https://github.com/walkraw123345/labs.git

1.

//Описать функцию PowerA3(A, B), вычисляющую третью степень числа A и возвращающую ее в переменной B (A — входной, B — выходной параметр; оба параметра являются вещественными). С помощью этой функции найти третьи степени пяти данных чисел.

#include <stdio.h>

void PowerA3(float A, float\* B) //Создаем функцию для вычисления 3 степени числа a

{

\*B = A \* A \* A;

}

int main()

{ // Создаем цикл для рассчета третьей степени для 5 чисел

for (int i = 1; i <= 5; ++i) {

float A;

printf("A:");

scanf\_s("%f", &A);

float B;

PowerA3(A, &B);

printf\_s("B: %f\n", B); // Выводим результат

}

return 0;

}  
  
2.

//Описать функцию Sign(X) целого типа, возвращающую для вещественного числа X следующие значения:

−1, если X < 0; 0, если X = 0; 1, если X > 0.

С помощью этой функции найти значение выражения Sign(A) + Sign(B) для данных вещественных чисел A и B.

#include <stdio.h>

int sign(float x) // Создаем функцию, удовлетворяющюю условиям задания

{

if (x < 0)

return -1;

if (x == 0)

return 0;

if (x > 0)

return 1;

}

int main(void)

{

float a, b; // Объявляем переменные и ввод для пользователя

printf("A:");

scanf\_s("%f", &a);

printf("B:");

scanf\_s("%f", &b);

printf("Sign(A)+Sign(B):%i\n", sign(a) + sign(b)); // Выводим результат

return 0;

}

3.

//Описать функцию RingS(R1, R2) вещественного типа, находящую площадь кольца, заключенного между двумя окружностями с общим центром и радиусами R1 и R2 (R1 и R2 — вещественные, R1 > R2). С ее помощью найти площади трех колец, для которых даны внешние и внутренние радиусы.

#include <stdio.h>

float rings(float r1, float r2) { // Функция для вычисления площади кольца

return 3.14 \* (r1 \* r1 - r2 \* r2);

}

int main(void)

{

int i;

for (i = 1; i <= 3; ++i) { // Создаем цикл для расчета площади 3 различных колец

float r1, r2; // Объявляем переменные и реализуем ввод для пользователя

printf("R1:");

scanf("%f", &r1);

printf("R2:");

scanf("%f", &r2);

printf("S:%f\n", rings(r1, r2)); // Выводим результат

}

return 0;

}

4.

//Описать функцию Quarter(x, y) целого типа, определяющую номер координатной четверти, в которой находится точка с ненулевыми вещественными координатами (x, y). С помощью этой функции найти номера координатных четвертей для трех точек с данными ненулевыми координатами

#include <stdio.h>

int quarter(float x, float y) { // Создаем функцию для определения номера координатной четверти

if ((x > 0) && (y > 0))

return 1;

if ((x < 0) && (y > 0))

return 2;

if ((x < 0) && (y < 0))

return 3;

if ((x > 0) && (y < 0))

return 4;

}

int main(void)

{

int i;

for (i = 1; i <= 3; ++i) { // Создаем цикл для определения номера координатной четверти три раза.

float x, y;

printf("X:");

scanf("%f", &x);

printf("Y:");

scanf("%f", &y);

printf("Quarter:%i\n", quarter(x, y));

}

return 0;

}

5.

//Описать функцию Fact2(N) вещественного типа, вычисляющую двойной факториал:

N!! = 1·3·5·. . .·N, если N — нечетное;

N!! = 2·4·6·. . .·N, если N — четное (N > 0 — параметр целого типа; вещественное возвращаемое значение используется для того, чтобы избежать целочисленного переполнения при больших значениях N).

#include <iostream>

float Fact2(unsigned int n) { // Создаем функцию по заданным условиям

int fact\_double = 1;

if (n % 2 == 0) {

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

fact\_double \*= i;

}

else {

for (int i = 1; i <= n; i += 2)

fact\_double \*= i;

}

return fact\_double;

}

int main() {

int n;

std::cin >> n;

std::cout << Fact2(n); // Вызываем функцию и выводим результат.

return 0;

}