**ПРАКТИЧНЕ ЗАВДАННЯ № 2**

**Завдання № 1:**

Напишите программу для ввода количества отработанных персоналом часов и размера почасовой оплаты каждого. Затем выведите суммарную зарплату персонала. (Удостоверьтесь в правильности ввода.)

**Код програми:**

#include <iostream>

#include <cassert>

using namespace std;

int main(){

int hours, wage;

cout << "Enter hours & wage" <<endl;

cin >> hours >> wage;

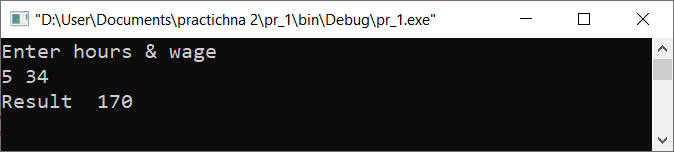
assert(hours > 0); assert( wage > 0);

cout << "Result " << hours \* wage << endl;

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання № 2:**

Напишите программу для преобразования метров в футы и футов в дюймы. Организуйте ввод числа метров и вывод на экран соответствующего числа футов и дюймов. Повторяйте эту процедуру до тех пор, пока пользователь не введет 0 в качестве числа метров.

**Код програми:**

#include <iostream>

using namespace std;

#define feet 3.280839895

#define inch 39.37007874

//enum { feet = 3.280839895, inch = 39.37007874 };

int main() {

double meters;

do {

cin >> meters;

cout << "feets " << meters \* feet << endl;

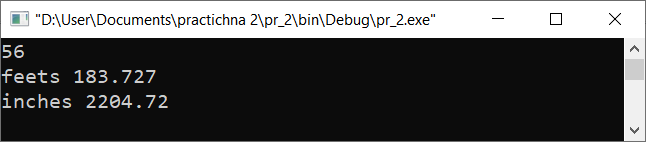
cout << "inches " << meters \* inch << endl;

} while(meters != 0);

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання № 3:**

Ниже приведена программа на языке С. Перепишите ее в соответствии со стилем ввода/вывода C++. /\* Преобразуйте эту программу на С в соответствии со стилем программирования C++. Эта программа подсчитывает наименьшее общее кратное

**Код програми:**

#include <iostream>

using namespace std;

int main() {

int a, b, i, min;

cout << "Enter 2 numbers" << endl;

cin >> a >> b;

min = a > b ? b : a;

for (i = 2; i < min; ++i) {

if (a % i == 0 && b % i == 0)

break ;

if (i == min) {

cout << "No common denominators\n";

return 0;

}

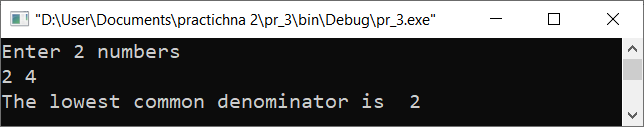
}

cout << "The lowest common denominator is " << i << endl;

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання № 4:**

Создайте класс card, который поддерживает каталог библиотечных карточек. Этот класс должен хранить заглавие книги, имя автора и выданное на руки число экземпляров книги. Заглавие и имя автора храните в виде строки символов, а количество экземпляров - в виде целого числа. Используйте открытую функцию-член store() для запоминания информации о книгах и открытую функцию-член show() для вывода информации на экран. В функцию main() включите краткую демонстрацию работы созданного класса

**Код програми:**

#include <iostream>

#include <cstring>

typedef const char cchar;

using namespace std;

class card {

private:

int copies;

char name[100];

char author[100];

public:

void get\_data(cchar\* title, cchar\* men, int num);

void show();

};

void card::get\_data(cchar\* title, cchar\* men, int num) {

copies = num;

strcpy(name, title);

strcpy(author, men);

}

void card::show() {

cout << name << ' ' << author << ' ';

cout << copies << endl;

}

main() {

card book1, book2;

book1.get\_data("The Da Vinci Code", "Dan Brown", 14);

book2.get\_data("The Three Musketeers", "Alexandre Dumas", 25);

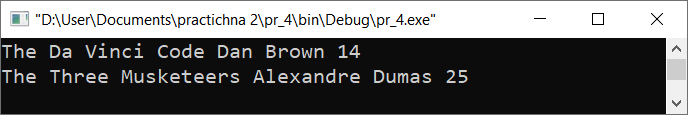
book1.show();

book2.show();

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання № 5:**

Создайте класс с циклической очередью целых (см. Unit 2 "A more practical example"). Сделайте очередь длиной 100 целых. В функцию main() включите краткую демонстрацию ее работы. Провести инициализацию с помощью конструктора.

**Код програми**

**Завдання № 6:**

Реализуйте на С решение Задачи 2.5. (см. Unit 1 "A singly-linked list"). Пример циклической очереди, построенной при помощи указателей:

**Код програми:**

#include <stdio.h>

#include <stdlib.h>

struct node {

int data;

struct node \*next;

};

struct node \*head = NULL;

void addlist(struct node\*\* new\_node) {

struct node\* temp;

if (head == NULL) {

head = \*new\_node;

head->next = head;

}

else {

temp = head;

while (temp->next != head)

temp = temp->next;

temp->next = \*new\_node;

temp = \*new\_node;

temp->next = head;

}

}

void disp(int size) {

struct node\* temp;

printf("%3d\n", head->data);

temp = head->next;

for (int i = 0; i < size - 1; ++i) {

printf("%3d\n", temp->data);

temp = temp->next;

}

}

int main() {

struct node\* new\_node;

int size, num;

printf("Enter size, number of elements in queue\n");

scanf("%d %d", &size, &num);

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

new\_node = (struct node\*)malloc(sizeof(struct node));

// new\_node -> data = i;

new\_node->data = rand() % 64;

addlist(&new\_node);

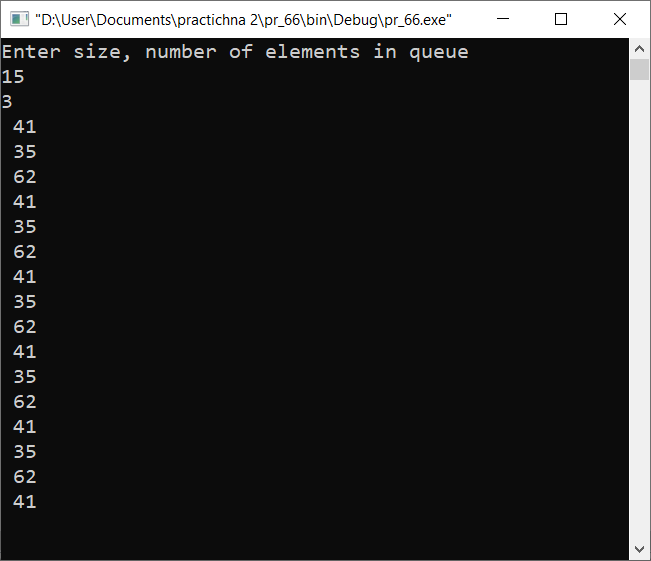
}

disp(size);

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання № 7:**

Объясните, почему работает такой код:

**Код програми:**

#include <cstdlib>

#include <iostream>

using namespace std;

int main(int argc, char\*\* argv) {

cout << argv[argc - 1] << endl << argc[argv - 1] << endl;

return EXIT\_SUCCESS;

}

//argv и argc - это встороеные аргументы функции main предназначеные для

//получения аргументов командной строки

//аргумент argc - это количество аргументов командной строки

//имя програмы - это первый аргумент создержащийся в argc

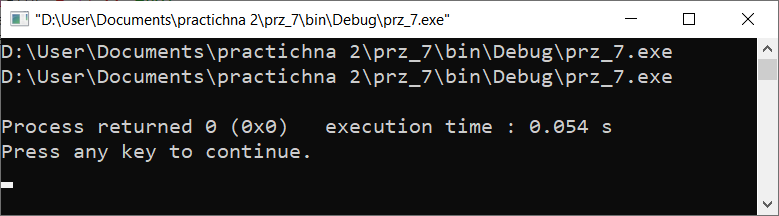
//потому если argc < 1 то выводиться имя программы

//argv - указатель на массив символьных указателейб указателей на на аргументы

//командной строкию Таким образом argv[0] - указатель на первую строку,

// на имя программы

**Результати роботи програми (скриншоти):**



**Завдання № 8:**

Создайте функцию myroot(), которая возвращает квадратный корень своего аргумента. Перегрузите myroot() тремя способами (см. Unit 2 "Introducing Function Overloading"): чтобы получить квадратный корень целого, длинного целого и числа с плавающей точкой двойной точности. (Для непосредственного подсчета квадратного корня вы можете использовать стандартную библиотечную функцию sqrt().

**Код програми:**

#include <iostream>

#include <cassert>

#include <cmath>

using namespace std;

int myroot(int a);

double myroot(double a);

long int myroot(long int a);

int main() {

int num1;

double num2;

long int num3;

cin >> num1 >> num2 >> num3;

assert(num1 > 0 && num2 > 0 && num3 > 0);

cout << myroot(num1) << ' ' << myroot(num2) << ' ' << myroot(num3);

return 0;

}

int myroot(int num) {

return sqrt(num);

}

double myroot(double num){

return sqrt(num);

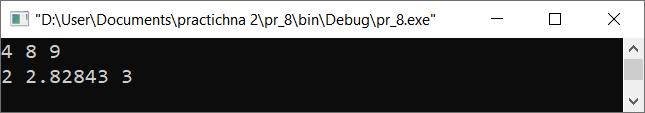
}

long int myroot(long int num){

return sqrt(num);

}

**Результати роботи програми (скриншоти):**



**Завдання № 9:**

Стандартная библиотека C++ содержит три функции:

double atof (const char \*s);

int atoi (const char \*s);

long atol(const char \*s);

Эти функции возвращают численное значение, содержащееся в строке, на которую указывает s. Заметьте, что atof() возвращает double, atoi возвращает int и atol возвращает long. Почему нельзя перегрузить эти функции? Продемонстрируйте пример кода.

**Код програми:**

#include <iostream>

#include <cstdlib>

using namespace std;

double atof(const char \*s);

int atoi(const char \*s);

long int atol(const char \*s);

int main() {

char res[256], res1[256], res2[256];

cout << atof("243.345 radom words") << endl;

cout << atoi("435Hi hello bye") << endl;

cout << atol("384758689 very long number kek") << endl;

/\* cin >> res;

cout << atof(res) << endl;

cin >> res;

cout << atof(res) << endl;

cin >> res1;

cout <<atoi(res1) << endl;

cin >> res2;

cout << atol(res2) << endl;\*/

return 0;

}

/\*Функция atoi(const char \*s); превращает строку содержащую целое число в тип int, если

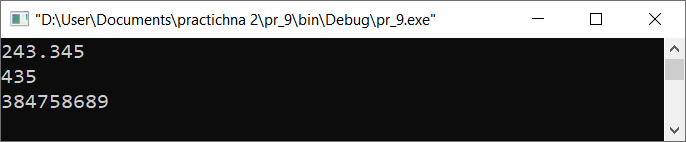
строка не содержит числа типа Int то функция возвражает непределеное значение

Функция double atof(const char \*s); превращает строку начинающуюся на число с плавающей точкой в тип double. Если число заканчиваеться точкой или символом е E то функция возвражает непределеное значение.

Функция atol(const char \*s); делает то же, что и предыдущые, но для чисел типа long при этом число может завершаться любим символом (если atol(“345.445”), то возвращает 345), если строка содержит не коректное предствление числа, то функция возвраащет 0;

Предполагаю, что перезагрузка невозможна из-за того, что для роботы функций строка должна содержать коректное представление числа, которое для каждой функции отличаеться.\*/

**Результати роботи програми (скриншоти):**



**Завдання № 10:**

Создайте свою функцию min(), которая возвращает наименьший из двух численных аргументов, используемых при ее вызове. Перегрузите функцию min() так, чтобы она воспринимала в качестве аргументов символы, целые и действительные двойной точности (см. Unit 2 "Introducing Function Overloading").

**Код програми:**

#include <iostream>

using namespace std;

int min(int a, int b) {

return a > b ? b : a;

}

double min(double a, double b){

return a > b ? b : a;

}

char min(char a, char b){

return a > b ? b : a;

}

int main() {

cout << min(2, 45) << endl;

cout << min (345.345, 2.34) << endl;

cout << min('a', 'c') << endl;

return 0;

}

#include <iostream>

using namespace std;

int min(int a, int b);

double min(double a, double b);

char min(char a, char b);

int main() {

int a, b;

double c, d;

char e, f;

cin >> a >> b >> c >> d >> e >> f;

cout << min(a, b) << ' ' << min(c, d) << ' ' << min(e, f)<<endl;

return 0;

}

int min(int a, int b) {

return a > b ? b : a;

}

double min(double a, double b){

return a > b ? b : a;

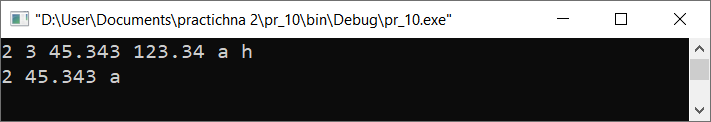
}

char min(char a, char b){

return a > b ? b : a;

}

**Результати роботи програми (скриншоти):**



**Завдання № 11:**

Создайте свою функцию sleep(), приостанавливающую работу компьютера на столько секунд, сколько указано в аргументе функции. Перегрузите sleep() так, чтобы она могла вызываться или с целым, или со строкой, задающей целое. Например, оба этих вызова должны заставить компьютер остановиться на 10 секунд: sleep(10); sleep("10"); Продемонстрируйте работу ваших функций, включив их в короткую программу.

**Код програми:**

#include <iostream>

#include <ctime>

#include <windows.h>

using namespace std;

void sleep(const size\_t seconds) {

clock\_t start = clock();

while((double)(clock() - start)/CLOCKS\_PER\_SEC < seconds);

}

void sleep(const char \* seconds) {

clock\_t start = clock();

while((double)(clock() - start)/CLOCKS\_PER\_SEC < atoi(seconds));

}

int main() {

srand(time(NULL));

int num;

cin >> num;

for (int i = 0; i < 10; ++i) {

cout << rand() % 64 << endl;

if (i == 4) {

cout << "\a" << endl;

sleep(num);

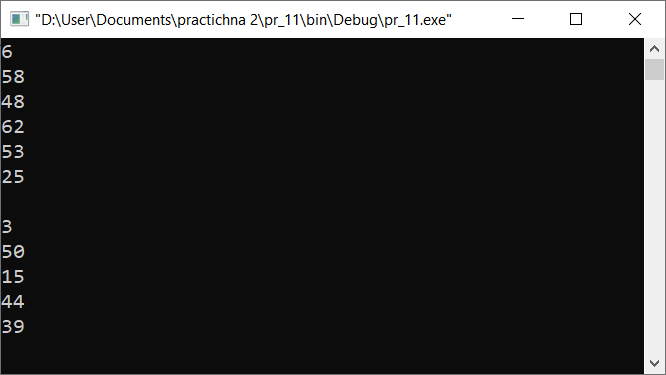
}

}

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання № 12:**

Напишите программу, использующую стиль ввода/вывода C++, для ввода двух целых с клавиатуры и затем вывода на экран результата возведения первого в степень второго. (Например, пользователь вводит 2 и 4, тогда результатом будет 24, или 16.)

**Код програми:**

#include <iostream>

#include <cmath>

using namespace std;

int pow\_ab(int a, int b){

return pow(a, b);

}

/\*

int pow\_ba(int a, int b){

return pow(b, a);

}\*/

int main() {

int a, b;

cin >> a >> b;

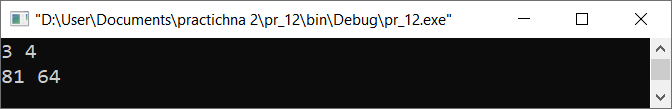
cout << pow\_ab(a, b) << ' ' << pow\_ab(b,a) << endl;

// cout << pow\_ab(a, b) << pow\_ba(a, b) << pow\_ab(b,a) << endl;

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання № 13:**

Создайте функцию rev\_str() для изменения порядка следования символов строки на обратный. Перегрузите rev\_str() так, чтобы она могла вызываться с одним или двумя символьными строками. Если функция вызывается с одной строкой, то операция должна осуществляться с ней. Если она вызывается с двумя строками, то результирующая строка должна оказаться во втором аргументе (см. Unit 2 "Overloaded functions that differ in the number of their arguments").

**Код програми:**

#include <iostream>

#include <cstring>

#define SIZE 100

using namespace std;

void reverse\_string(char \*str1, char \*str2){

for(int i = 0, j = strlen(str1); i < j; ++i)

str2[i] = str1[j - i - 1];

cout << str2 << endl;

}

void reverse\_string(char \*str1){

char str3[SIZE];

for(int i = 0, j = strlen(str1); i < j; ++i)

str3[i] = str1[j - i - 1];

strncpy(str1, str3, strlen(str3));

cout << str3 << endl;

}

/\*

void reverse\_1(char \*str){

int i, j;

for(i = 0, j = strlen(str) - 1; i < j; ++i, --j)

swap(str[i], str[j]);

cout << "reverse\_1 " << str << endl;

}

\*/

int main() {

char str1[SIZE], str2[SIZE];

cin >> str1;

reverse\_string(str1);

//reverse\_1(str1);

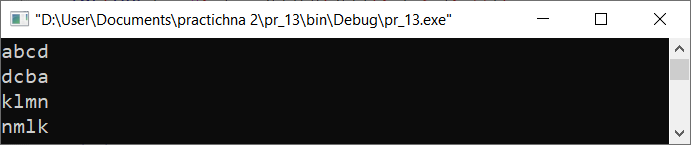
cin >> str1;

reverse\_string(str1, str2);

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання № 14:**

Создайте класс stopwatch для имитации секундомера. Используйте конструктор для начальной установки секундомера в 0. Образуйте две функции-члена start() и stop() соответственно для запуска и остановки секундомера. Включите в класс и функцию-член show() для вывода на экран величины истекшего промежутка времени. Также используйте деструктор для автоматического вывода на экран времени, прошедшего с момента создания объекта класса stopwatch , до его удаления. (Для простоты время приведите в секундах.)

**Код програми:**

#include <iostream>

#include <ctime>

#include <windows.h>

using namespace std;

class watch{

clock\_t start\_p;

clock\_t start\_time;

public:

watch();

~watch();

void start();

void show();

};

watch::watch(){

start\_p = clock();

}

void watch::start(){

start\_time = clock();

}

void watch::show(){

cout << (double)(clock() - start\_time)/CLOCKS\_PER\_SEC << endl;

}

watch::~watch(){

cout << (double)(clock() - start\_p)/CLOCKS\_PER\_SEC << endl;

}

int main(){

srand(time(NULL));

watch timer;

int num;

cin >> num;

timer.start();

for (int i = 0; i < 10; ++i) {

cout << rand() % 64 << ' ';

if (i == 4){

Sleep(num \* 1000);

puts("\n");

}

}

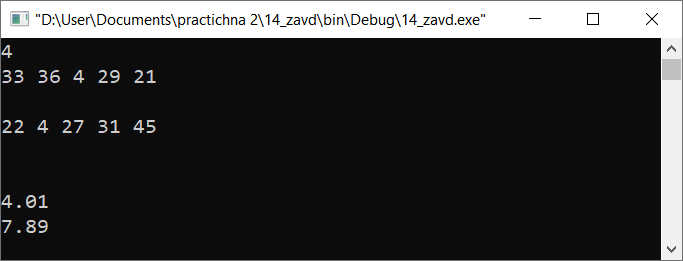
puts("\n\n");

timer.show();

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання № 15:**

Создайте новый вариант разработанного ранее класса strtype, в котором используется конструктор с параметром (см . Unit 2 "An Example that Shows the Need for Both a Constructor and a Destructor Function"). В вашей версии класса strtype строка получает свое начальное значение с помощью конструктора.

**Код програми:**

#include <iostream>

#include <cstring>

#include <cstdlib>

using namespace std;

class strtype {

char \*p;

int len;

public:

strtype(const char \*ptr);

~strtype();

void show();

};

strtype::strtype(const char \*ptr) {

len = strlen(ptr);

p = (char \*)malloc(strlen(ptr) + 1);

if(!p)

exit(1);

strcpy(p, ptr);

}

strtype::~strtype(){

free(p);

}

void strtype::show(){

cout << p <<" lenght " << len << endl;

}

int main(){

strtype s1("ABC");

strtype s2("ABCDEFG");

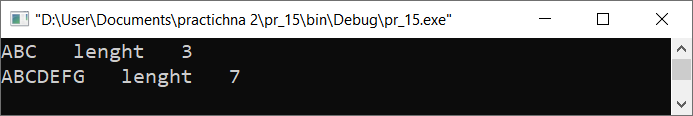
s1.show();

s2.show();

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання № 16:**

Измените класс stack (см . Unit 2 "An Improved Version of the Stack class") так, чтобы память для стека выделялась динамически. При этом длина стека должна задаваться параметром конструктора. (Не забудьте освободить эту память с помощью деструктора!)

**Код програми:**

#include <iostream>

using namespace std;

class stack{

int size;

char \*stack\_1;

public:

stack();

~stack();

void push(char name);

char pop();

};

stack::~stack(){

delete[] stack\_1;

}

stack::stack(){

size = 16;

stack\_1 = new char[size];

}

char stack::pop(){

//if (size == 0)

--size;

return stack\_1[size];

}

void stack::push(char name){

stack\_1[size] = name;

++size;

}

int main(){

stack s1;

s1.push('a');

s1.push('b');

s1.push('c');

s1.push('d');

s1.push('e');

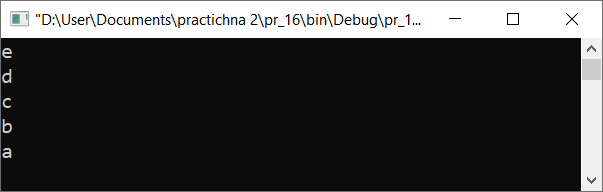
for (int i = 0; i < 5; i++)

cout << s1.pop() << endl;

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання № 17:**

Создайте класс time\_and\_date, которому при его создании передается текущее системное время и дата в виде параметров конструктора. Этот класс должен включать в себя функцию-член, выводящую время и дату на экран.

Подсказка : Для нахождения и вывода на экран этих данных воспользуйтесь стандартной библиотечной функцией времени и даты.

**Код програми:**

#include <iostream>

#include <ctime>

using namespace std;

class get\_date {

public:

get\_date(char \*data);

};

get\_date::get\_date(char \*date){

cout << date << endl;

}

int main() {

time\_t current = time(NULL);

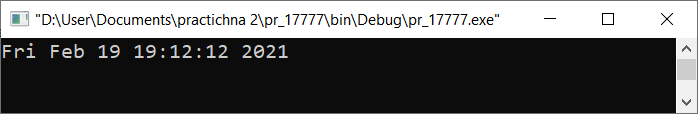
char \*date = ctime(&current);

get\_date type(date);

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання № 18:**

Создайте класс mybox, конструктору которого передаются три значения типа double,

представляющие собой длины сторон параллелепипеда. Класс mybox должен подсчитывать его объем и хранить результат также в виде значения типа double. Включите в класс функцию-член volume(), которая будет выводить на экран объем любого объекта типа mybox.

**Код програми:**

#include <iostream>

#include <cstring>

#include <cassert>

using namespace std;

//typedef double dl;

class figure{

double side\_a;

double side\_b;

double side\_c;

double volume;

public:

figure(double side\_a, double side\_b, double side\_c);

void show();

};

figure::figure(double side\_a, double side\_b, double side\_c){

volume = side\_a \* side\_b \* side\_c;

}

void figure::show(){

cout << volume;

}

int main(){

double a, b, c;

cin >> a >> b >> c;

assert(a > 0 && b > 0 && c > 0);

figure box(a, b, c);

box.show();

return 0;

}

b)

#include <iostream>

#include <cstring>

#include <cassert>

using namespace std;

class figure{

double side\_a;

double side\_b;

double side\_c;

public:

figure(double side\_a, double side\_b, double side\_c);

void show();

};

figure::figure(double side\_a, double side\_b, double side\_c){

cout << side\_a \* side\_b \* side\_c;

}

int main(){

double a, b, c;

cin >> a >> b >> c;

assert(a > 0 && b > 0 && c > 0);

figure box(a, b, c);

return 0;

}

**Результати роботи програми (скриншоти):**

