**Лабораторна робота №2**

***Типи даних. Арифметичні вирази. Програмування лінійних алгоритмів***

***Мета:*** ознайомитися з поняттям і структурою мови програмування Сі, з величинами простих типів, їх властивостями та операціями над ними.

**Хід роботи**

**Завдання 1.** Перевести числа із звичайного виду у формат з плаваючою комою:

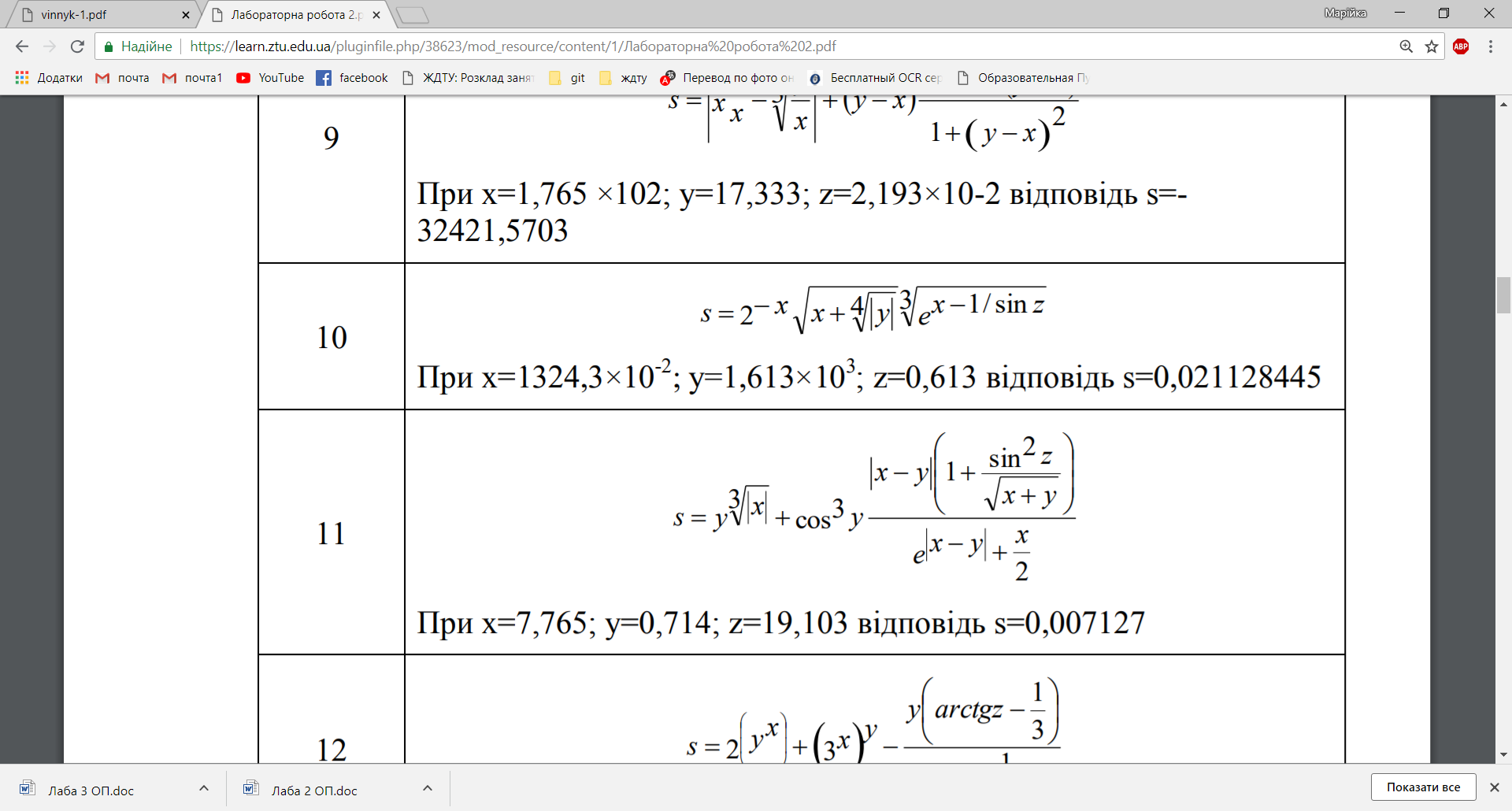
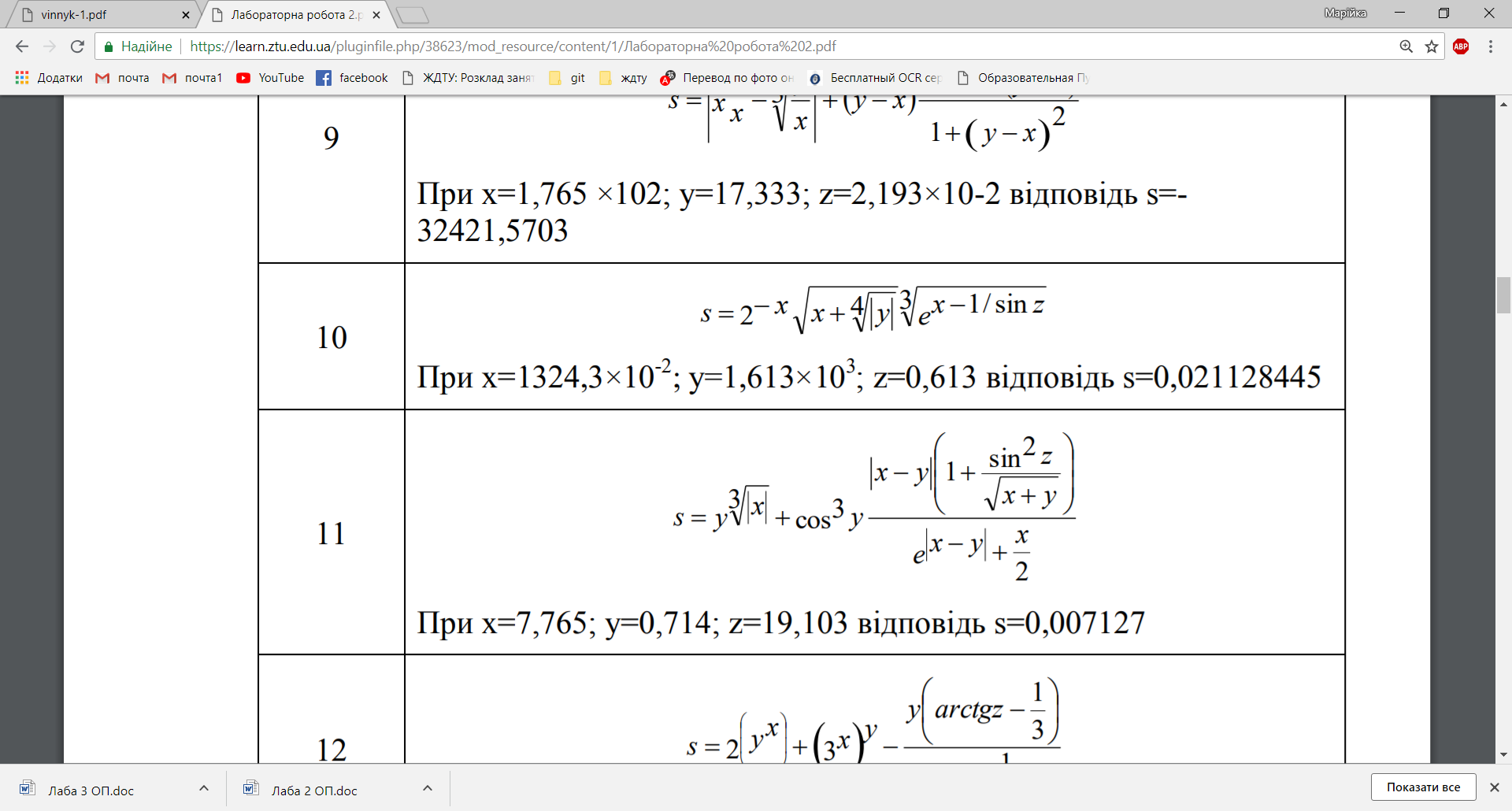
1. 0.0000007 = 7E-7
2. 180000000 = 1.8E+8
3. 0.0000187 = 1.87E-5
4. 5\*10-18 = 5E-18
5. 0.1\*10-10 = 1E-11
6. 1.4\*10+18 = 1.4E+18

Привести числа до нормального виду

1. 0.1E+6 = 1\*105
2. 1.87E-18 = 1.87\*10-18
3. 17E+3 = 1.7\*104
4. 0.14E-8 = 1.4\*10-9
5. 11E+4 = 1.1\*105
6. 3E-14 = 3\*10-14

Знайдіть помилку і поясніть:

1. 15-E6 (або мінус або Е не на правильному місці) 15 E-6 / E 6-15
2. 0.1Е-8 (число 0.1 виходить за діапазон)
3. сos3 (відсутні дужки для аргументу) сos(3)
4. E-5 (відсутнє число перед записом) ?E-5
5. 1.3E+39.1 (число 39.1 записано неправильно)
6. +E-12 (відсутнє число перед записом) ?E-12
7. 7E (відсутній ступінь) 7E??
8. E+9 (відсутнє число) ?E+9
9. 123E0 (відсутній знак, число 123 виходить за діапазон) 123E?0

**Завдання 2**. Написати програму для обрахунку значення виразу при заданих вхідних даних. Отриманий результат порівняти з правильною відповіддю.

**10**

Рис. 1. Заданий вираз та значення для обрахунку програми.

Лістинг програми:

#include "pch.h"

#include "stdio.h"

#include <math.h>

int main()

{

float x, y, z, s; //Обязательные переменные

float a, b, c; //Дополнительные переменные

//x = 13.243

//y = 1613

//z = 0.613

//s должна иметь значение 0.021128445

printf("x=");

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

printf("y=");

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

printf("z=");

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

a = 1.0/pow(2, x);

b = sqrt(x + pow(fabs(y), 1 / 4.0));

c = pow(exp((x - 1.0) / sin(z)), 1 / 3.0);

s = a \* b \* c;

printf("Result s = %.9f", s);

printf("\n\n\n");

return 0;

}

Результат виконання програми:

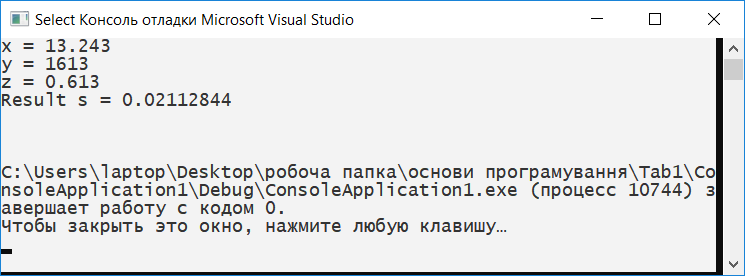


Рис. 2. Результат виконання програми

**Завдання 3**: Напишіть програму згідно Вашого індивідуального завдання. Всі вхідні данні є дійсними:

Дано змінні A, B, C, D. Змінити місцями зміст змінних А і D, D і С, A i В та вивести результат на екран. !!! Додаткові змінні не використовувати.

Дано значення кута в радіанах (0<α<2π). Визначити значення кута у градусах, якщо 180 градусів = π радіанів.

Лістинг програми:

#include"pch.h"

#include<stdio.h>

#include<math.h>

#include<locale.h>

int main()

{

setlocale(LC\_ALL, "Rus");

int A, B, C, D; //A=2; B=5; C=4; D=10;//

printf("A="); scanf\_s("%d", &A);

printf("B="); scanf\_s("%d", &B);

printf("C="); scanf\_s("%d", &C);

printf("D="); scanf\_s("%d", &D);

A = A + D;

D = A - D;

A = A - D;

printf("\nA=%d, D=%d\n", A, D);

D = D + C;

C = D - C;

D = D - C;

printf("D=%d, C=%d\n", D, C);

A = A + B;

B = A - B;

A = A - B;

printf("A=%d, B=%d\n", A, B);

printf("\n");

if (0 < A, B, C, D < 360)

{

printf("A=%lf радиан B=%lf радиан C=%lf радиан D=%lf радиан", A / 57.3, B / 57.3, C / 57.3, D / 57.3);

}

else

{

printf("Введите число в пределах от 0 до 360");

}

printf("\n");

return 0;

}

Результат виконання програми:

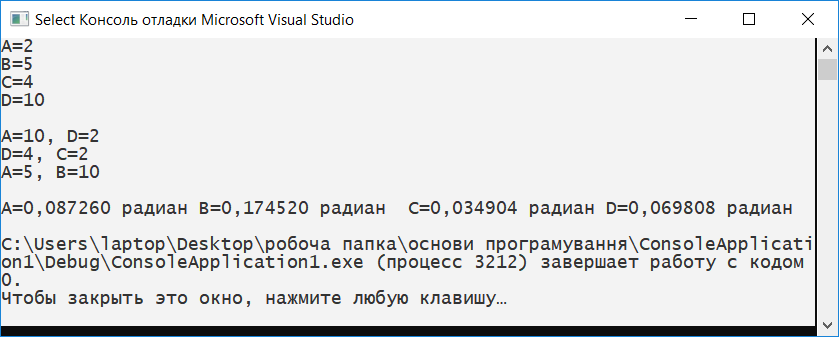


Рис. 3. Результат виконання програми

*Дано значення кута в радіанах (0<α<2π). Визначити значення кута у градусах, якщо 180 градусів = π радіанів*.

Лістинг програми:

/\*Програма запитує у користувача значення кута у радіанах

переводить кут у градуси та друкує значення\*/

#include"pch.h"

#include "stdio.h"

#include<math.h>

int main()

{

float rad, deg;

printf("\n\n Vvedite Rad: ");

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

deg = rad \* (180 / 3.1415);

printf("Result: %.4lf\n\n", deg);

return 0;

}

Результат виконання програми:

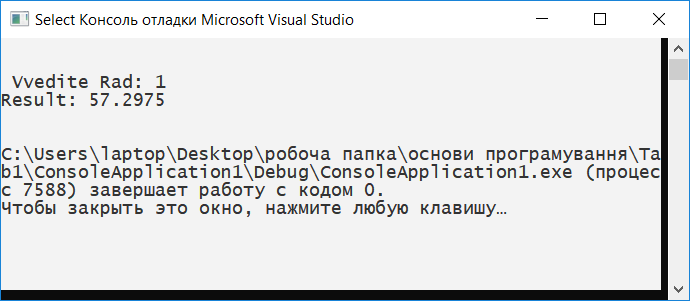


Рис. 4. Результат виконання програми

**Завдання на самостійну роботу:**

Напишіть програму згідно Вашого індивідуального завдання. Дано довільний трикутник ABC, для якого визначений наступний набір параметрів: а, b, с - сторони трикутника; α, β, γ - кути (у градусах); h - висота, опущена на сторону с; S - площа; Р - периметр трикутника. По трьом заданим параметрам обчислити всі інші.

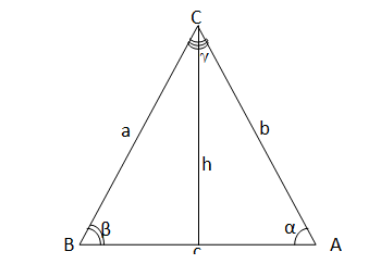


Рис 5. Візуалізація умови до завдання для самостійної роботи

|  |  |  |  |
| --- | --- | --- | --- |
| 10 | c, α, β | S, h, β | a, b, S |

Рис 6. Варіанти заданих величин до завдання для самостійної роботи.

Лістинг програми:

#include "pch.h"

#include <iostream>

#include<math.h>

#include<locale.h>

#define PI 3.14159265358979323846

int main()

{

setlocale(LC\_ALL, "Rus");

double a, b, c, h, P, S, alpha, betta, gamma;

int N;

printf("Введите номер задания:"); scanf\_s("%d", &N);

printf("\n\nДано:");

switch (N) {

case 1: {

printf("\nc= "); scanf\_s("%lf", &c);

printf("alpha= "); scanf\_s("%lf", &alpha);

printf("betta= "); scanf\_s("%lf", &betta);

printf("\n");

gamma = 180 - (alpha + betta);

a = c \* alpha / gamma;

b = c \* betta / gamma;

S = a \* b \* gamma / 2.0;

P = a + b + c;

h = 2.0 \* S / c;

if (P > a + b && a > 0 && b > 0 && c > 0 && a + b > c && c + a > b && c + b > a && a > h && b > h)

{

printf("\n\nРезультат:\na=%lf см, \nb=%lf см, \nS=%lf см2, \nP=%lf см, \nh=%lf см, \ngamma=%lf \n\n\n", a, b, S, P, h, gamma);

}

else

{

printf("Такой треугольник не существует\n\n\n");

}

break;

}

case 2:

{

printf("\nS= "); scanf\_s("%lf", &S);

printf("h= "); scanf\_s("%lf", &h);

printf("betta= "); scanf\_s("%lf", &betta);

c = 2 \* S / h;

b = sqrt(pow(h, 2) + pow(c, 2.0));

a = sqrt(pow(h, 2) + pow(c / 2, 2.0));

P = a + b + c;

alpha = h / b;

alpha = alpha \* 180.0 / PI;

gamma = 180 - (alpha + betta);

if (P > a + b && a > 0 && b > 0 && c > 0 && a + b > c && c + a > b && c + b > a && a > h && b > h)

{

printf("\n\nРезультат:\na=%lf см, \nb=%lf см, \nc=%lf см, \nalpha=%lf , \ngamma=%lf, \nP=%lf см \n\n\n ", a, b, c, P, alpha, gamma);

}

else {

printf("Такой треугольник не существует\n\n\n");

}

break;

}

case 3:

{

printf("\na= "); scanf\_s("%lf", &a);

printf("b= "); scanf\_s("%lf", &b);

printf("S= "); scanf\_s("%lf", &S);

gamma = 2 \* S / a \* b;

c = sqrt(pow(a, 2) + pow(b, 2) - 2 \* a\*b\*gamma);

h = 2 \* S / c;

P = a + b + c;

alpha = h / b;

alpha = alpha \* 180 / PI;

betta = 180 - (alpha + gamma);

betta = betta \* 180 / PI;

if (P > a + b && a > 0 && b > 0 && c > 0 && a + b > c && c + a > b && c + b > a && a > h && b > h)

{

printf("\n\nРезультат:\nc=%lf см, \nalpha=%lf см, \nbetta=%lf, \ngammma=%lf , \nP=%lf см, \nh=%lf см \n\n\n ", c, alpha, betta, gamma, P, h);

}

else {

printf("Такой треугольник не существует\n\n\n");

}

break;

}

}

system("pause");

return 0;

}

Результат виконання програми:

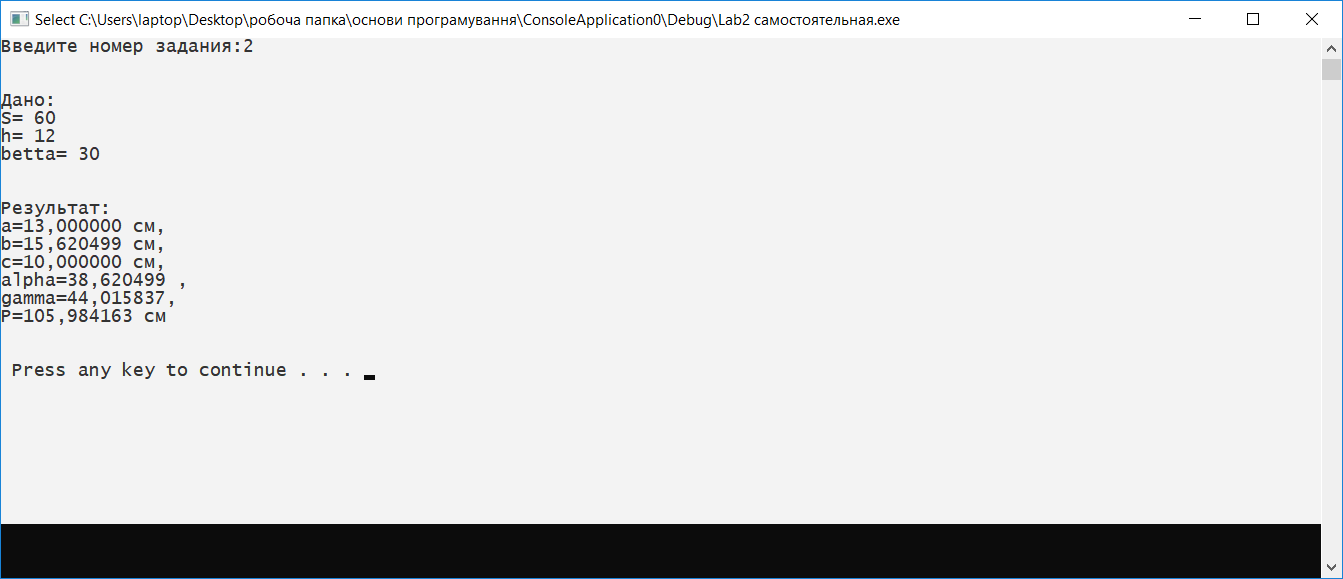


Рис. 7. Результат виконання програми

**Контрольні питання:**

***1. Які прості типи даних та їх модифікатори використовуються у***

***мові програмування Сі?***

Для вказівки простих типів зазначаються специфікатори int, char, float та double. До змінних можуть підставлятися модифікатор unsigned (беззнаковий), signed (знаковий), short, long, long long.

За замовчуванням усі числа є знаковими, відповідно, можуть знаходитися в діапазоні тільки позитивних чисел. Щоб визначити змінну типу char як знакову, пишеться signed char. Long, long long і short вказують, як багато місця в пам'яті відводиться для зберігання. Найбільше - long long, найменше - short. Char - найменший тип даних в мові Сі. Для зберігання значень виділяється всього 1 байт пам'яті. Змінної типу character зазвичай присвоюються символи, рідше - цифри. Символьні значення беруться в лапки. char a = "a"; char b = 2; Тип int зберігає цілі числа, його розмір не визначений займає до 4 байт пам'яті, в залежності від архітектури комп'ютера. int a = 12; Явне перетворення беззнаковой змінної задається так: unsigned int a = 12; Неявне виглядає так: int a = 12u; Float і double визначають числа з крапкою. Числа float представляються у вигляді -2.3 або 334. Double використовується для більшої точності - після роздільника цілої та дробової частини вказується більше цифр. Цей тип займає більше місця в пам'яті, ніж float.

double a = 124567; Void має пусте значення. Він визначає функції, які нічого не повертають. За допомогою цього специфікатора вказується пусте значення в аргументах методів. Покажчики, які можуть приймати будь-який тип даних, також визначаються як void.

***2. За якими правилами відбувається неявне перетворення типів***

***даних?***

Неявні перетворення типу виконуються головним чином для узгодження аргументів оператора або функції (якщо це можливо) зі значеннями, передбачуваними в цих операторах або функціях.

u при виконанні арифметичних операцій

u при виконанні операцій присвоювання

u при передачі аргументів у функцію

**В операціях присвоювання** тип значення, яке присвоюється, перетвориться до типу змінної, яка отримує це значення

**В арифметичних операціях:**

· Операнди типів **char** и **short** перетвориться до типу **int**, Операнди типу **float** перетвориться до типу **double**

· Якщо хоча б один з операндів має тип **double**, То і другий операнд перетвориться до типу **double**, Результат має тип **double**

· Якщо хоча б один операнд має тип **long**, То і другий операнд перетвориться до типу **long**, Результат має тип **long**

· Якщо хоча б один операнд має тип **unsigned**, То і другий операнд перетвориться до типу **unsigned**, Результат має тип **unsigned.**

**Перетворення при виконанні функції**: Якщо заданий прототип функції і він включає оголошення типів аргументів, то над аргументами у виклику функції виконуються тільки звичайні арифметичні перетворення

***3. Як здійснюється форматований ввід та вивід даних?***

У мові C++ дії, що пов’язані з операціями введення і виве­дення, виконуються за допомогою функцій бібліотек. Функції і ведення і виведення бібліотек мови дозволяють читати дані з файлів та пристроїв і писати дані у файли і на пристрої.

Бібліотека мови C++ підтримує **три рівня введення-виведення даних:**

□     введення-виведення потоку;

□     введення-виведення нижнього рівня;

□     введення-виведення для консолі і порту.

При введенні-виведенні потоку всі дані розглядаються як потік окремих байтів. Для користувача потік — це файл на дис­ку або фізичний пристрій, наприклад, дисплей чи клавіатура, або пристрій для друку, з якого чи на який направляється потік даних. Операції введення-виведення для потоку дозволяють об­робляти дані різних розмірів і форматів від одиночного символу до великих структур даних. Програміст може використовувати функції бібліотеки, розробляти власні і включати їх у бібліоте­ку. Для доступу до бібліотеки цих класів треба включити в програму відповідні заголовні файли.

За замовчуванням стандартні введення і виведення повідомлень про помилки відносяться до консолі користувача (клавіатури та екрана). Це означає, що завжди, коли програма очікує введення зі стандартного потоку, дані повинні надходити з клавіатури, а якщо програма виводить дані — то на екран.

***Висновки:*** в ході виконання лабораторної роботи було ознайомлено з поняттям і структурою мови програмування Сі, з величинами простих типів, їх властивостями та операціями над ними. Навчено переводити числа із звичайного виду у формат з плаваючою комою та навпаки; Навчено писати програму для обрахунку значення виразу при заданих вхідних даних. Навчено писати програму для обрахунку значення виразу при заданих вхідних даних.