Математическая модель: .

Список идентификаторов:

|  |  |  |
| --- | --- | --- |
| Имя | Значение | Тип |
| \_H | Постоянная Планка | float |
| \_Me | Масса электрона | float |
| \_C | Скорость света в вакууме | float |
| \_Lm | Длина волны | float |
| a | Работа выхода |  |

Код программы:

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#define \_H 6.626e-34

#define \_Me 9.1e-31

#define \_C 2.998e8

#define \_Lm 2.5e-7

float func1(float a)

{

float hv = (float)(\_H \* \_C) / \_Lm; // vi = c/lm

if (hv < a)

return 0;

else {

float v = sqrt((2 / \_Me) \* (hv - a));

return v;

}

}

int main()

{

float a;

printf("Enter the number A: ");

scanf("%f", &a);

a \*= 1.602e-19;

printf("V = %g\n", func1(a));

return 0;

}

Результат выполненной работы:

