**Лабораторная работа №6**

**Тема лабораторной работы**

Указатели, арифметика указателей. Введение в функции.

**Задача 1**

**Постановка задачи**

Внутри функции main(void) определите указатель double \*\*pointer = NULL;. В оперативной памяти создайте конструкцию, показанную на рисунке 1.

**Список идентификаторов**

| **Имя** | **Смысл** | **Тип** |
| --- | --- | --- |
| \*\*pointer | указатель | double |

**Код программы**

#include <stdio.h>

#include <stdlib.h>

int main() {

double \*\*pointer = NULL;

pointer = (double \*\*) malloc(sizeof(double \*));

pointer[0] = (double \*) malloc(sizeof(double));

\*pointer[0] = 2;

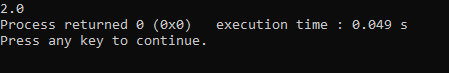
printf("%.1f", \*pointer[0]);

free(pointer);

return 0;

}

**Результат выполненной работы**

****

**Задача 2**

**Постановка задачи**

Напишите программу, которая складывает два числа с использованием указателей на эти числа.

**Математическая модель**

x+y

**Список идентификаторов**

| **Имя** | **Смысл** | **Тип** |
| --- | --- | --- |
| x | вводимое с клавиатуры число | int |
| y | вводимое с клавиатуры число | int |
| \*px | указатель | int |
| \*py | указатель | int |

**Код программы**

#include <stdio.h>

#include <stdlib.h>

int main(void) {

int x, y;

scanf("%d\n%d", &x, &y);

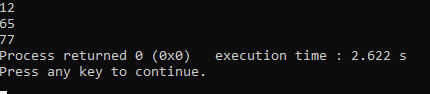
int \*px = &x, \*py = &y;

printf("%d", \*px + \*py);

return 0;

}

**Результат выполненной работы**



**Задача 3**

**Постановка задачи**

Напишите программу, которая находить максимальное число из двух чисел, используя указатели на эти числа

**Математическая модель**

**Список идентификаторов**

| **Имя** | **Смысл** | **Тип** |
| --- | --- | --- |
| x | вводимое с клавиатуры число | int |
| y | вводимое с клавиатуры число | int |
| \*px | указатель | int |
| \*py | указатель | int |

**Код программы**

#include <stdio.h>

#include <stdlib.h>

int main(void) {

int x, y;

scanf("%d\n%d", &x, &y);

int \*px = &x, \*py = &y;

if (\*px > \*py)

printf("%d", \*px);

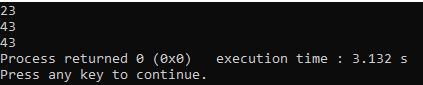
else

printf("%d", \*py);

return 0;

}

**Результат выполненной работы**



**Задача 4**

**Постановка задачи**

Напишите программу, которая создаёт одномерный динамический массив из чисел с плавающей точкой двойной точности, заполняет его значениями с клавиатуры и распечатывает все элементы этого массива, используя арифметику указателей (оператор +), а не обычный оператор доступа к элементу массива — [].

**Математическая модель**

A[i] = A+i

**Список идентификаторов**

| **Имя** | **Смысл** | **Тип** |
| --- | --- | --- |
| \*A | указатель | double |
| i | параметр цикла | int |
| n | размер массива | int |

**Код программы**

#include <stdio.h>

#include <stdlib.h>

int main(void) {

int n;

double \*A;

printf("Quantity of numbers:\n");

scanf("%d", &n);

A = (double \*)malloc(n \* sizeof(double));

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

printf("A[%d] = ", i);

scanf("%lf", (A + i));

}

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

printf("A[%d] = %lf\n", i, \*(A + i));

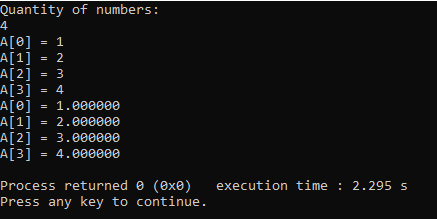
}

free(A);

return 0;

}

**Результат выполненной работы**



**Задача 5**

**Постановка задачи**

Вычислить факториал заданного числа, используя указатель на целое число, а просто не переменную целого типа

**Математическая модель**

a\*i

**Список идентификаторов**

| **Имя** | **Смысл** | **Тип** |
| --- | --- | --- |
| n | вводимое с клавиатуры число | int |
| a | промежуточная переменная | int |
| \*pa | указатель | int |
| i | параметр цикла | int |

**Код программы**

#include <stdlib.h>

#include <stdio.h>

int main(void) {

int n, a = 1;

printf("Factorial: ");

scanf("%d", &n);

int \*pa = &a;

for (int i = 1; i <= n; i++)

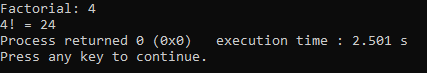
\*pa \*= i;

printf("%d! = %d", n, \*pa);

return 0;

}

**Результат выполненной работы**



**Задача 6**

**Постановка задачи**

Вывести элементы динамического массива целых чисел в обратном порядке, используя указатель и операцию декремента (--).

**Математическая модель**

**Список идентификаторов**

| **Имя** | **Смысл** | **Тип** |
| --- | --- | --- |
| \*A | указатель | int |
| n | размер массива | int |
| i | параметр цикла | int |

**Код программы**

#include <stdio.h>

#include <stdlib.h>

int main(void) {

int \*A, n;

printf("Array size: ");

scanf("%d", &n);

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

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

printf("A[%d] = ", i);

scanf("%d", (A + i));

}

for (int i = n-1; i >= 0; i--)

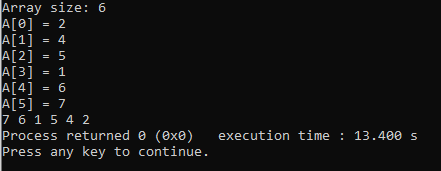
printf("%d ", \*(A+i));

free(A);

return 0;

}

**Результат выполненной работы**



**Задача 7**

**Постановка задачи**

Отсортируйте заданный массив целых чисел, используя указатели, а не доступ по индексу ([]).

**Математическая модель**

**Список идентификаторов**

| **Имя** | **Смысл** | **Тип** |
| --- | --- | --- |
| \*A | указатель | int |
| i | параметр цикла |
| j | параметр цикла |
| n | вводимое с клавиатуры число |

**Код программы**

#include <stdio.h>

#include <stdlib.h>

int main(void) {

int n, \*A;

printf("Array size = ");

scanf("%d", &n);

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

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

printf("A[%d] = ", i);

scanf("%d", (A + i));

}

for (int i = 0; i < n - 1; i++) {

for (int j = n - 1; j > i; j--) {

if (\*(A+j-1) > \*(A+j)) {

int temp = \*(A+j);

\*(A+j) = \*(A+j-1);

\*(A+j-1) = temp;

}

}

}

printf("Sorted array:\n");

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

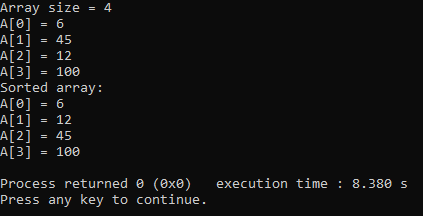
printf("A[%d] = %d\n", i, \*(A + i) );

free(A);

return 0;

}

**Результат выполненной работы**



**Задача 9**

**Постановка задачи**

Выделите память под двумерный динамический массив — матрицу — таким образом, чтобы данные все строки этой матрицы гарантированно располагались в оперативной памяти друг за другом (C 2D array contiguous memory allocation).

**Математическая модель**

**Список идентификаторов**

| **Имя** | **Смысл** | **Тип** |
| --- | --- | --- |
| \*A | указатель | int |
| \*s | указатель |
| i | параметр цикла |
| j | параметр цикла |
| N | кол-во строк матрицы | const |
| M | кол-во столбцов матрицы | const |

**Код программы**

#include <stdio.h>

#include <stdlib.h>

#define N 3

#define M 2

int main(void) {

int \*\*A = (int\*\*)malloc(N \* sizeof(int\*) + N \* M \* sizeof(int));

int \*s = (int \*)((char \*)A + N \* sizeof(int \*));

for (int i = 0; i < N; i++)

A[i] = s + i \* M;

for (int i = 0; i < N; i++)

for (int j = 0; j < M; j++) {

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

scanf("%d", &A[i][j]);

}

printf("\n");

for (int i = 0; i < N; i++) {

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

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

printf("\n");

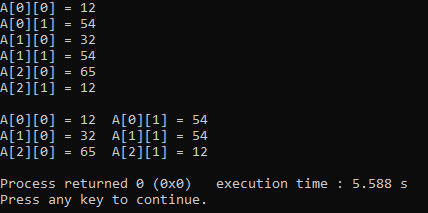
}

free(A);

return 0;

}

**Результат выполненной работы**



**Задача 1**

**Постановка задачи**

Создайте две функции, которые вычисляют факториал числа, используя цикл: • функцию, которая вычисляет факториал, используя цикл;

**Математическая модель**

**Список идентификаторов**

| **Имя** | **Смысл** | **Тип** |
| --- | --- | --- |
| f | функция | int |
| fa | функция | int |
| n | накопительная переменная | int |
| i | параметр цикла | int |
| r | накопительная переменная | int |

**Код программы**

#include <stdio.h>

int f(int n) {

if (n == 1 || n == 0)

n = 1;

else

n = n \* f(n - 1);

return n;

}

int fa(int n) {

int r = 1;

for (int i = 1; i <= n; i ++)

r \*= i;

return r;

}

int main(void) {

int k = f(4);

printf("4! = %d\n", k);

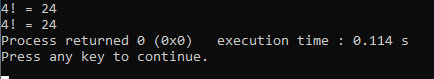
k = fa(4);

printf("4! = %d", k);

return 0;

}

**Результат выполненной работы**



**Задача 2**

**Постановка задачи**

Напишите отдельные функции для вычисления:

-количество размещений из n по k

-количество сочетаний из n по k

**Математическая модель**





**Список идентификаторов**

| **Имя** | **Смысл** | **Тип** |
| --- | --- | --- |
| n | вводимое с клавиатуры число | int |
| k | вводимое с клавиатуры число | int |
| f(a) | функция | int |
| r | накопительная переменная | long int |

**Код программы**

**Размещения**

#include <stdio.h>

#include <math.h>

int main (void)

{ int n,k;

int f(int a)

{

long int r = 1;

for (int i=1;i<=a;i++)

r\*=i;

return r;

}

printf("n = ");

scanf("%d",&n);

printf("k = ");

scanf("%d",&k);

printf("A = %d",f(n)/f(n-k));

return 0;

}

**Сочетания**

#include <stdio.h>

#include <math.h>

int main (void)

{ int n,k;

int f(int a)

{

long int r = 1;

for (int i=1;i<=a;i++)

r\*=i;

return r;

}

printf("n = ");

scanf("%d",&n);

printf("k = ");

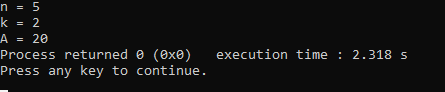
scanf("%d",&k);

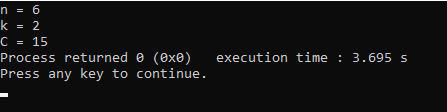
printf("C = %d",f(n)/(f(k)\*f(n-k)));

return 0;

}

**Результат выполненной работы**





**Задача 3**

**Постановка задачи**

Объявите указатель на массив типа int и динамически выделите память для 12-ти элементов. Напишите функцию, которая поменяет значения четных и нечетных ячеек массива

**Математическая модель**

**Список идентификаторов**

| **Имя** | **Смысл** | **Тип** |
| --- | --- | --- |
| n | размер массива | const |
| i | параметр цикла | int |
| \*array | указатель |
| s | функция |
| arr | параметр цикла |

**Код программы**

#include <stdio.h>

#include <stdlib.h>

#define n 12

void s(int\* a) {

int i = 0;

while (i < n - 1) {

int t = a[i+1];

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

a[i] = t;

i += 2;

}

}

int main(void) {

int \*array;

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

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

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

scanf("%d", &array[i]);

}

s(array);

printf("\n");

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

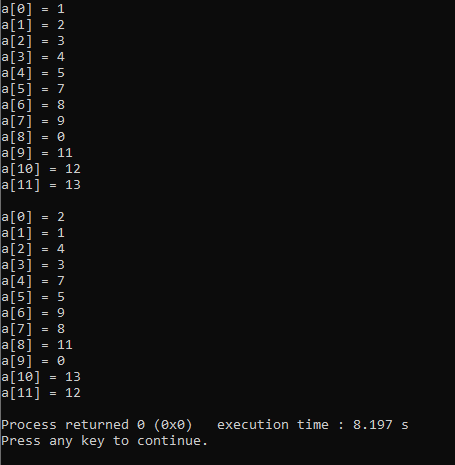
printf("a[%d] = %d\n", i, array[i]);

free(array);

return 0;

}

**Результат выполненной работы**



**Задача 4**

**Постановка задачи**

Создать функцию для динамического выделения памяти под двумерный динамический массив типа double — матрицу

**Математическая модель**

**Список идентификаторов**

| **Имя** | **Смысл** | **Тип** |
| --- | --- | --- |
| \*\*p | указатель | double |
| n | строка | int |
| m | столбец | int |
| i | параметр цикла | int |
| j | параметр цикла | int |

**Код программы**

#include <stdio.h>

#include <stdlib.h>

int main(void) {

double \*\*p = NULL;

int n = 3;

int m = 4;

p = (double \*\*) malloc(n \* sizeof(double \*));

if (p == NULL) {

return 1;

}

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

p[i] = (double \*) malloc(m \* sizeof(double));

}

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

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

p[i][j] = rand()%100;

}

}

printf("p =\n");

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

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

printf("%.1f ", p[i][j]);

}

printf("\n");

}

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

free(p[i]);

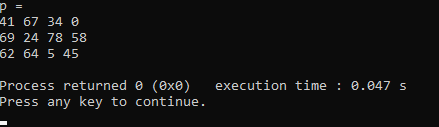
}

free(p);

return 0;

}

**Результат выполненной работы**

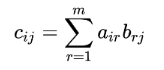


**Задача 5**

**Постановка задачи**

Создать функцию умножения матрицы типа double на вектор-столбец типа double

**Математическая модель**



**Список идентификаторов**

| **Имя** | **Смысл** | **Тип** |
| --- | --- | --- |
| n\_rows | кол-во строк матрицы | int |
| m\_columns | кол-во столбцов матрицы | int |
| \*vector | указатель | double |
| n | размер вектор-столбца | int |
| i | параметр цикла | int |
| j | параметр цикла | int |
| r | параметр цикла | int |

**Код программы**

#include <stdio.h>

#include <stdlib.h>

void fillMatrix(int n\_rows, int m\_columns, double \*\*\_A) {

for (int i = 0; i < n\_rows; i++)

for (int j = 0; j < m\_columns; j++) {

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

scanf("%lf", &\_A[i][j]);

}

}

void freeMatrix(int n\_rows, double \*\*\_A) {

for (int i = 0; i < n\_rows; i++)

free(\_A[i]);

free(\_A);

}

double\*\* createMatrix(int n\_rows, int m\_columns) {

double \*\*\_A;

\_A = (double \*\*)malloc(n\_rows \* sizeof(double \*));

for (int i = 0; i < n\_rows; i ++)

\_A[i] = (double \*)malloc(m\_columns \* sizeof(double));

return \_A;

}

void multVector(int n\_rows, int m\_columns, double \*\*\_A, int n, double \*\_vector) {

double \*result;

result = (double \*) calloc(n, sizeof(double));

for (int i = 0; i < n\_rows; i++) {

for (int j = 0; j < m\_columns; j++)

result[i] += \_vector[j] \* \_A[i][j];

printf("A[%d] = %lf\n", i, result[i]);

}

free(result);

}

void multMatrix(int l, int q, int k, double \*\*\_A1, double \*\*\_A2) {

double \*\*result;

result = createMatrix(l, q);

for (int i = 0; i < l; i++)

for (int j = 0; j < q; j++) {

result[i][j] = 0;

for (int r = 0; r < k; r++)

result[i][j] += \_A1[i][r] \* \_A2[r][j];

printf("A[%d][%d] = %lf\n", i, j, result[i][j]);

}

freeMatrix(l, result);

}

int main(void) {

double\*\* matrix1;

int n\_rows1, m\_columns1;

int choice;

printf("Matrix size:\nn = ");

scanf("%d", &n\_rows1);

printf("m = ");

scanf("%d", &m\_columns1);

matrix1 = createMatrix(n\_rows1, m\_columns1);

fillMatrix(n\_rows1, m\_columns1, matrix1);

printf("1 - multiply matrix by column-vector\n"

"2 - multiply matrix by matrix\n"

"(1/2)\n");

scanf("%d", &choice);

if (choice == 1) {

double \*vector;

int n;

printf("Enter size of column-vector:\nn = ");

scanf("%d", &n);

if (n\_rows1 != n) {

printf("Error!");

return 1;

}

vector = (double \*) malloc(n \* sizeof(double));

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

printf("Vector[%d][0] = ", i);

scanf("%lf", &vector[i]);

}

multVector(n\_rows1, m\_columns1, matrix1, n, vector);

free(vector);

}

else if (choice == 2) {

double \*\*matrix2;

int n\_rows2, m\_columns2;

printf("Matrix size:\nn = ");

scanf("%d", &n\_rows2);

printf("m = ");

scanf("%d", &m\_columns2);

if (n\_rows1 != m\_columns2) {

printf("Error!");

return 1;

}

matrix2 = createMatrix(n\_rows2, m\_columns2);

fillMatrix(n\_rows2, m\_columns2, matrix2);

multMatrix(n\_rows1, m\_columns2, n\_rows2, matrix1, matrix2);

freeMatrix(n\_rows2, matrix2);

}

freeMatrix(n\_rows1, matrix1);

return 0;

}

**Результат выполненной работы**

