**Задание 2**

**Main.cpp**

#include <iostream>

#include <fstream>

#include "teenager.h"

int main() {

Adult\* hum1 = new Adult;

Child\* hum2 = new Child;

Adult\* hum3 = new Adult;

Child\* hum4 = new Child;

Teenager\* hum5 = new Teenager;

IHuman\*\* arr = new IHuman\*[5]{ hum1, hum2, hum3, hum4, hum5 };

bool start = true;

do {

setlocale(LC\_ALL, "");

std::cout

<< "Выберите человека: " << std::endl

<< "1 = первый" << std::endl << "2 = второй" << std::endl

<< "3 = третий" << std::endl << "4 = четвертый" << std::endl

<< "5 = пятый" << std::endl << "0 = выход из программы" << std::endl;

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

char choice;

std::cin >> choice;

switch (choice) {

case '1':

arr[0]->readFromConsole();

break;

case '2':

arr[1]->readFromConsole();

break;

case '3':

arr[2]->readFromConsole();

break;

case '4':

arr[3]->readFromConsole();

break;

case '5':

arr[4]->readFromConsole();

break;

case '0': {

start = false;

break;

}

default:

break;

}

} while (start);

start = true;

do {

setlocale(LC\_ALL, "");

char choice1;

std::cout

<< "Вы хотите вывести информацию о каком-либо человеке: " << std::endl

<< "1 = да" << std::endl << "0 = нет" << std::endl;

std::cin >> choice1;

if (choice1 == '0') {

start = false;

} else if (choice1 == '1') {

char choice2;

std::cout

<< "Для какого пользователя вы хотите вывести информацию: " << std::endl

<< "1 = первый" << std::endl << "2 = второй" << std::endl

<< "3 = третий" << std::endl << "4 = четвертый" << std::endl

<< "5 = пятый" << std::endl << "0 = выход из программы" << std::endl;

std::cin >> choice2;

switch (choice2) {

case '1': arr[0]->display(); break;

case '2': arr[1]->display(); break;

case '3': arr[2]->display(); break;

case '4': arr[3]->display(); break;

case '5': arr[4]->display(); break;

case '0': start = false; break;

default: break;

}

}

} while (start);

--\*hum2;

std::cout << \*hum2;

++\*hum5;

std::cout << \*hum5;

if (\*hum2 > \*hum5) std::cout << "Рост " << hum2->getFirstName() << " больше" << std::endl;

else if (\*hum2 < \*hum5) std::cout << "Рост " << hum5->getFirstName() << " больше" << std::endl;

else std::cout << "Равны по росту";

std::ofstream out("Ihuman.txt", std::ios::out | std::ios::trunc);

out.close();

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

arr[i]->writeToFile();

std::cout << "Человек \"" << arr[i]->getFirstName() << " " << arr[i]->getLastName() << "\" успешно записан в файл." << std::endl;

}

delete hum1;

delete hum2;

delete hum3;

delete hum4;

delete hum5;

delete[] arr;

return 0;

}

**Ihuman.h**

#pragma once

#include <iostream>

#include <string>

#include <fstream>

#include "age.h"

using namespace age;

class IHuman {public:

IHuman();

IHuman(std::string firstName, std::string lastName, int age, double weight, double height);

virtual ~IHuman();

void setFirstName(std::string firstName);

std::string getFirstName();

void setLastName(std::string lastName);

std::string getLastName();

void setAge(int age);

int getAge();

void setWeight(double weight);

double getWeight();

void setHeight(double height);

double getHeight();

double BodyIndex();

int YearBirt();

IHuman& operator++();

IHuman& operator--();

bool operator<(IHuman& other);

bool operator>(IHuman& other);

friend std::ostream& operator<<(std::ostream& output, IHuman& human);

virtual void readFromConsole() = 0;

virtual void display();

virtual void writeToFile();

private:

std::string \_firstName;

std::string \_lastName;

int \_age;

double \_weight;

double \_height;

};

**Ihuman.cpp**

#include "Ihuman.h"

IHuman::IHuman() {

this->\_firstName = "undefined";

this->\_lastName = "undefined";

this->\_age = 0;

this->\_weight = 0.0;

this->\_height = 0.0;

}

IHuman::IHuman(std::string firstName, std::string lastName, int age, double weight, double height) : \_firstName(age::correctAlpha(firstName)),

\_lastName(age::correctAlpha(lastName)), \_age(age::correctCount(age)), \_weight(age::correctDecimal(weight)), \_height(age::correctDecimal(height)) {}

IHuman::~IHuman() {}

void IHuman::setFirstName(std::string firstName)

{

this->\_firstName = age::correctAlpha(firstName);

}

std::string IHuman::getFirstName()

{

return \_firstName;

}

void IHuman::setLastName(std::string lastName)

{

this->\_lastName = age::correctAlpha(lastName);

}

std::string IHuman::getLastName()

{

return \_lastName;

}

void IHuman::setAge(int age)

{

this->\_age = age::correctCount(age);

}

int IHuman::getAge()

{

return \_age;

}

void IHuman::setWeight(double weight)

{

this->\_weight = age::correctDecimal(weight);

}

double IHuman::getWeight()

{

return \_weight;

}

void IHuman::setHeight(double height)

{

this->\_height = age::correctDecimal(height);

}

double IHuman::getHeight()

{

return \_height;

}

double IHuman::BodyIndex() {

double massIndex = (\_weight / 2.205) / ((\_height / 39.37) \* (\_height / 39.37));

return massIndex;

}

int IHuman::YearBirt() {

return 2024 - \_age;

}

IHuman& IHuman::operator++() {

\_height += 10;

return \*this;

}

IHuman& IHuman::operator--() {

\_height -= 10;

return \*this;

}

bool IHuman::operator<(IHuman& other) {

return this->\_height < other.\_height;

}

bool IHuman::operator>(IHuman& other) {

return this->\_height > other.\_height;

}

std::ostream& operator<<(std::ostream& output, IHuman& human) {

output << "-----------------\n"

<< "Имя: " << human.getFirstName() << " " << human.getLastName() << "\n"

<< "Возраст: " << human.getAge() << " лет\n"

<< "Вес: " << human.getWeight() << " кг\n"

<< "Рост: " << human.getHeight() << " см\n"

<< "ИМТ: " << human.BodyIndex() << "\n"

<< "Год рождения: " << human.YearBirt()

<< "\n" << "-----------------\n";

return output;

}

void IHuman::display() {

std::cout << "Имя: " << \_firstName << " " << \_lastName << "\n"

<< "Возраст: " << \_age << "\n"

<< "Вес: " << \_weight << " kg\n"

<< "Рост: " << \_height << " cm\n"

<< "ИМТ: " << BodyIndex() << "\n"

<< "Год рождения: " << YearBirt() << "\n";

}

void IHuman::readFromConsole()

{

setlocale(LC\_ALL, "");

std::string firstName; std::cout << "Введите имя: "; std::cin >> firstName; setFirstName(firstName);

std::string lastName; std::cout << "Введите фамилию: "; std::cin >> lastName; setLastName(lastName);

int age; std::cout << "Введите возраст: "; std::cin >> age; setAge(age);

double weight; std::cout << "Введите вес: "; std::cin >> weight; setWeight(weight);

double height; std::cout << "Введите рост: "; std::cin >> height; setHeight(height);

}

void IHuman::writeToFile() {}

**Age.h**

#pragma once

#include <string>

namespace age

{

std::string correctAlpha(std::string str);

int correctCount (int count);

double correctDecimal (double decimal);

}

**Age.cpp**

#include "age.h"

#include <iostream>

std::string age::correctAlpha(std::string str)

{

setlocale(LC\_ALL, "");

for (int i = 0; i < str.length(); ++i)

{

if (!isalpha(str.at(i)))

{

std::cout << "Некорректный ввод данных в поле\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

return "undefined";

}

}

return (str == "undefined") ? "undefined" : str;

}

int age::correctCount(int count)

{

setlocale(LC\_ALL, "");

if (count >= 0) return count;

else

{

std::cout << "Некорректный ввод данных в поле\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

return 0;

}

}

double age::correctDecimal (double decimal)

{

setlocale(LC\_ALL, "");

if (decimal >= 0) return decimal;

else

{

std::cout << "Некорректный ввод данных в поле\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

return 0;

}

}

**Adult.h**

#pragma once

#include "Ihuman.h"

class Adult : virtual public IHuman

{public:

Adult();

Adult(std::string firstName, std::string lastName, int age, double weight, double height, std::string profession);

Adult(std::string profession);

~Adult();

void setProfession(std::string profession);

std::string getProfession();

friend std::ostream& operator<<(std::ostream& output, Adult& adult);

void readFromConsole();

void display();

void writeToFile();

private:

std::string \_profession;

};

**Adult.cpp**

#include "adult.h"

Adult::Adult()

{

setFirstName("undefined");

setLastName("undefined");

setAge(0);

setWeight(0);

setHeight(0);

this->\_profession = "undefined";

}

Adult::Adult(std::string firstName, std::string lastName, int age, double weight, double height, std::string profession) :

IHuman(firstName, lastName, age, weight, height), \_profession(age::correctAlpha(profession)) {}

Adult::Adult(std::string profession) : \_profession(age::correctAlpha(profession)) {}

Adult::~Adult() {}

void Adult::setProfession(std::string profession)

{

this->\_profession = age::correctAlpha(profession);

}

std::string Adult::getProfession()

{

return \_profession;

}

std::ostream& operator<<(std::ostream& output, Adult& adult) {

output << "-----------------\n"

<< "Имя: " << adult.getFirstName() << " " << adult.getLastName() << "\n"

<< "Возраст: " << adult.getAge() << " лет\n"

<< "Вес: " << adult.getWeight() << " кг\n"

<< "Рост: " << adult.getHeight() << " см\n"

<< "ИМТ: " << adult.BodyIndex() << "\n"

<< "Год рождения: " << adult.YearBirt() << "\n"

<< "Профессия: " << adult.getProfession() << "\n"

<< "-----------------\n";

return output;

}

void Adult::display()

{

setlocale(LC\_ALL, "");

IHuman::display();

std::cout << "Профессия: " << \_profession << std::endl;

}

void Adult::readFromConsole()

{

setlocale(LC\_ALL, "");

IHuman::readFromConsole();

std::string profession;std::cout << "Введите профессию: "; std::cin >> profession; setProfession(profession);

}

void Adult::writeToFile()

{

std::ofstream out;

out.open("Human.txt", std::ios::app);

if (out.is\_open())

{

out << "Имя: " << getFirstName() << std::endl

<< "Фамилия: " << getLastName() << std::endl

<< "Возраст: " << getAge() << std::endl

<< "Вес: " << getWeight() << std::endl

<< "Рост: " << getHeight() << std::endl

<< "Профессия: " << \_profession << std::endl

<< "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << std::endl;

}

out.close();

}

**Child.h**

#pragma once

#include "Ihuman.h"

class Child : virtual public IHuman

{public:

Child();

Child(std::string firstName, std::string lastName, int age, double weight, double height, int grade);

Child(int grade);

~Child();

void setGrade(int grade);

int getGrade();

friend std::ostream& operator<<(std::ostream& output, Child& child);

void readFromConsole();

void display();

void writeToFile();

private:

int \_grade;

};

**Child.cpp**

#include "child.h"

Child::Child()

{

setFirstName("undefined");

setLastName("undefined");

setAge(0);

setWeight(0);

setHeight(0);

this->\_grade = 0;

}

Child::Child(std::string firstName, std::string lastName, int age, double weight, double height, int grade) :

IHuman(firstName, lastName, age, weight, height), \_grade(age::correctCount(grade)) {}

Child::Child(int grade) : \_grade(age::correctCount(grade)) {}

Child::~Child() {}

void Child::setGrade(int grade)

{

this->\_grade = age::correctCount(grade);

}

int Child::getGrade()

{

return \_grade;

}

std::ostream& operator<<(std::ostream& output, Child& child) {

output << "-----------------\n"

<< "Имя: " << child.getFirstName() << " " << child.getLastName() << "\n"

<< "Возраст: " << child.getAge() << " лет\n"

<< "Вес: " << child.getWeight() << " кг\n"

<< "Рост: " << child.getHeight() << " см\n"

<< "ИМТ: " << child.BodyIndex() << "\n"

<< "Год рождения: " << child.YearBirt() << "\n"

<< "Класс: " << child.getGrade() << "\n"

<< "-----------------\n";

return output;

}

void Child::display()

{

setlocale(LC\_ALL, "");

IHuman::display();

std::cout << "Класс: " << \_grade << std::endl;

}

void Child::readFromConsole()

{

setlocale(LC\_ALL, "");

IHuman::readFromConsole();

int grade; std::cout << "Введите класс: "; std::cin >> grade; setGrade(grade);

}

void Child::writeToFile()

{

std::ofstream out;

out.open("Human.txt", std::ios::app);

if (out.is\_open())

{

out << "Имя: " << getFirstName() << std::endl

<< "Фамилия: " << getLastName() << std::endl

<< "Возраст: " << getAge() << std::endl

<< "Вес: " << getWeight() << std::endl

<< "Рост: " << getHeight() << std::endl

<< "Класс: " << \_grade << std::endl

<< "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << std::endl;

}

out.close();

}

**Teenager.h**

#pragma once

#include "Ihuman.h"

#include "adult.h"

#include "child.h"

class Teenager : public Adult, Child

{public:

Teenager();

Teenager(std::string firstName, std::string lastName, int age, double weight, double height, std::string profession, int grade, std::string gender);

~Teenager();

void setGender(std::string gender);

std::string getGender();

friend std::ostream& operator<<(std::ostream& output, Teenager& teenager);

void readFromConsole();

void display();

void writeToFile();

private:

std::string \_gender;

};

**Teenager.cpp**

#include "teenager.h"

Teenager::Teenager() : Teenager("undefied", "undefied", 0, 0, 0, "undefied", 0, "undefied") {}

Teenager::Teenager(std::string firstName, std::string lastName, int age, double weight, double height, std::string profession, int grade, std::string gender) :

IHuman(firstName, lastName, age, weight, height), Adult(profession), Child(grade), \_gender(gender) {}

Teenager::~Teenager() {}

void Teenager::setGender(std::string gender)

{

this->\_gender = age::correctAlpha(gender);

}

std::string Teenager::getGender()

{

return \_gender;

}

std::ostream& operator<<(std::ostream& output, Teenager& teenager) {

output << "-----------------\n"

<< "Имя: " << teenager.getFirstName() << " " << teenager.getLastName() << "\n"

<< "Возраст: " << teenager.getAge() << " лет\n"

<< "Вес: " << teenager.getWeight() << " кг\n"

<< "Рост: " << teenager.getHeight() << " см\n"

<< "ИМТ: " << teenager.BodyIndex() << "\n"

<< "Год рождения: " << teenager.YearBirt() << "\n"

<< "Профессия: " << teenager.getProfession() << "\n"

<< "Класс: " << teenager.getGrade() << "\n"

<< "Пол: " << teenager.getGender() << "\n"

<< "-----------------\n";

return output;

}

void Teenager::display()

{

setlocale(LC\_ALL, "");

IHuman::display();

std::cout << "Профессия: " << getProfession() << std::endl;

std::cout << "Класс обучения: " << getGrade() << std::endl;

std::cout << "Пол: " << \_gender << std::endl;

}

void Teenager::readFromConsole()

{

setlocale(LC\_ALL, "");

IHuman::readFromConsole();

std::cout << "Введите профессию: ";

std::string profession; std::cin >> profession; setProfession(profession);

std::cout << "Введите класс: ";

int grade; std::cin >> grade; setGrade(grade);

std::cout << "Введите пол: ";

std::string gender; std::cin >> gender; setGender(gender);

}

void Teenager::writeToFile()

{

std::ofstream out;

out.open("Human.txt", std::ios::app);

if (out.is\_open())

{

out << "Имя: " << getFirstName() << std::endl

<< "Фамилия: " << getLastName() << std::endl

<< "Возраст: " << getAge() << std::endl

<< "Вес: " << getWeight() << std::endl

<< "Рост: " << getHeight() << std::endl

<< "Профессия: " << getProfession() << std::endl

<< "Класс: " << getGrade() << std::endl

<< "Пол: " << \_gender << std::endl

<< "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << std::endl;

}

out.close();

}