**ПРАКТИЧНЕ ЗАВДАННЯ № 3**

**Завдання № 1:**

Программа (см. Unit 3 "An example of class inheritance"), которая определяет общий базовый класс Fruit, описывающий некоторые характеристики фруктов. Этот класс наследуется двумя производными классами Apple и Orange. Эти классы содержат специальную информацию о конкретном фрукте (яблоке или апельсине). Требуется расширить программу, перегрузив методы seto и seta, таким образом, чтобы их вызовы выглядели вот так:

int main() {

Apple a2;

Orange o2;

a2.seta("Jonathan", red, yes, no, yes);

o2.seto("Valencia", orange, yes, yes, no);

...

**Код програми:**

#include <iostream>

#include <cstring>

using namespace std;

enum color { red, yellow, green, orange };

enum answer { no, yes };

void out (enum answer x);

char const \*c[] = {"red", "yellow", "green", "orange"};

class Fruit {

public:

enum answer annual;

enum answer perennial;

enum answer tree;

enum answer tropical;

enum color clr;

char name[40];

};

class Apple: public Fruit{

enum answer cooking;

enum answer crunchy;

enum answer eating;

public:

void seta(char const \*n, enum color c, enum answer ck, enum answer crchy, enum answer e);

void show();

};

class Orange: public Fruit {

enum answer juice;

enum answer sour;

enum answer eating;

public:

void seto(char const \*n, enum color c, enum answer j, enum answer sr, enum answer e);

void show();

};

void out(enum answer x){

cout << ((x == no) ? "no" : "yes") << endl;

}

void Apple::seta(char const \*n, enum color c, enum answer ck, enum answer crchy, enum answer e){

strcpy(name, n);

annual = no;

perennial = yes;

tree = yes;

tropical = no;

clr = c;

cooking = ck;

crunchy = crchy;

eating = e;

}

void Orange::seto(char const \*n, enum color c, enum answer j, enum answer sr, enum answer e){

strcpy(name, n);

annual = no;

perennial = yes;

tree = yes;

tropical = yes;

clr = c;

juice = j;

sour = sr;

eating = e;

}

void Apple::show(){

cout << name << " apple is " <<"\n";

cout << "Annual: "; out(annual);

cout <<"Perennial: "; out(perennial);

cout << "Tree: "; out (tree);

cout << "Tropical: "; out(tropical);

cout << "Color: " << c[clr] << "\n";

cout << "Good for cooking: "; out(cooking);

cout << "Crunchy: "; out(crunchy);

cout << "Good for eating: "; out(eating);

}

void Orange::show(){

cout << name << " orange is " <<"\n";

cout << "Annual: "; out(annual);

cout <<"Perennial: "; out(perennial);

cout << "Tree: "; out (tree);

cout << "Tropical: "; out(tropical);

cout << "Color: " << c[clr] << "\n";

cout << "Good for juice: "; out(juice);

cout << "Sour: "; out(sour);

cout << "Good for eating: "; out(eating);

}

int main(){

Apple a1; Orange o1;

a1.seta("Red Delicious ", red, no, yes, yes);

o1.seto("Navel ", orange, no, no, yes);

a1.show();

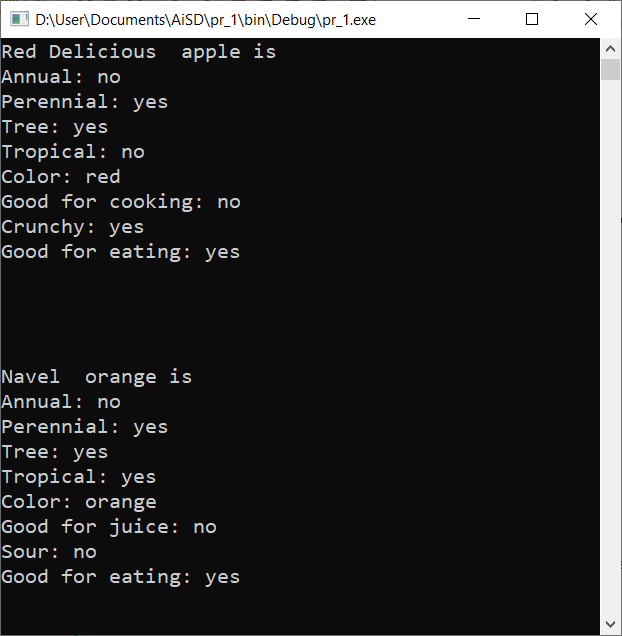
puts("\n\n\n");

o1.show();

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання № 2:**

Дан следующий базовый класс:

class Area {

public:

double height;

double width;

}

Создайте два производных класса Rectangle и Isosceles, которые наследуют базовый класс Aгеа. Каждый класс должен включать в себя функцию агеа(), которая возвращает площадь соответственно прямоугольника (rectangle) и равнобедренного треугольника (isosceles). Для инициализации переменных height и width (высота и длина основания, соответственно) используйте конструктор с параметрами. Добавьте производный класс, который наследует класс Aгеа, назовите этот класс Cylinder и пусть он вычисляет площадь поверхности цилиндра. Эта площадь задается так: 2\*pi\*R\*R + pi\*D\*height.

**Код програми:**

#include <iostream>

#include <cmath>

using namespace std;

class Area {

public:

double height;

double width;

double radius;

};

class rectangle:public Area {

public:

rectangle(double h, double w);

double area(){

return (width \* height);

}

};

class isosceles : public Area {

public:

isosceles(double h, double w);

double area(){

return 0.5 \* width \* height;

}

};

class cylinder : public Area {

public:

cylinder(double r);

double area(){

return 2 \* 3.1415 \* radius\*radius + 2 \* 3.1415 \* radius;

}

};

rectangle::rectangle(double h, double w) {

height = h;

width = w;

}

isosceles::isosceles(double h, double w) {

height = h;

width = w;

}

cylinder::cylinder(double r) {

radius = r;

}

int main() {

double a, b, c;

cin >> a >> b >> c;

rectangle g(a, b);

isosceles i(a, b);

cylinder h(c);

cout << g.area() << endl;

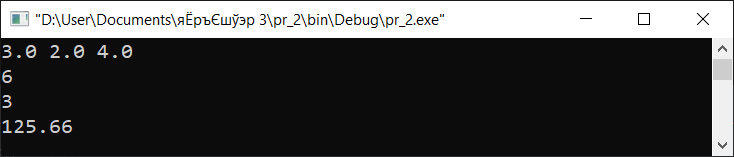
cout << i.area() << endl;

cout << h.area() << endl;

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання № 3:**

Используйте объединение (union), чтобы поменять местами старший и младший байты int (Намек: наверное ваш компьютер использует 64-битное целое).

**Код програми:**

#include <iostream>

#define SIZE 100

using namespace std;

union byte\_swap {

unsigned char c[SIZE];

unsigned i;

byte\_swap(unsigned x);

void swapping(int n);

};

byte\_swap::byte\_swap(unsigned x) {

i = x;

}

void byte\_swap::swapping(int n) {

swap(c[0], c[n]);

cout << i << endl;

}

int main() {

int n;

cin >> n;

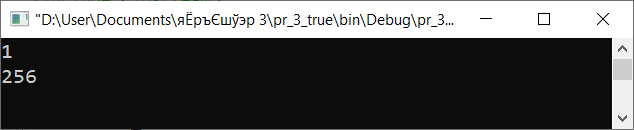
byte\_swap t(n);

t.swapping(n);

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання № 4:**

Компиляторы бывают разные. В зависимости от типа вашего компилятора возможны некоторые ограничения на использование встраиваемых функций. Например, некоторые компиляторы не воспринимают функцию как встраиваемую, если функция является рекурсивной или если она содержит либо статическую static переменную, либо любую инструкцию выполнения цикла, либо инструкцию switch. Вам необходимо написать программу, при помощи которой выяснить ограничения на использование встраиваемых функций. Подсказка: целесообразно просмотреть руководство по вашему компилятору, чтобы точно определить ограничения на использование встраиваемых функций.

**Код програми:**

#include <iostream>

using namespace std;

inline int power(int x, int n){

return (n == 1) ? x : (x \* power(x, n - 1));

}

int main(){

int x, n;

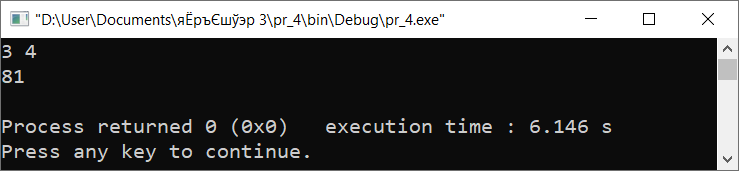
cin >> x >> n;

cout << power(x, n) << endl;

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання № 5:**

B Unit 2 вы перегружали функцию abs() так, чтобы она находила абсолютные значения типа int, long и double. Модифицируйте программу, чтобы эти функции сталивстраиваемыми.

**Код програми**

#include <iostream>

using namespace std;

inline int absolute(int n) {

return n < 0 ? -n : n;

}

inline long absolute(long n) {

return n < 0 ? -n : n;

}

inline double absolute(double n) {

return n < 0 ? -n : n;

}

int main() {

cout << absolute(-4) << endl;

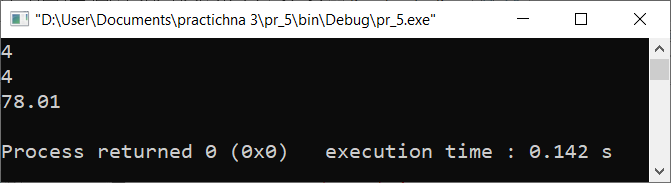
cout << absolute(-4L) << endl;

cout << absolute(-78.01) << endl;

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання № 6:**

Переделайте класс stack (см. Unit 2 "A more practical example"), так, чтобы в классе, где это возможно, использовались встраиваемые функции.

**Код програми:**

**A)**

#include <iostream>

#define SIZE 10

using namespace std;

class stack {

int index;

char stack\_1[SIZE];

public:

void init();

void push(char name);

char pop();

};

inline void stack::init(){

index = 0;

}

inline void stack::push(char name){

if(index == SIZE)

return;

stack\_1[index] = name;

++index;

}

inline char stack::pop(){

if (index == 0){

return 0;

}

--index;

return stack\_1[index];

}

int main(){

stack s1, s2;

s1.init(); s2.init();

s1.push('a');

s1.push('b');

s1.push('c');

s1.push('d');

s2.push('1');

s2.push('2');

s2.push('3');

s2.push('4');

for (int i = 0; i < 4; ++i)

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

puts("\n\n");

for (int i = 0; i < 4; ++i)

cout << s2.pop() << endl;

return 0;

}

**B)**

#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];

}

inline char stack::pop(){

if (size != 0)

--size;

return stack\_1[size];

}

inline void stack::push(char name){

stack\_1[size] = name;

++size;

}

int main(){

stack s1, s2;

s1.push('a');

s1.push('b');

s1.push('c');

s1.push('d');

s2.push('1');

s2.push('2');

s2.push('3');

s2.push('4');

for (int i = 0; i < 4; ++i)

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

puts("\n\n");

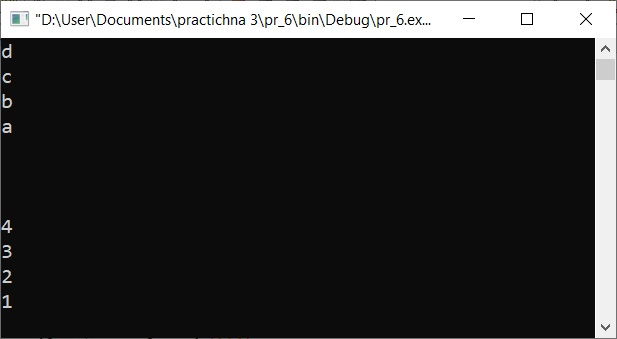
for (int i = 0; i < 4; ++i)

cout << s2.pop() << endl;

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання № 7:**

Создайте класс Dice, который содержит закрытую целую переменную. Создайте функцию roll(), использующую стандартный генератор случайных чисел rand(), для получения чисел от 1 до 6. Сделайте 5 бросков четырех игральных костей. Функция roll() должна вывести эти значения на экран.

**Код програми:**

#include <iostream>

#include <cstdlib>

using namespace std;

class Dice {

int n;

public:

void roll();

};

void Dice::roll() {

n = (rand() % 6) + 1;

cout << n << ' ';

}

int main() {

Dice a, b, c, d, e, f;

a.roll();

b.roll();

c.roll();

d.roll();

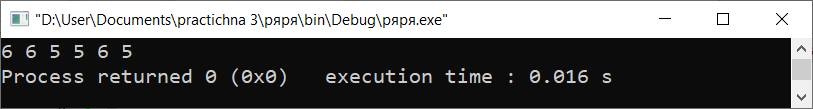
e.roll();

f.roll();

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання № 8:**

Используя класс stack (см. Unit 3 Example 3.7) добавьте в программу функцию showstack(), которой в качестве аргумента передается объект типа stack. Эта функция должна выводить содержимое стека на экран.

**Код програми:**

#include <iostream>

#define SIZE 16

using namespace std;

class stack {

char stack\_1[SIZE];

int index;

public:

stack();

void push(char name);

char pop();

};

stack::stack() {

index = 0;

}

void stack::push(char name) {

if (index == SIZE)

return;

stack\_1[index] = name;

++index;

}

char stack::pop() {

if (index == 0)

return 0;

--index;

return stack\_1[index];

}

void show(stack o);

int main() {

stack s1, s2;

s1.push('a');

s1.push('b');

s1.push('c');

s1.push('d');

s2.push('1');

s2.push('2');

s2.push('3');

s2.push('4');

show(s1);

show(s2);

return 0;

}

void show(stack a) {

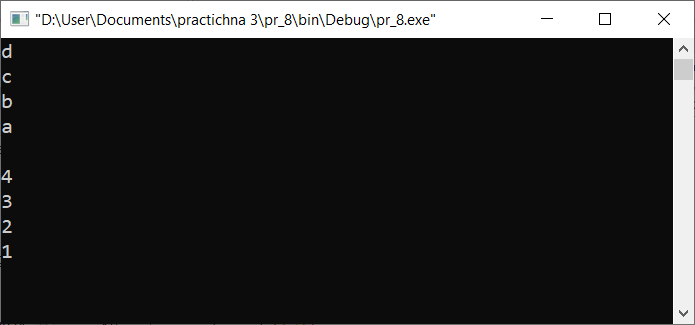
while (char ch = a.pop())

cout << ch << endl;

cout << "\n";

}

**Результати роботи програми (скриншоти):**



**Завдання № 9:**

Создайте класс who. Конструктор who должен иметь один символьный аргумент, который будет использоваться для идентификации объекта. При создании объекта конструктор должен выводить на экран сообщение:

Constructing who #x

где х — идентифицирующий символ, свой для каждого объекта. При удалении объекта на экран должно выводиться сообщение:

Destroying who #x

где х — снова идентифицирующий символ. Наконец, создайте функцию make\_who(), которая возвращает объект who. Присвойте каждому объекту уникальное имя. Проанализируйте и объясните выводимый на экран результат работы программы.

**Код програми:**

#include <iostream>

using namespace std;

class who {

char name;

public:

who(char c);

~who();

who create(char ch);

};

who::who(char c) {

name = c;

cout << "Constructing who # " << name << "\n";

}

who::~who() {

cout << "Destroying who # " << name << "\n\n";

}

who create(char ch) {

return who(ch);

}

int main() {

create('a');

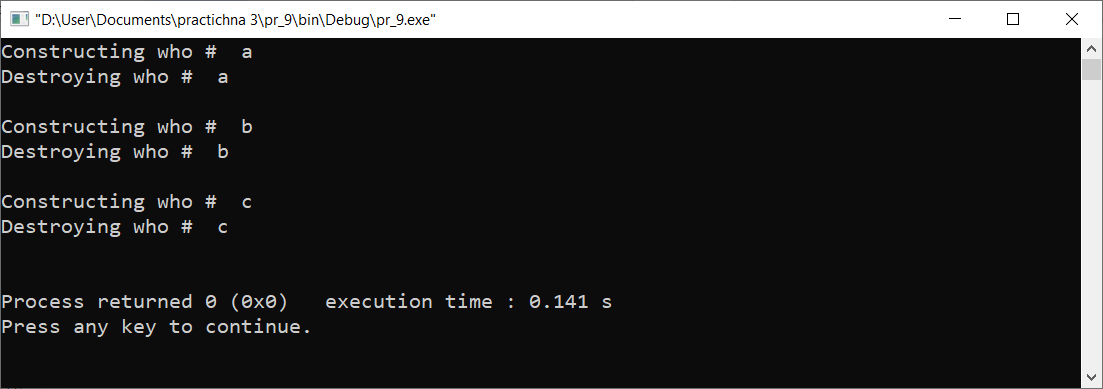
create('b');

create('c');

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання № 10:**

Придумайте пример программы, в которой, как и при неправильном освобождении динамической памяти, возвращать объект из функции было бы также ошибочно.

**Код програми:**

#include <iostream>

#include <cstring>

#define SIZE 100

using namespace std;

class str {

char \*string;

public:

str(char \*ptr){

string = new char[SIZE];

strcpy(string, ptr);

}

~str(){

// cout << "Deleting string" << endl;

delete[] string;

}

str();

str error(str data);

char \* get(){

return string;

}

};

str str::error(str data){

return data;

}

int main(){

str symb("abcdefgh");

str symba = symb.error(symb);

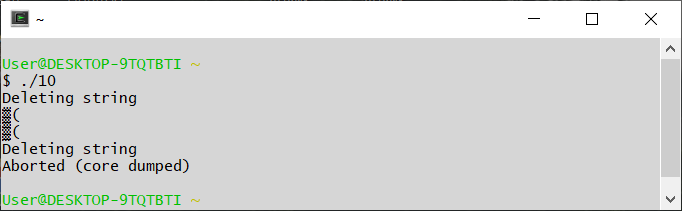
cout << symb.get() << endl;

cout << symba.get() << endl;

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання № 11:**

Представьте себе ситуацию, в которой показанные ниже два класса рг1 и рг2 используют общий принтер. Для оставшейся части программы необходимо знать, когда принтер занят объектом одного из этих классов. Создайте функцию inuse(), которая возвращает true, когда принтер занят объектом одного из классов и false - в противном случае. Сделайте эту функцию дружественной как классу pr1, так и классу рг2.

class pr1 {

int printing;

// ...

public:

pr1() { printing = 0; }

void set\_print(int status) { printing = status; }

// ...

};

class pr2 {

int printing;

// ...

public:

void set\_print(int status) { printing = status; }

// ...

};

**Код програми:**

#include <iostream>

using namespace std;

class pr2;

class pr1 {

private:

int printing;

public:

pr1(){ printing = 0; }

void set\_print(int status){

printing = status;

}

friend int inuse(pr1 a, pr2 b);

};

class pr2 {

private:

int printing;

public:

pr2(){

printing = 0;

}

void set\_print(int status){

printing = status;

}

friend int inuse(pr1 a, pr2 b);

};

int inuse(pr1 a, pr2 b){

return (a.printing || b.printing) ? 1: 0;

}

/\*

bool inuse(pr1 a, pr2 b){

return (a.printing == 1 || b.printing == 1 ? true : false);

}\*/

int main(){

pr1 a; pr2 b;

inuse(a, b) ? cout << "full" : cout << "empty" << endl;

a.set\_print(2);

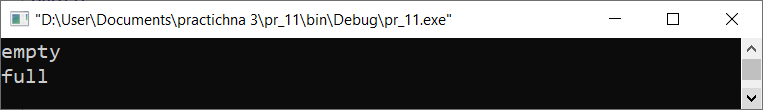
b.set\_print(4);

inuse(a, b) ? cout << "full" : cout << "empty" << endl;

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання № 12:**

У вас есть следующий класс:

class planet {

int moons;

double dist\_from\_sun; // in miles

double diameter;

double mass;

public:

// ....

double get\_miles() { return dist\_from\_sun; }

};

Создайте функцию light(), получающую в качестве аргумента объект типа planet и возвращающую число секунд, за которые свет достигает планеты. (Предположим, что скорость света равна 186000 миль в секунду и что значение dist\_from\_sun, т. е. расстояние от Солнца, задано в милях.)

**Код програми:**

#include <iostream>

using namespace std;

class planet {

int moons;

double dist\_from\_sun;

// double diameter;

// double mass;

public:

planet(double i){

dist\_from\_sun = i;

}

double get\_miles() {

return dist\_from\_sun;

}

};

double light(planet a) {

return a.get\_miles() / 18600;

}

int main(){

planet Venera(67682000);

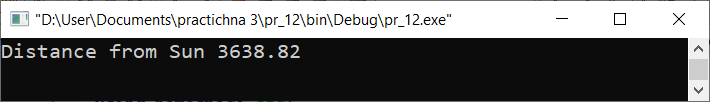
Venera.get\_miles();

cout << "Distance from Sun " << light(Venera) << endl;

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання № 13:**

Используя класс stack (см. Unit 2 "A more practical example"), напишите функцию loadstack(), которая бы возвращала стек, заполненный буквами алфавита (a-z). В вызывающей программе присвойте этот стек другому объекту и докажите, что и в этом объекте находится алфавит. *(Замечание.* Удостоверьтесь, что длина стека достаточна для хранения алфавита.). Перегрузите функцию loadstack() так, чтобы она получала в качестве аргумента целое число upper. В перегруженной версии, если переменная upper будет равной 1, загрузите стек символами алфавита в верхнем регистре. В противном случае загрузите его символами алфавита в нижнем регистре.

**Код програми:**

#include <iostream>

#define SIZE 30

using namespace std;

class stack {

char stack\_1[SIZE];

int index;

public:

stack();

void push(char name);

char pop();

};

stack::stack() {

index = 0;

}

void stack::push(char name) {

if (index == SIZE)

return;

stack\_1[index] = name;

++index;

}

char stack::pop() {

if (index == 0)

return 0;

--index;

return stack\_1[index];

}

void show(stack o);

stack fill\_s1(int g);

int main() {

stack s1, s2;

s1.push('a');

s1.push('b');

s1.push('c');

s1.push('d');

s2.push('1');

s2.push('2');

s2.push('3');

s2.push('4');

show(s1);

show(s2);

show(fill\_s1(1));

show(fill\_s1(0));

//show(fill\_s1(0));

return 0;

}

void show(stack a) {

while (char ch = a.pop())

cout << ch << ' ';

cout << "\n";

}

stack fill\_s1(int g) {

stack a;

if (g) {

for (char ch = 'Z'; ch >= 'A'; --ch)

a.push(ch);

return a;

}

else {

for (char ch = 'z'; ch >= 'a'; --ch)

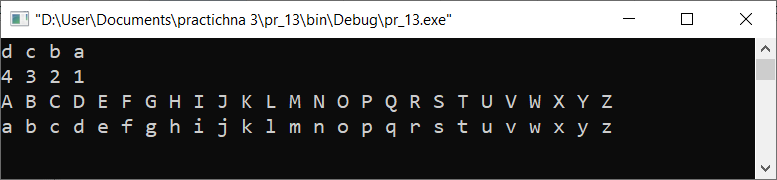
a.push(ch);

return a;

}

}

**Результати роботи програми (скриншоти):**



**Завдання № 14:**

Используя класс strtype (см. Unit 3 Example 3.2а), добавьте дружественную функцию, которая получает в качестве аргумента указатель на объект типа strtype и возвращает указатель на строку. (Таким образом, функция должна возвращать указатель р). Назовите эту функцию get\_string().

**Код програми:**

#include <iostream>

#include <cstring>

#include <cstdlib>

using namespace std;

class strtype {

char \*str;

//int len;

public:

strtype(const char \*ptr);

~strtype();

void show();

friend char \*get\_str(strtype \*str);

};

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

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

if (!str)

exit(1);

strcpy(str, ptr);

}

strtype::~strtype(){

free(str);

}

char \*get\_str(strtype \*ptr){

return ptr->str;

}

void strtype::show(){

cout << str << endl;

}

int main(){

char y[100];

cin.getline(y, 100);

strtype s(y);

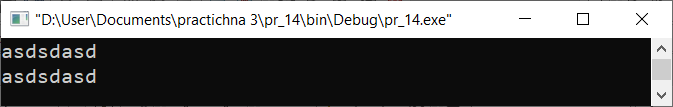
char \*p = get\_str(&s);

cout << p;

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання № 15:**

Проведите эксперимент: если объект производного класса присваивается другому объекту того же производного класса, будут ли также копироваться данные, связанные с базовым классом? Чтобы ответить, воспользуйтесь следующими двумя классами и напишите программу.

class base {

int a;

public:

void load\_a(int n) { a = n; }

int get\_a() { return a; }

};

class derived: public base {

int b;

public:

void load\_b(int n) { b = n; }

int get\_b() { return b; }

};

**Код програми:**

#include<iostream>

using namespace std;

class base {

int a;

public:

void load\_a(int n) {

a = n;

}

int get\_a() {

return a;

}

};

class derived : public base {

int b;

public:

void load\_b(int n) {

b = n;

}

int get\_b() {

return b;

}

};

int main(){

derived m, n;

int a, b;

cin >> a >> b;

m.load\_a(a);

m.load\_b(b);

n = m;

cout << n.get\_a() << " <- n.a , n.b -> " << n.get\_b() << endl;

return 0;

}

**Результати роботи програми (скриншоти):**

