1. Пример наследования:

#include <iostream>

#include <string>

using namespace std;

class Pet {

public:

int legs = 4;

void legsCount() {

cout << legs << endl;

}

};

class Cat : public Pet {

public:

void Voice() {

legsCount();

cout << "Meow!" << endl;

}

};

int main()

{

Pet Timmy;

Timmy.legsCount();

Cat Barsic;

Barsic.Voice();

return 0;

}

#include <iostream>

using namespace std;

class Computer

{

public:

void turn\_on()

{

cout << "Welcome!" << endl;

}

};

class Monitor

{

public:

void show\_image()

{

cout << "Imagine image here" << endl;

}

};

class Laptop : public Computer, public Monitor

{

public:

void mymodel()

{

cout << "My model is hdkaja1298";

}

};

int main() {

Laptop model;

model.turn\_on();

model.show\_image();

return 0;

}

1. Пример работы с модификаторами доступа:

#include <iostream>

#include <string>

using namespace std;

class Animal

{

public:

string name;

int age;

Animal(string n, int a)

{

name = n; age = a;

}

void print()

{

std::cout << "Name: " << name << "\tAge: " << age << std::endl;

}

};

class Dragon : public Animal

{

public:

Dragon(string n, int a, bool c) :Animal(n, a)

{

wing = c;

}

bool wing;

};

int main()

{

Dragon Bob("Bob", 5000, 0);

cout << Bob.wing << endl;

return 0;

}

1. Пример переопределения методов базового класса:

#include <iostream>

#include <string>

using namespace std;

class Pet {

public:

void Voice() {

cout << "????" << endl;

}

};

class Cat : private Pet {

public:

void Voice() {

cout << "Meow!" << endl;

}

};

int main()

{

Pet Timmy;

Timmy.Voice();

Cat Barsic;

Barsic.Voice();

return 0;

}

1. Пример создания финального класса:

class PrintA final {

public:

void func() { cout << "A" << endl; }

};

1. Пример перечисления

#include <iostream>

#include <string>

enum Colors

{

YELLOW, // 0

BLACK, // 1

PINK // 2

};

int main()

{

Colors currentColor = BLACK;

std::cout << "Colors: " << currentColor << std::endl;

return 0;

}