Учебный курс: Технологии программирования

Кафедра: ИС Факультет: ИТиП

Лабораторная работа №1. Пространства имен

Выполнил: Трофимов В.А. Группа: 1511

Преподаватель: Повышев В.В.

# Задание

Согласно варианту описать указанные типы данных (Матрица 3x3) и поместить их в отдельный заголовочный файл. Аналогично поступить с функциями (Вывод матрицы на экран. Перемножение двух матриц. Умножение матрицы на число). Реализацию функций выполнить в отдельном файле.

Типы данных и функции разместить в соответствующем пространстве имен.

Написать программу, использующую описанные типы данных: инициализация переменных (ввод пользователя), выполнение действий (в зависимости от дальнейшего ввода пользователя).

Примечания.

Для всех вариантов описать и реализовать функцию print, которая бы выводила параметры объекта на экран (например для шара: координаты его центра и радиус).

Если для выполнения задания удобно определить дополнительные типы данных и функции – их, естественно, можно и нужно определять.

Все определения (как основные так и дополнительные) следует размещать в двух разных заголовочных файлах (один для типов данных, второй – для функций). Реализацию функций ­­– разместить в отдельном cpp файле. Текст программы (функция main), проверяющий работу разработанный функций, также следует разместить в отдельном файле. Итого – 4 файла.

В заголовочных файлах используйте namespace. В cpp файлах либо разрешайте пространство имен с помощью :: либо же используйте using namespace.

# Входные/Выходные данные

## Консоль

>>0 1 2 3 4 5 6 7 8

>>9 5 1 7 5 3 8 2 1

23.000000 9.000000 5.000000

95.000000 45.000000 20.000000

167.000000 81.000000 35.000000

46.000000 18.000000 10.000000

190.000000 90.000000 40.000000

334.000000 162.000000 70.000000

# Исходный текст

## Файл main.cpp

// Course: Programming technologies (C Plus Plus)

// Lab 1. Namespaces.

// Student: Trofimov V.A. Group: 1511

// Teacher: Povishev V.V.

// Created 21.02.2013 Modified: 22.02.2013

// Description: Program entry point.

#include "functions.h"

using namespace matrix;

int main() {

mtrx a, b, c;

read(a);

read(b);

c = mul(a, b);

print(c);

c = mul(c, 2);

print(c);

return 0;

}

## Файл declarations.h

// Course: Programming technologies (C Plus Plus)

// Lab 1. Namespaces.

// Student: Trofimov V.A. Group: 1511

// Teacher: Povishev V.V.

// Created 21.02.2013 Modified: 22.02.2013

// Description: Declarations header file.

#pragma once

namespace matrix {

const int size = 3;

struct mtrx {

double data[size][size];

};

}

## Файл functions.h

// Course: Programming technologies (C Plus Plus)

// Lab 1. Namespaces.

// Student: Trofimov V.A. Group: 1511

// Teacher: Povishev V.V.

// Created 21.02.2013 Modified: 22.02.2013

// Description: Functions header file.

#pragma once

#include "declarations.h"

namespace matrix {

mtrx fill(mtrx & a, double value);

mtrx cpy(const mtrx & a);

mtrx mul(const mtrx & a, double b);

mtrx mul(const mtrx & a, const mtrx & b);

int print(const mtrx & a);

int read(mtrx & a);

}

## Файл functions.cpp

// Course: Programming technologies (C Plus Plus)

// Lab 1. Namespaces.

// Student: Trofimov V.A. Group: 1511

// Teacher: Povishev V.V.

// Created 21.02.2013 Modified: 22.02.2013

// Description: Functions source code file.

#include "functions.h"

#include <stdio.h>

using namespace matrix;

mtrx matrix::fill(mtrx & a, double value) {

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

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

a.data[i][j] = value;

return a;

}

mtrx matrix::cpy(const mtrx & a) {

mtrx temp;

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

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

temp.data[i][j] = a.data[i][j];

return temp;

}

mtrx matrix::mul(const mtrx & a, double b) {

mtrx temp = cpy(a);

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

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

temp.data[i][j] \*= b;

return temp;

}

mtrx matrix::mul(const mtrx & a, const mtrx & b) {

mtrx temp; fill(temp, 0);

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

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

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

temp.data[i][j] += a.data[i][k] \* b.data[k][j];

return temp;

}

int matrix::print(const mtrx & a) {

return printf("\n%lf %lf %lf\n%lf %lf %lf\n%lf %lf %lf\n\n",

a.data[0][0], a.data[0][1], a.data[0][2],

a.data[1][0], a.data[1][1], a.data[1][2],

a.data[2][0], a.data[2][1], a.data[2][2]);

}

int matrix::read(mtrx & a) {

return scanf("%lf %lf %lf %lf %lf %lf %lf %lf %lf",

&a.data[0][0], &a.data[0][1], &a.data[0][2],

&a.data[1][0], &a.data[1][1], &a.data[1][2],

&a.data[2][0], &a.data[2][1], &a.data[2][2]);

}