Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

 Лабораторная работа 1

По дисциплине «Основы алгоритмизации и программирования»

На тему «Указатели на функции»

Выполнил:

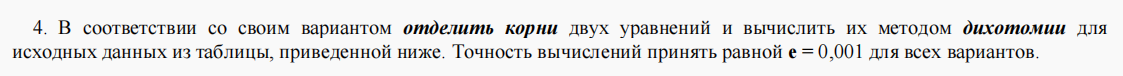
Студент 1 курса 6 группы

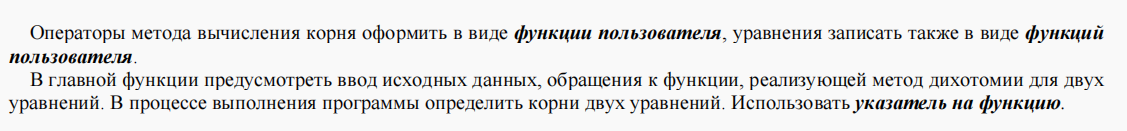
Романов Игорь Вячеславович

Преподаватель: асс. Андронова М.В.

2024, Минск

Задание 4





Вариант 12 (основной)



#include <iostream> // подключение библиотеки для работы с вводом/выводом данных на экран

using namespace std; // подключение стандартного пространства имен

// прототипы функций

double function1(double x);

double function2(double x);

double methodDY(double (\*function)(double), double a, double b);

// объявление функции 1

double function1(double x)

{

return 2 \* x + x \* x \* x - 7;

}

// объявление функции 2

double function2(double x)

{

double e = 0.001;

return pow(e, x) + 2 \* x ;

}

//метод дитохомии

double methodDY(double (\*function)(double), double a, double b) {

double e = 0.001, x; // определение погрешности и переменной x

while (abs(a - b) > 2 \* e) {

x = (a + b) / 2;

if (function(x) \* function(a) <= 0) {

b = x;

}

else {

a = x;

}

}

return x;

}

int main()

{

setlocale(LC\_ALL, "RUS"); // подключение русского алфавита

double a, b;

cout << "Введите интервал [a; b]: "; cin >> a; cin >> b; //ввод интервала

// проверка корректности интервала

if (function1(a) \* function1(b) > 0 && function2(a) \* function2(b) > 0) {

cout << "На концах интервала функции имеют одинаковый знак. Измените интервал.";

return 1; //возврат 1 в случае ошибки

}

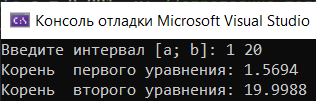
// поиск корней

cout << "Корень первого уравнения: " << methodDY(function1, a, b) << endl; // вызов функции для нахождения корня в первом уравнении

cout << "Корень второго уравнения: " << methodDY(function2, a, b); // вызов функции для нахождения корня во втором уравнении

return 0; // возвращение значения 0 по окончанию программы

}



Доп. задания

Вариант 10



#include <iostream> // подключение библиотеки для работы с вводом/выводом данных на экран

using namespace std; // подключение стандартного пространства имен

// прототипы функций

double function1(double x);

double function2(double x);

double methodDY(double (\*function)(double), double a, double b);

// объявление функции 1

double function1(double x)

{

return sin(x) + x \* x \* x;

}

// объявление функции 2

double function2(double x)

{

return 0.4 + x \* x \* x;

}

//метод дитохомии

double methodDY(double (\*function)(double), double a, double b) {

double e = 0.001, x; // определение погрешности и переменной x

while (abs(a - b) > 2 \* e) {

x = (a + b) / 2;

if (function(x) \* function(a) <= 0) {

b = x;

}

else {

a = x;

}

}

return x;

}

int main()

{

setlocale(LC\_ALL, "RUS"); // подключение русского алфавита

double a, b;

cout << "Введите интервал [a; b]: "; cin >> a; cin >> b; //ввод интервала

// проверка корректности интервала

if (function1(a) \* function1(b) > 0 && function2(a) \* function2(b) > 0) {

cout << "На концах интервала функции имеют одинаковый знак. Измените интервал.";

return 1; //возврат 1 в случае ошибки

}

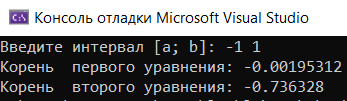
// поиск корней

cout << "Корень первого уравнения: " << methodDY(function1, a, b) << endl; // вызов функции для нахождения корня в первом уравнении

cout << "Корень второго уравнения: " << methodDY(function2, a, b); // вызов функции для нахождения корня во втором уравнении

return 0; // возвращение значения 0 по окончанию программы

}



Вариант 11



#include <iostream> // подключение библиотеки для работы с вводом/выводом данных на экран

using namespace std; // подключение стандартного пространства имен

// прототипы функций

double function1(double x);

double function2(double x);

double methodDY(double (\*function)(double), double a, double b);

// объявление функции 1

double function1(double x)

{

double e = 0.001;

return pow(e, x) + 2 \* x \* x - 3;

}

// объявление функции 2

double function2(double x)

{

return x \* x \* x + 3;

}

//метод дитохомии

double methodDY(double (\*function)(double), double a, double b) {

double e = 0.001, x; // определение погрешности и переменной x

while (abs(a - b) > 2 \* e) {

x = (a + b) / 2;

if (function(x) \* function(a) <= 0) {

b = x;

}

else {

a = x;

}

}

return x;

}

int main()

{

setlocale(LC\_ALL, "RUS"); // подключение русского алфавита

double a, b;

cout << "Введите интервал [a; b]: "; cin >> a; cin >> b; //ввод интервала

// проверка корректности интервала

if (function1(a) \* function1(b) > 0 && function2(a) \* function2(b) > 0) {

cout << "На концах интервала функции имеют одинаковый знак. Измените интервал.";

return 1; //возврат 1 в случае ошибки

}

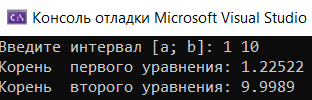
// поиск корней

cout << "Корень первого уравнения: " << methodDY(function1, a, b) << endl; // вызов функции для нахождения корня в первом уравнении

cout << "Корень второго уравнения: " << methodDY(function2, a, b); // вызов функции для нахождения корня во втором уравнении

return 0; // возвращение значения 0 по окончанию программы

}



Вариант 13



#include <iostream> // подключение библиотеки для работы с вводом/выводом данных на экран

using namespace std; // подключение стандартного пространства имен

// прототипы функций

double function1(double x);

double function2(double x);

double methodDY(double (\*function)(double), double a, double b);

// объявление функции 1

double function1(double x)

{

return 5 \* x - 1 + x \* x \* x;

}

// объявление функции 2

double function2(double x)

{

return x \* x + (1 / x);

}

//метод дитохомии

double methodDY(double (\*function)(double), double a, double b) {

double e = 0.001, x; // определение погрешности и переменной x

while (abs(a - b) > 2 \* e) {

x = (a + b) / 2;

if (function(x) \* function(a) <= 0) {

b = x;

}

else {

a = x;

}

}

return x;

}

int main()

{

setlocale(LC\_ALL, "RUS"); // подключение русского алфавита

double a, b;

cout << "Введите интервал [a; b]: "; cin >> a; cin >> b; //ввод интервала

// проверка корректности интервала

if (function1(a) \* function1(b) > 0 && function2(a) \* function2(b) > 0) {

cout << "На концах интервала функции имеют одинаковый знак. Измените интервал.";

return 1; //возврат 1 в случае ошибки

}

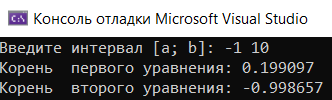
// поиск корней

cout << "Корень первого уравнения: " << methodDY(function1, a, b) << endl; // вызов функции для нахождения корня в первом уравнении

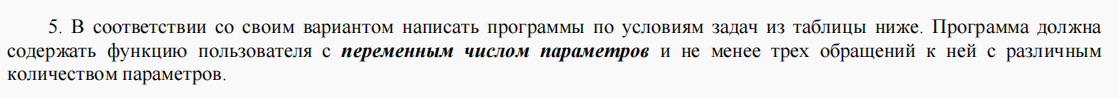
cout << "Корень второго уравнения: " << methodDY(function2, a, b); // вызов функции для нахождения корня во втором уравнении

return 0; // возвращение значения 0 по окончанию программы

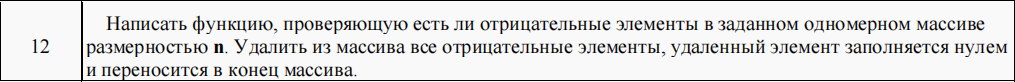
}



Задание 5



Вариант 12(основной)



#include <iostream> // подключение библиотеки для работы с вводом/выводом данных

using namespace std; // подключение стандартного пространства имен

// прототипы функций

void delNegative(int\* n, ...);

void outputArray(int\* array);

// функции удаления отрицательных элементов и добавления 0 в конец массива

void delNegative(int\* n, ...) {

int size = \*n; //размер массива

int\* ptr = n; // указатель на первый элемент массива

for (int i = 1; i <= size; i++) {

if (\*(ptr + i) < 0) {

for (int j = i; j < size; j++) { // перемещаем элементы на одну позицию влево

\*(ptr + j) = \*(ptr + j + 1);

}

\*(ptr + size) = 0; // заполняем последний элемент нулем

size--; // уменьшаем размер массива

i--; // проверяем текущий элемент заново

}

}

}

// функция вывода массивов

void outputArray(int\* array) {

int size = \*array;

for (int i = 1; i <= size; i++) {

cout << array[i] << " "; // начинаем считывать элементы со второго, так как первый элемент - это размер массива

}

cout << endl;

}

// Главная функция

int main()

{

setlocale(LC\_ALL, "RUS");

int arr1[4] = { 3, 7, -2, 9 }; // массив 1

int arr2[8] = { 5, -1, 10, 8, 4, -2 }; // массив 2

int arr3[9] = { 8, 0, 7, -3, -2, 6, -1, 1, 9 }; // массив 3

//вывод массивов поочередно

cout << "Первый массив: "; outputArray(arr1);

cout << "Второй массив: "; outputArray(arr2);

cout << "Третий массив: "; outputArray(arr3);

// вызовы функции

delNegative(arr1);

delNegative(arr2);

delNegative(arr3);

cout << "Массивы после модификации: " << endl;

//вывод массивов поочередно

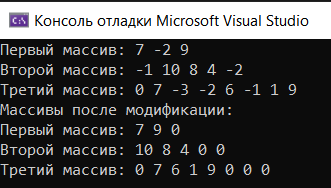
cout << "Первый массив: "; outputArray(arr1);

cout << "Второй массив: "; outputArray(arr2);

cout << "Третий массив: "; outputArray(arr3);

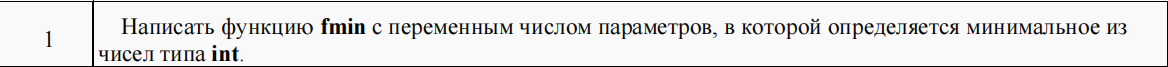
return 0;

}



Доп. задания:

Вариант 1



#include <iostream> // подключение библиотеки для работы с вводом/выводом данных

using namespace std; // подключение стандартного пространства имен

// прототипы функций

int fmin(int\* n, ...);

void outputArray(int\* array);

// функции поиска минимального числа

int fmin(int\* n, ...) {

int size = \*n; //размер массива

int\* ptr = n; // указатель на первый элемент массива

int imin = \*(ptr + 1); //минимальное число принимается за первое число из массива

for (int i = 1; i <= size; i++) {

if (\*(ptr + i) < imin) {

imin = \*(ptr + i);

}

}

return imin;

}

// функция вывода массивов

void outputArray(int\* array) {

int size = \*array;

for (int i = 1; i <= size; i++) {

cout << array[i] << " "; // начинаем считывать элементы со второго, так как первый элемент - это размер массива

}

cout << endl;

}

// Главная функция

int main()

{

setlocale(LC\_ALL, "RUS");

int arr1[4] = { 3, 7, -2, 9 }; // массив 1

int arr2[8] = { 5, -1, 10, 8, 4, -2 }; // массив 2

int arr3[9] = { 8, 0, 7, -3, -2, 6, -1, 1, 9 }; // массив 3

//вывод массивов поочередно

cout << "Первый массив: "; outputArray(arr1);

cout << "Второй массив: "; outputArray(arr2);

cout << "Третий массив: "; outputArray(arr3);

// вывод соответствующих сообщений

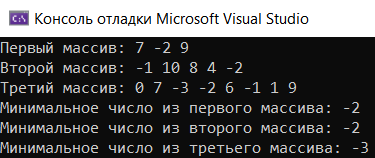
cout << "Минимальное число из первого массива: " << fmin(arr1) << endl;

cout << "Минимальное число из второго массива: " << fmin(arr2) << endl;

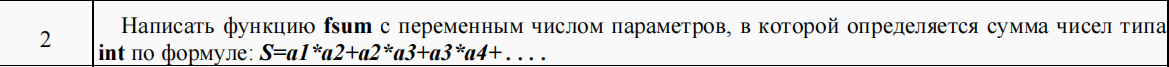
cout << "Минимальное число из третьего массива: " << fmin(arr3);

return 0;

}



Вариант 2



#include <iostream> // подключение библиотеки для работы с вводом/выводом данных

using namespace std; // подключение стандартного пространства имен

// прототипы функций

int fsum(int\* n, ...);

void outputArray(int\* array);

// функции расчет суммы

int fsum(int\* n, ...) {

int size = \*n; //размер массива

int\* ptr = n; // указатель на первый элемент массива

int sum = 0; //создание переменной для суммы

for (int i = 1; i < size; i++) {

sum += \*(ptr + i) \*= \*(ptr + i + 1);

}

return sum;

}

// функция вывода массивов

void outputArray(int\* array) {

int size = \*array;

for (int i = 1; i <= size; i++) {

cout << array[i] << " "; // начинаем считывать элементы со второго, так как первый элемент - это размер массива

}

cout << endl;

}

// Главная функция

int main()

{

setlocale(LC\_ALL, "RUS");

int arr1[4] = { 3, 7, -2, 9 }; // массив 1

int arr2[8] = { 5, -1, 10, 8, 4, -2 }; // массив 2

int arr3[9] = { 8, 0, 7, -3, -2, 6, -1, 1, 9 }; // массив 3

//вывод массивов поочередно

cout << "Первый массив: "; outputArray(arr1);

cout << "Второй массив: "; outputArray(arr2);

cout << "Третий массив: "; outputArray(arr3);

// вывод соответствующих сообщений

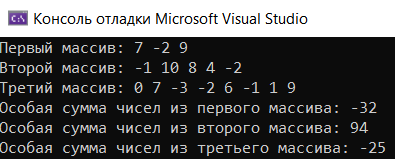
cout << "Особая сумма чисел из первого массива: " << fsum(arr1) << endl;

cout << "Особая сумма чисел из второго массива: " << fsum(arr2) << endl;

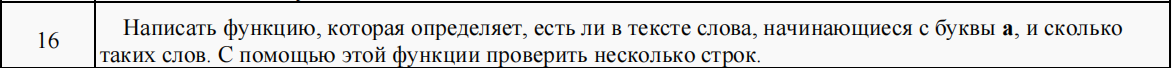
cout << "Особая сумма чисел из третьего массива: " << fsum(arr3);

return 0;

}



Вариант 16



#include <iostream> // подключение библиотеки для работы с вводом/выводом данных

#include <cctype> // подключение библиотеки для работы с символами

using namespace std; // подключение стандартного пространства имен

// прототип функции

int countA(char\* str, ...);

// функции для подсчета букв a

int countA(char\* str, ...) {

char \*ptr = str; // указатель на символы в строке

int counterA = 0; // добавляем счетчик слов, начинающихся с "a"

for (int i = 0; i <= strlen(str); i++) {

if (i == 0) { // проверяем первый символ

if (tolower(\*(ptr + i)) == 'a') {

counterA++;

}

}

if (\*(ptr + i) == ' ') { //проверяем символ после пробела

if (tolower(\*(ptr + i + 1)) == 'a') {

counterA++;

}

}

}

return counterA;

}

// Главная функция

int main()

{

setlocale(LC\_ALL, "RUS");

char str1[] = "Apple Jack attracts Amelia"; // строка 1

char str2[] = "Capybara forever A a"; // строка 2

char str3[] = "A lot of different animals"; // строка 3

// вывод соответствующих предложений

cout << "В предложении " << str1 << " количество слов, начинающихся с 'a': " << countA(str1) << endl;

cout << "В предложении " << str2 << " количество слов, начинающихся с 'a': " << countA(str2) << endl;

cout << "В предложении " << str3 << " количество слов, начинающихся с 'a': " << countA(str3);

return 0;

}

