**Белорусский государственный технологический университет**

**Факультет информационных технологий**

**Кафедра ПИ**

Лабораторная работа № 16

По дисциплине «Основы алгоритмизации и программирования»

На тему «Функции пользователя»

Выполнил:  
Студент 1 курса, 10 группа  
Короткевич Артём Сергеевич  
Преподаватель: асс. Харланович А.В

2024, Минск  
  
  
  
  
  
**Задание 1**Изучить способы передачи значений *переменных* в функцию, выполнив программы, записанные в правой части.  
Записать условие задачи.  
 **Условие**Написать алгоритмы, который будет выводить элементы, возведённые в квадрат от 1 до 10(включительно), при помощи передачи параметра  
по значению, передачи параметра по указателю и передачи параметра по ссылке

**Код(1)**

#include <iostream>  
using namespace std;

int square(int);

int main()

{

for (int x = 1; x <= 10; x++)

cout << square(x) << ' ';

return 0;

}

int square(int y)

{

return y \* y;   
}

**Код(2)**

#include <iostream>  
using namespace std;

int square(int\*);

int main()

{

for (int x = 1; x <= 10; x++)

cout << square(&x) << ' ';

return 0;

}

int square(int\* y)

{

return (\*y) \* (\*y);   
}

**Код(3)**

#include <iostream>  
using namespace std;

int square(int&);

int main()

{

for (int x = 1; x <= 10; x++)

cout << square(x) << ' ';

return 0;

}

int square(int& y)

{

return y \* y;   
}

**Результат**  
  
**Задание 2**  
Изучить использование *указателей* и *ссылок* как *формальных параметров* функции при работе с *одномерным* *массивом данных*, опробовав работу программ в правой части.  
Написать условие задачи.  
  
**Условие**  
Написать алгоритм, который будет высчитывать сумму элементов массива с 1 по 5 элемент(не включительно) при помощи указателей и ссылок в виде формальных параметров функции при работе с одномерным массивом данных.  
  
**Код**

#include <iostream>

using namespace std;

int sum(int& x)

{

int\* p = &x, res = 0;

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

res += p[i];

return res;

}

int main()

{

int mas[5], i;

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

{

mas[i] = i \* i;

cout << mas[i] << ' ';

}

cout << endl << "result=" << sum(\*mas) << endl;

}

**Результат**

  
 **Задание 3**  
В программе, записанной справа, используется меню, разработанное с помощью оператора **switch**, которое позволяет делать выбор между двумя функциями.  
Написать условие задачи и комментарии к программе.  
  
**Условие**Написать алгоритм, для которого будет создано “меню” с вариантами выбора для работы с одномерным массивом и матрицей.  
  
**Код + комментарии**

#include <iostream>  
using namespace std;

int massiv(int\* M, int size); // прототип функции для одномерного массива

void matrix(); // прототип функции для работы с матрицей

int main(void)

{

setlocale(LC\_CTYPE, "Russian");

int choice;

do // меню пользователя

{

cout << "Выберите вариант работы" << endl;

cout << "1 - с одномерным массивом" << endl;

cout << "2 - с матрицей" << endl;

cout << "3 - выход" << endl;

cin >> choice;

switch (choice)

{

case 1: {

// работа с массивом

int\* M, size, i, av;

cout << "Введите размер массива ";

cin >> size;

M = new int[size]; // выделяем памяти для массива

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

{

cout << "Введите " << i + 1 << " число ";

cin >> \*(M + i);

}

av = massiv(M, size); // вызов функции для обработки массива и получение результата

cout << "Результат=" << av << endl;

delete[] M; // освобождаем память

break;

}

case 2:

// работа с матрицей

matrix();

break;

case 3:

break; // завершение работы программы при "3"

default: // если пользователь ввел некорретный символ в меню

cout << "Некорректный выбор, попробуйте снова." << endl;

}

} while (choice != 3); // цикл продолжается, пока выбор не равен 3 (выход)

}

int massiv(int\* M, int size) // функция для обработки одномерного массива

{

int sum = 0, avar; // сумма элементов массива и переменная для среднего значения

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

sum += \*(M + i); // подсчет суммы элементов массива

avar = sum / size; // вычисление среднего значения

return avar; // возврат результата

}

void matrix() // Функция для обработки матрицы

{

int\*\* A, row, col, i, j, m; // переменные для работы с матрицей

cout << "Введите число строк матрицы "; // запрос числа строк

cin >> row;

cout << "Введите чиcло столбцов "; // запрос числа столбцов

cin >> col;

A = new int\* [row]; // выделяем память для матрицы

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

A[i] = new int[col];

for (i = 0; i < row; i++) // вводим элементы матрицы

for (j = 0; j < col; j++)

{

cout << "Введите A[" << i << "],[" << j << "]= ";

cin >> \*(\*(A + i) + j);

}

m = A[0][0]; // инициализация максимального элемента

for (i = 0; i < row; i++) // ищем максимальный элемент в матрице

for (j = 0; j < col; j++)

if (\*(\*(A + i) + j) > m)

m = \*(\*(A + i) + j); // обновляем максимальный элемент

cout << "Результат=" << m << endl;

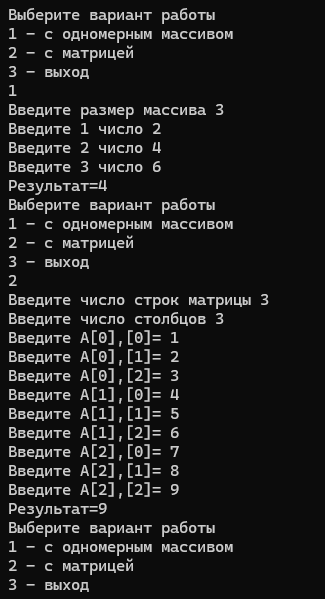
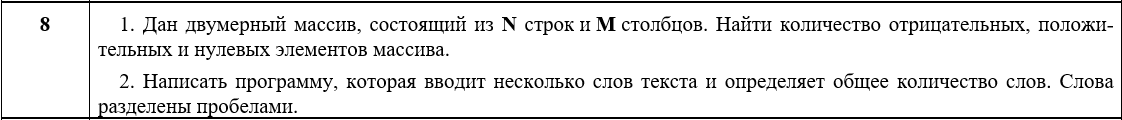
for (i = 0; i < row; i++)

delete[] A[i];

delete[] A; // освобождаем память матрицы

}

**Результат**

  
  
**Задание 4**  
  
**Вариант 8**  
  
  
  
**Код**

#include <iostream>  
#include <sstream>

#include <Windows.h>

using namespace std;

int matrix(int\*\* A, int N, int M, int k) { // подключение двух функций для матрицы и строки

return 0;

}

int massiv(char str1, char str) {

return 0;

}

int main()

{

setlocale(LC\_ALL, "Russian");

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int choice;

do //диалоговая программа с решением наших задач

{

cout << "Выберите вариант работы" << endl;

cout << "1 - с матрицей" << endl;

cout << "2 - со строкой" << endl;

cout << "3 - выход" << endl;

cin >> choice;

switch (choice)

{

case 1: { // работа с матрицей

int\*\* A, N, M; // инициализация переменных

bool found = false; // инициализация флага

cout << "Введите число строк: ";

cin >> N;

cout << "Введите число столбцов: ";

cin >> M;

A = new int\* [N]; // инициализация динамического массива

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

A[i] = new int[M];

}

cout << "Введите матрицу: " << endl;

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

{

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

{

cout << "A(" << i + 1 << ", " << j + 1 << "): ";

cin >> A[i][j];

}

}

cout << "Исходная матрица: " << endl;

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

{

cout << "[";

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

{

cout << " " << A[i][j] << " ";

}

cout << "]" << endl;

}

int maxElement = A[0][0]; // иницмализация максимального элемента

int positions[100][2]; // предполагаем, что максимальных элементов не больше 100

int posCount = 0; // счетчик позиции максимального элемента

int sum = 0; // ининициализируем переменную, которая будет суммировать элементы, которые ниже главное диагонали

for (int i = 0; i < N; i++) { //с помощью цикла находим один или несколько максимальных элементов и их позиции

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

if (A[i][j] > maxElement) {

maxElement = A[i][j];

posCount = 0;

positions[posCount][0] = i;

positions[posCount][1] = j;

posCount++;

}

else if (A[i][j] == maxElement) {

positions[posCount][0] = i;

positions[posCount][1] = j;

posCount++;

}

if (i > j) { // находим сумму элементов, которые ниже главной диагонали

sum += A[i][j];

}

}

}

cout << "Максимальный элемент: " << maxElement << endl;

cout << "Позиции максимального элемента: " << endl;

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

cout << "(" << positions[i][0] + 1 << ", " << positions[i][1] + 1 << ")" << endl;

}

cout << "Сумма элементов ниже главной диагонали: " << sum << endl;

break;

}

//работа со строкой

case 2: {

setlocale(LC\_ALL, "Russian");

char str1[50], str[] = "\*"; // объявление двух массивов

int n; // объявление переменной, которая показывает содержание символов в строке

cout << "Введите строку: ";

cin >> str1;

n = strlen(str1); // находим длину строки

// создаем цикл for, который проходит по символам строки и ищет "а"

for (int i = 0; str1[i] != '\0'; i++) {

if (str1[i] == 'c')

{

// если в строке есть символ "а", то после его необходимо вставить "!"

for (int j = n + 1; j != i; j--)

{

str1[j + 1] = str1[j];

}

str1[i + 1] = '\*';

}

}

cout << "Вывод строки: " << str1 << endl;

break;

}

case 3:

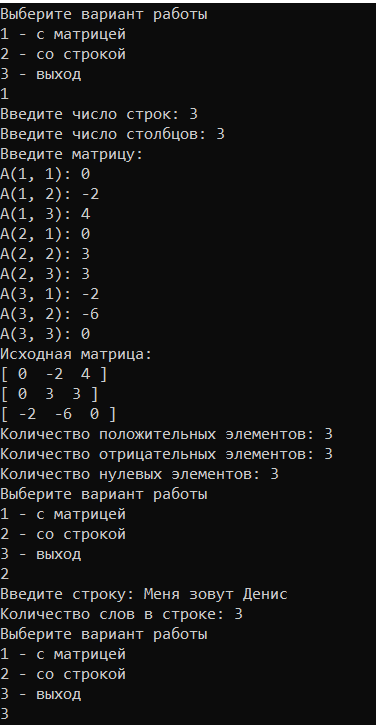
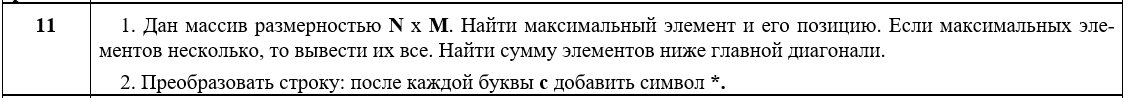
break; // выход из программы

}

} while (choice != 3); // работаем с диалогом, пока не выберем 3 вариант

return 0;

}

**Результат**  
  
  
  
  
**Дополнительные задания**  
  
**Вариант 11**  
  
  
  
  
**Код**

#include <iostream>  
#include <sstream>

#include <Windows.h>

using namespace std;

int matrix(int\*\* A, int N, int M, int k) { // подключение двух функций для матрицы и строки

return 0;

}

int massiv(char str1, char str) {

return 0;

}

int main()

{

setlocale(LC\_ALL, "Russian");

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int choice;

do //диалоговая программа с решением наших задач

{

cout << "Выберите вариант работы" << endl;

cout << "1 - с матрицей" << endl;

cout << "2 - со строкой" << endl;

cout << "3 - выход" << endl;

cin >> choice;

switch (choice)

{

case 1: { // работа с матрицей

int\*\* A, N, M; // инициализация переменных

bool found = false; // инициализация флага

cout << "Введите число строк: ";

cin >> N;

cout << "Введите число столбцов: ";

cin >> M;

A = new int\* [N]; // инициализация динамического массива

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

A[i] = new int[M];

}

cout << "Введите матрицу: " << endl;

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

{

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

{

cout << "A(" << i + 1 << ", " << j + 1 << "): ";

cin >> A[i][j];

}

}

cout << "Исходная матрица: " << endl;

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

{

cout << "[";

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

{

cout << " " << A[i][j] << " ";

}

cout << "]" << endl;

}

int maxElement = A[0][0]; // иницмализация максимального элемента

int positions[100][2]; // предполагаем, что максимальных элементов не больше 100

int posCount = 0; // счетчик позиции максимального элемента

int sum = 0; // ининициализируем переменную, которая будет суммировать элементы, которые ниже главное диагонали

for (int i = 0; i < N; i++) { //с помощью цикла находим один или несколько максимальных элементов и их позиции

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

if (A[i][j] > maxElement) {

maxElement = A[i][j];

posCount = 0;

positions[posCount][0] = i;

positions[posCount][1] = j;

posCount++;

}

else if (A[i][j] == maxElement) {

positions[posCount][0] = i;

positions[posCount][1] = j;

posCount++;

}

if (i > j) { // находим сумму элементов, которые ниже главной диагонали

sum += A[i][j];

}

}

}

cout << "Максимальный элемент: " << maxElement << endl;

cout << "Позиции максимального элемента: " << endl;

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

cout << "(" << positions[i][0] + 1 << ", " << positions[i][1] + 1 << ")" << endl;

}

cout << "Сумма элементов ниже главной диагонали: " << sum << endl;

break;

}

//работа со строкой

case 2: {

setlocale(LC\_ALL, "Russian");

char str1[50], str[] = "\*"; // объявление двух массивов

int n; // объявление переменной, которая показывает содержание символов в строке

cout << "Введите строку: ";

cin >> str1;

n = strlen(str1); // находим длину строки

// создаем цикл for, который проходит по символам строки и ищет "а"

for (int i = 0; str1[i] != '\0'; i++) {

if (str1[i] == 'c')

{

// если в строке есть символ "а", то после его необходимо вставить "!"

for (int j = n + 1; j != i; j--)

{

str1[j + 1] = str1[j];

}

str1[i + 1] = '\*';

}

}

cout << "Вывод строки: " << str1 << endl;

break;

}

case 3:

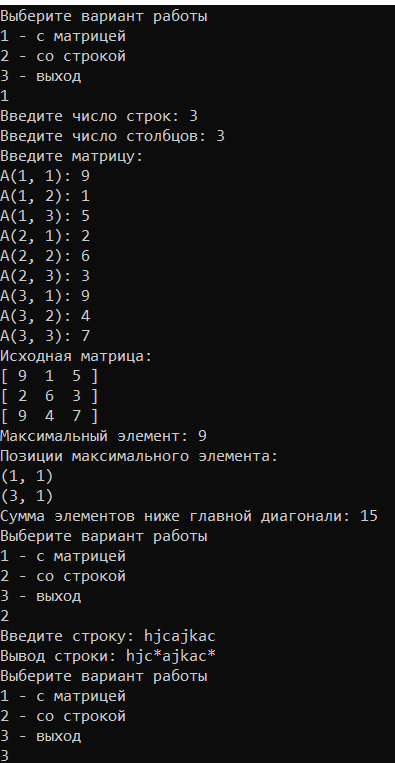
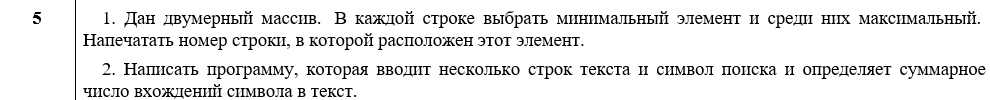
break; // выход из программы

}

} while (choice != 3); // работаем с диалогом, пока не выберем 3 вариант

return 0;

}

**Результат**  
  
  
  
  
**Вариант 5**  
  
  
  
**Код**

#include <iostream>  
#include <string>

#include <Windows.h>

using namespace std;

void processMatrix() { // функция для работы с матрицей

int\*\* A, N, M;

cout << "Введите число строк: ";

cin >> N;

cout << "Введите число столбцов: ";

cin >> M;

A = new int\* [N]; // выделяем память для матрицы

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

A[i] = new int[M];

}

cout << "Введите элементы матрицы:" << endl;

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

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

cout << "A(" << i + 1 << ", " << j + 1 << "): ";

cin >> A[i][j];

}

}

cout << "Исходная матрица:" << endl;

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

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

cout << A[i][j] << " ";

}

cout << endl;

}

// ищем минимальные элементы в строках

int maxOfMin = INT\_MIN; // максимум среди минимальных(можно было обойтись, не использовав библиотеку <Windows.h>, а записать значение переменной как 1e9

int rowWithMaxMin = -1; // номер строки с максимальным из минимальных

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

int minInRow = A[i][0]; // минимум в текущей строке

for (int j = 1; j < M; j++) {

if (A[i][j] < minInRow) {

minInRow = A[i][j];

}

}

if (minInRow > maxOfMin) {

maxOfMin = minInRow;

rowWithMaxMin = i + 1; // запоминаем номер строки (с 1)

}

}

cout << "Максимальный из минимальных элементов: " << maxOfMin << endl;

cout << "Номер строки с этим элементом: " << rowWithMaxMin << endl;

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

delete[] A[i];

}

delete[] A;

}

void processText() { // функция для работы с текстом

cin.ignore(); // очищаем буфер после ввода

string text;

char searchChar;

cout << "Введите текст (несколько строк, завершите ввод пустой строкой):" << endl;

getline(cin, text); // чтение текста

cout << "Введите символ для поиска: ";

cin >> searchChar;

int count = 0; // подсчёт вхождений символа

for (char c : text) {

if (c == searchChar) {

count++;

}

}

cout << "Символ '" << searchChar << "' встречается " << count << " раз." << endl;

}

int main() {

setlocale(LC\_ALL, "Russian");

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int choice;

do {

cout << "Выберите вариант работы:" << endl;

cout << "1 - работа с матрицей" << endl;

cout << "2 - работа с текстовой строкой" << endl;

cout << "3 - выход" << endl;

cin >> choice;

switch (choice) {

case 1:

processMatrix();

break;

case 2:

processText();

break;

case 3:

cout << "Выход из программы." << endl;

break;

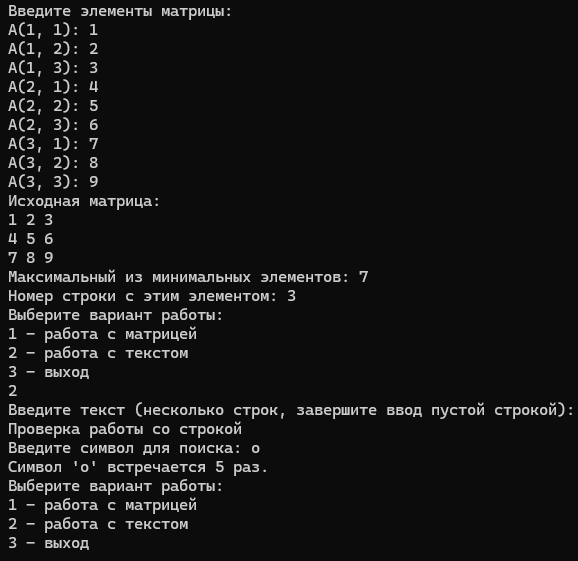
default:

cout << "Некорректный выбор. Попробуйте снова." << endl;

}

} while (choice != 3);

return 0;  
}

**Результат**  
  
  
  
  
  
  
**Вариант 16**  
  
  
  
**Код**

#include <iostream>  
#include <string>

using namespace std;

int countMaxOccurrences(int\* M, int size); // прототип функции для работы с одномерным массивом

void findShortestAndLongestWord(); // прототип функции для работы со строками

int main(void)

{

setlocale(LC\_CTYPE, "Russian");

int choice;

do

{

// меню выбора

cout << "Выберите вариант работы" << endl;

cout << "1 - с одномерным массивом" << endl;

cout << "2 - с текстовой строкой" << endl;

cout << "3 - выход" << endl;

cin >> choice;

switch (choice)

{

case 1: {

int\* M, size;

cout << "Введите размер массива: ";

cin >> size;

if (size <= 0) {

cout << "Размер массива должен быть положительным." << endl;

break;

}

M = new int[size]; // выделяем память для массива

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

{

cout << "Введите " << i + 1 << " число: ";

cin >> \*(M + i);

}

int count = countMaxOccurrences(M, size); // вызов функции для подсчёта вхождений максимального числа

cout << "Максимальное число встречается " << count << " раз." << endl;

delete[] M; // освобождение память из массива

break;

}

case 2:

findShortestAndLongestWord();

break;

case 3:

cout << "Выход из программы." << endl; // завершение работы программы

break;

default:

cout << "Некорректный выбор, попробуйте снова." << endl; // сообщение при некорректном вводе

}

} while (choice != 3); // цикл продолжается, пока не выбран выход "3"

return 0;

}

int countMaxOccurrences(int\* M, int size) // функция для подсчёта количества вхождений максимального числа в массиве

{

int maxNum = M[0]; // инициализация первого элемента как максимального

int count = 0; // счётчик количества вхождений

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

{

if (\*(M + i) > maxNum) {

maxNum = \*(M + i); // обновляем максимальное число

count = 1; // сбрасываем счётчик

}

else if (\*(M + i) == maxNum) {

count++; // увеличиваем счётчик, если число равно текущему максимуму

}

}

return count;

}

void findShortestAndLongestWord() // функция для поиска самого короткого и самого длинного слова в строке

{

cin.ignore(); // очищаем буфера ввода после выбора в меню

string input;

cout << "Введите строку: ";

getline(cin, input); // ввод строки от пользователя

string shortestWord = "", longestWord = ""; // переменные для самого короткого и длинного слова

string currentWord = ""; // переменная для текущего слова

int minLength = input.length(), maxLength = 0; // инициализация длины слов

for (char c: input) // перебор символов строки

{

if (c == ' ' || c == '\t') {

if (!currentWord.empty()) { // если встретился пробел, проверяем текущее слово

if (shortestWord.empty() || currentWord.length() < minLength) {

shortestWord = currentWord; // устанавливаем текущее слово как самое короткое

minLength = currentWord.length();

}

if (currentWord.length() > maxLength) {

longestWord = currentWord; // устанавливаем текущее слово как самое длинное

maxLength = currentWord.length();

}

}

else {

currentWord += c; // добавляем символ к текущему слову

}

}

if (!currentWord.empty()) { // проверяем последнего слова (если строка не заканчивается пробелом)

if (currentWord.length() < minLength) {

shortestWord = currentWord;

}

if (currentWord.length() > maxLength) {

longestWord = currentWord;

}

}

if (shortestWord.empty() || longestWord.empty()) {

cout << "Введена пустая строка или отсутствуют слова." << endl;

}

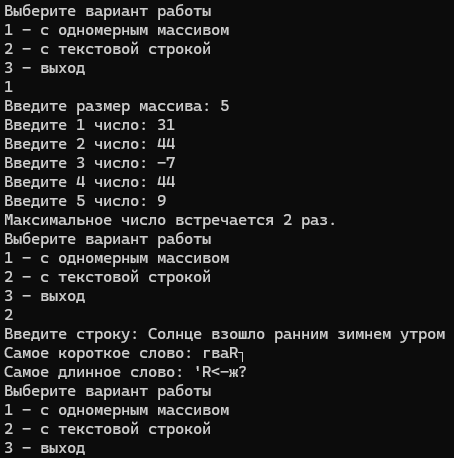
else {

cout << "Самое короткое слово: " << shortestWord << endl;

cout << "Самое длинное слово: " << longestWord << endl;

}

}

**Результат**  
  
  
  
Примечание: Не смог найти в чём ошибка при выводе переменных shortestWord и longestWord