№1

*main.cpp*

#include <iostream>

#include "human.h"

using namespace std;

int main() {

Human hum1;

hum1.printHuman();

Human hum2("Vova");

hum2.printHuman();

Human hum3("Vovan", "Ivanov");

hum3.printHuman();

Human hum4("Vovanchik", "Ivanov", -31);

hum4.printHuman();

Human \*hum5 = new Human("Vovchansky", "Ivanov", 31, -84);

hum5->printHuman();

Human \*hum6 = new Human();

hum6->setName("Vovinsky");

hum6->setSecondName("Ivanov");

hum6->setAge(31);

hum6->setWeight(-84);

hum6->setLength(-178);

hum6->printHuman();

Human hum7("Vladimir", "Ivanov", 31, 84, 178);

hum7.printHuman();

delete hum5;

delete hum6;

return 0;

}

*human.h*

#include <iostream>

class Human {

private:

const char \*name;

const char \*secondname;

int age;

int weight;

int length;

public:

Human();

Human(const char \*n, const char \*sn, int a, int w, int l);

Human(const char \*n, const char \*sn, int a, int w);

Human(const char \*n, const char \*sn, int a);

Human(const char \*n, const char \*sn);

Human(const char \*n);

~Human();

const char\* getName() const;

const char\* getSecondName() const;

int getAge() const;

int getWeight() const;

int getLength() const;

void setName(const char \*n);

void setSecondName(const char \*sn);

void setAge(int a);

void setWeight(int w);

void setLength(int l);

double BodyIndex()const;

int YearBirt()const;

void printHuman() const;

};

*humam.cpp*

#include <iostream>

#include "human.h"

Human::Human() {

this->name = "Unknown";

this->secondname = "Unknown";

this->age = 0;

this->weight = 0;

this->length = 0;

}

Human::Human(const char\* name) {

this->name = name;

this->secondname = "Unknown";

this->age = 0;

this->weight = 0;

this->length = 0;

}

Human::Human(const char\* name, const char\* secondname) {

this->name = name;

this->secondname = secondname;

this->age = 0;

this->weight = 0;

this->length = 0;

}

Human::Human(const char\* name, const char\* secondname, int age) {

this->name = name;

this->secondname = secondname;

if (age > 0) {

this->age = age;

} else {

std::cout << "Недопустимый возраст!" << std::endl;

this->age = 0;

}

this->weight = 0;

this->length = 0;

}

Human::Human(const char\* name, const char\* secondname, int age, int weight) {

this->name = name;

this->secondname = secondname;

if (age > 0) {

this->age = age;

} else {

std::cout << "Недопустимый возраст!" << std::endl;

this->age = 0;

}

if (weight > 0) {

this->weight = weight;

} else {

std::cout << "Недопустимый вес!" << std::endl;

this->weight = 0;

}

this->length = 0;

}

Human::Human(const char\* name, const char\* secondname, int age, int weight, int length) {

this->name = name;

this->secondname = secondname;

if (age > 0) {

this->age = age;

} else {

std::cout << "Недопустимый возраст!" << std::endl;

this->age = 0;

}

if (weight > 0) {

this->weight = weight;

} else {

std::cout << "Недопустимый вес!" << std::endl;

this->weight = 0;

}

if (length > 0) {

this->length = length;

} else {

std::cout << "Недопустимый рост!" << std::endl;

this->length = 0;

}

}

Human::~Human() {

std::cout << "Деструктор вызван для " << name << std::endl;

}

const char\* Human::getName() const { return name; }

const char\* Human::getSecondName() const { return secondname; }

int Human::getAge() const { return age; }

int Human::getWeight() const { return weight; }

int Human::getLength() const { return length; }

void Human::setName(const char \*n) { name = n; }

void Human::setSecondName(const char \*sn) { secondname = sn; }

void Human::setAge(int a) {

if (a > 0) age = a;

else std::cout << "Недопустимый возраст!" << std::endl;

}

void Human::setWeight(int w) {

if (w > 0) weight = w;

else std::cout << "Недопустимый вес!" << std::endl;

}

void Human::setLength(int l) {

if (l > 0) length = l;

else std::cout << "Недопустимый рост!" << std::endl;

}

double Human::BodyIndex() const{

if ((weight or length) == 0) {

throw std::invalid\_argument("IMB cannot be zero");

}

double massIndex = (weight / 2.205) / ((length / 39.37) \* (length / 39.37));

return massIndex;

}

int Human::YearBirt() const{

if (age == 0) {

throw std::invalid\_argument("Age cannot be zero");

}

return 2024 - age;

}

void Human::printHuman() const {

std::cout << "------------------" << std::endl;

std::cout << "Имя: " << name << std::endl;

std::cout << "Фамилия: " << secondname << std::endl;

std::cout << "Возраст: " << age << std::endl;

std::cout << "Вес: " << weight << std::endl;

std::cout << "Рост: " << length << std::endl;

try {

std::cout << "ИМТ: " << BodyIndex() << std::endl;

}

catch (const std::invalid\_argument& e) {

std::cout << "(Нехватает данных)" << std::endl;

}

try {

std::cout << "Год рождения: " << YearBirt() << std::endl;

}

catch (const std::invalid\_argument& r) {

std::cout << "(Нехватает данных)" << std::endl;

}

std::cout << "------------------" << std::endl;

}

№2

*main.cpp*

#include <iostream>

#include "car.h"

using namespace std;

int main() {

Car car1;

car1.printCar();

Car car2("Toyota1");

car2.printCar();

Car car3("Toyota2", "Red");

car3.printCar();

Car car4("Toyota3", "Red", "Sedan");

car4.printCar();

Car\* car5 = new Car("Toyota4", "Red", "Sedan", -2.5);

car5->printCar();

Car\* car6 = new Car();

car6->setBrand("Toyota5");

car6->setColor("Red");

car6->setBodyType("Sedan");

car6->setEngineVolume(2.5);

car6->setMileage(-50000);

car6->printCar();

Car car7("Toyota6", "Red", "Sedan", 2.5, 50000);

car7.reduceMileage(10000);

car7.printCar();

delete car5;

delete car6;

return 0;

}

*car.h*

#include <iostream>

class Car {

private:

const char\* brand;

const char\* color;

const char\* bodyType;

double engineVolume;

double mileage;

public:

Car();

Car(const char\* brand);

Car(const char\* brand, const char\* color);

Car(const char\* brand, const char\* color, const char\* bodyType);

Car(const char\* brand, const char\* color, const char\* bodyType, double engineVolume);

Car(const char\* brand, const char\* color, const char\* bodyType, double engineVolume, double mileage);

~Car();

const char\* getBrand() const;

const char\* getColor() const;

const char\* getBodyType() const;

double getEngineVolume() const;

double getMileage() const;

void setBrand(const char\* brand);

void setColor(const char\* color);

void setBodyType(const char\* bodyType);

void setEngineVolume(double engineVolume);

void setMileage(double mileage);

double calculateFuelConsumption() const;

void reduceMileage(double amount);

void printCar() const;

};

*Car.cpp*

#include <iostream>

#include "car.h"

Car::Car() {

this->brand = "Unknown";

this->color = "Unknown";

this->bodyType = "Unknown";

this->engineVolume = 0.0;

this->mileage = 0.0;

}

Car::Car(const char\* brand) {

this->brand = brand;

this->color = "Unknown";

this->bodyType = "Unknown";

this->engineVolume = 0.0;

this->mileage = 0.0;

}

Car::Car(const char\* brand, const char\* color) {

this->brand = brand;

this->color = color;

this->bodyType = "Unknown";

this->engineVolume = 0.0;

this->mileage = 0.0;

}

Car::Car(const char\* brand, const char\* color, const char\* bodyType) {

this->brand = brand;

this->color = color;

this->bodyType = bodyType;

this->engineVolume = 0.0;

this->mileage = 0.0;

}

Car::Car(const char\* brand, const char\* color, const char\* bodyType, double engineVolume) {

this->brand = brand;

this->color = color;

this->bodyType = bodyType;

if (engineVolume > 0) {

this->engineVolume = engineVolume;

} else {

std::cout << "Недопустимый объем двигателя!" << std::endl;

this->engineVolume = 0;

}

this->mileage = 0.0;

}

Car::Car(const char\* brand, const char\* color, const char\* bodyType, double engineVolume, double mileage) {

this->brand = brand;

this->color = color;

this->bodyType = bodyType;

if (engineVolume > 0) {

this->engineVolume = engineVolume;

} else {

std::cout << "Недопустимый объем двигателя!" << std::endl;

this->engineVolume = 0;

}

if (mileage > 0) {

this->mileage = mileage;

} else {

std::cout << "Недопустимый пробег!" << std::endl;

this->mileage = 0;

}

}

Car::~Car() {

std::cout << "Деструктор вызван для " << brand << std::endl;

}

const char\* Car::getBrand() const { return brand; }

const char\* Car::getColor() const { return color; }

const char\* Car::getBodyType() const { return bodyType; }

double Car::getEngineVolume() const { return engineVolume; }

double Car::getMileage() const { return mileage; }

void Car::setBrand(const char\* brand) { this->brand = brand; }

void Car::setColor(const char\* color) { this->color = color; }

void Car::setBodyType(const char\* bodyType) { this->bodyType = bodyType; }

void Car::setEngineVolume(double engineVolume) {

if (engineVolume > 0) this->engineVolume = engineVolume;

else std::cout << "Недопустимый объем двигателя!" << std::endl;

}

void Car::setMileage(double mileage) {

if (mileage > 0) this->mileage = mileage;

else std::cout << "Недопустимый пробег!" << std::endl;

}

double Car::calculateFuelConsumption() const {

if (mileage == 0 or engineVolume == 0) {

throw std::invalid\_argument("Mileage cannot be zero");

}

return engineVolume / mileage \* 100;

}

void Car::reduceMileage(double amount) {

if (amount > 0 && amount <= mileage) {

mileage -= amount;

std::cout << "Пробег успешно изменен!" << std::endl;

} else {

std::cout << "Недопустимое значение для уменьшения пробега!" << std::endl;

}

}

void Car::printCar() const {

std::cout << "------------------" << std::endl;

std::cout << "Бренд: " << brand << std::endl;

std::cout << "Цвет: " << color << std::endl;

std::cout << "Тип кузова: " << bodyType << std::endl;

std::cout << "Объем двигателя: " << engineVolume << std::endl;

std::cout << "Пробег: " << mileage << " km" << std::endl;

try {

std::cout << "Расход топлива: " << calculateFuelConsumption() << " L/100km" << std::endl;

} catch (const std::invalid\_argument& e) {

std::cout << "(Нехватает данных)" << std::endl;

}

std::cout << "------------------" << std::endl;

}

№3

*main.cpp*

#include <iostream>

#include "animal.h"

using namespace std;

int main() {

Animal animal1;

animal1.printAnimal();

Animal animal2("Tom1");

animal2.printAnimal();

Animal animal3("Tom2", "Cat");

animal3.printAnimal();

Animal animal4("Tom3", "Cat", "Gray");

animal4.printAnimal();

Animal\* animal5 = new Animal("Tom4", "Cat", "Gray", -2);

animal5->printAnimal();

Animal\* animal6 = new Animal;

animal6->setName("Tom5");

animal6->setSpecies("Cat");

animal6->setColor("Grey");

animal6->setAge(2);

animal6->setWeight(-4.5);

animal6->printAnimal();

Animal animal7("Tom6", "Cat", "Gray", 2, 4.5);

animal7.increaseWeight(30);

animal7.changeName("Tomico");

animal7.printAnimal();

delete animal5;

delete animal6;

return 0;

}

*animal.h*

#include <iostream>

class Animal {

private:

const char\* name;

const char\* species;

const char\* color;

int age;

double weight;

public:

Animal();

Animal(const char\* name);

Animal(const char\* name, const char\* species);

Animal(const char\* name, const char\* species, const char\* color);

Animal(const char\* name, const char\* species, const char\* color, int age);

Animal(const char\* name, const char\* species, const char\* color, int age, double weight);

~Animal();

const char\* getName() const;

const char\* getSpecies() const;

const char\* getColor() const;

int getAge() const;

double getWeight() const;

void setName(const char\* name);

void setSpecies(const char\* species);

void setColor(const char\* color);

void setAge(int age);

void setWeight(double weight);

void increaseWeight(int days);

void changeName(const char\* newName);

void printAnimal() const;

};

*animal.cpp*

#include <iostream>

#include "animal.h"

Animal::Animal() {

this->name = "Unknown";

this->species = "Unknown";

this->color = "Unknown";

this->age = 0;

this->weight = 0.0;

}

Animal::Animal(const char\* name) {

this->name = name;

this->species = "Unknown";

this->color = "Unknown";

this->age = 0;

this->weight = 0.0;

}

Animal::Animal(const char\* name, const char\* species) {

this->name = name;

this->species = species;

this->color = "Unknown";

this->age = 0;

this->weight = 0.0;

}

Animal::Animal(const char\* name, const char\* species, const char\* color) {

this->name = name;

this->species = species;

this->color = color;

this->age = 0;

this->weight = 0.0;

}

Animal::Animal(const char\* name, const char\* species, const char\* color, int age) {

this->name = name;

this->species = species;

this->color = color;

if (age >= 0) {

this->age = age;

} else {

std::cout << "Недопустимый возраст!" << std::endl;

this->age = 0;

}

this->weight = 0.0;

}

Animal::Animal(const char\* name, const char\* species, const char\* color, int age, double weight) {

this->name = name;

this->species = species;

this->color = color;

if (age >= 0) {

this->age = age;

} else {

std::cout << "Недопустимый возраст!" << std::endl;

this->age = 0;

}

if (weight > 0) {

this->weight = weight;

} else {

std::cout << "Недопустимый вес!" << std::endl;

this->weight = 0.0;

}

}

Animal::~Animal() {

std::cout << "Деструктор вызван для " << name << std::endl;

}

const char\* Animal::getName() const { return name; }

const char\* Animal::getSpecies() const { return species; }

const char\* Animal::getColor() const { return color; }

int Animal::getAge() const { return age; }

double Animal::getWeight() const { return weight; }

void Animal::setName(const char\* name) { this->name = name; }

void Animal::setSpecies(const char\* species) { this->species = species; }

void Animal::setColor(const char\* color) { this->color = color; }

void Animal::setAge(int age) {

if (age >= 0) this->age = age;

else std::cout << "Недопустимый возраст!" << std::endl;

}

void Animal::setWeight(double weight) {

if (weight > 0) this->weight = weight;

else std::cout << "Недопустимый вес!" << std::endl;

}

void Animal::increaseWeight(int days) {

if (days > 0) this->weight += this->weight \* 0.01 \* days;

else std::cout << "Недопустимое количество дней!" << std::endl;

}

void Animal::changeName(const char\* newName) { this->name = newName;

std::cout << "Имя было изменено" << std::endl;

}

void Animal::printAnimal() const {

std::cout << "------------------" << std::endl;

std::cout << "Имя: " << name << std::endl;

std::cout << "Вид: " << species << std::endl;

std::cout << "Цвет: " << color << std::endl;

std::cout << "Возраст: " << age << std::endl;

std::cout << "Вес: " << weight << std::endl;

std::cout << "------------------" << std::endl;

}