БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

ИИТ БГУИР

Программное обеспечение информационных технологий

Отчет

По лабораторной работе №4

«Производные классы и наследование»

по курсу «ООТПиСП»

Выполнил студент группы 381064 Сорока А.А.

Проверил преподаватель: Лашкевич Е.М.

Минск 2015 г

**Цель работы** – изучение базовых средств языка С++ для выражения иерархических отношений между классами. Изучение основных правил наследования классов в языке С++. Использование абстрактных классов как средства для последовательного построения классов.

**Задание А4.** Создайте класс «Книга», который содержит следующую информацию: издательство, автор и количество страниц.

Создайте класс «Художественная книга» производный от «Книга» содержит дополнительную информацию: жанр.

Создайте класс «Научная книга» производный от «Книга» содержит дополнительную информацию: тематика, сложность.

**Код программы:**

Main.h:

#include <iostream>

#include <string>

using namespace std;

class Book

{

public:

Book();

Book(string, string, int);

~Book();

void SetPublisher(string str) {publisher = str; };

void SetAuthor(string str) { author = str; };

void SetPageCount(int a) { pageCount = a; };

string GetPublisher() { return publisher; };

string GetAuthor() { return author; };

int GetPageCount() { return pageCount; };

protected:

string publisher;

string author;

int pageCount;

};

class ArtBook : public Book

{

public:

ArtBook();

ArtBook(string, string, int, string);

~ArtBook();

void SetGenre(string str) { genre = str; };

string GetGenre() { return genre; };

private:

string genre;

};

class ScientificBook : public Book

{

public:

ScientificBook();

ScientificBook(string, string, int, string, int);

~ScientificBook();

void SetSubjects(string str) { subjects = str; };

string GetSubjects() { return subjects; };

void SetComplexity(int a) { complexity = a; };

int GetComplexity() { return complexity; };

private:

string subjects;

int complexity;

};

Main.cpp:

#include "main.h"

int main(int argc, char \*argv[])

{

Book a;

cout << "author 1 book " << a.GetAuthor() << endl;

ArtBook b("minsk", "sidorov", 100, "fantastic");

cout << "genre 2 book " << b.GetGenre() << endl;

ScientificBook c;

cout << "pagecount 3 book " << c.GetPageCount() << endl;

system("pause");

return 0;

}

Book::Book()

{

publisher = "123";

author = "Ivanov";

pageCount = 300;

}

Book::Book(string p, string a, int pc)

{

publisher = p;

author = a;

pageCount = pc;

}

Book::~Book()

{

}

ArtBook::ArtBook()

{

publisher = "123";

author = "Ivanov";

pageCount = 300;

genre = "test";

}

ArtBook::ArtBook(string p, string a, int pc, string g)

{

publisher = p;

author = a;

pageCount = pc;

genre = g;

}

ArtBook::~ArtBook()

{

}

ScientificBook::~ScientificBook()

{

}

ScientificBook::ScientificBook()

{

publisher = "123";

author = "Ivanov";

pageCount = 300;

subjects = "science";

complexity = 3;

}

ScientificBook::ScientificBook(string p, string a, int pc, string s, int c)

{

publisher = p;

author = a;

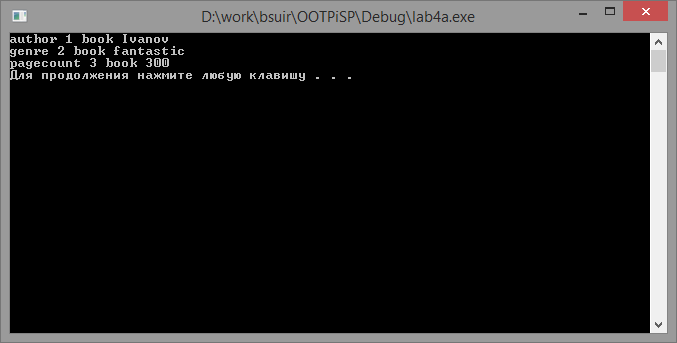
pageCount = pc;

subjects = s;

complexity = c;

}

**Скриншот выполнения программы:**



**Задание В2.**Создайте класс «Персонаж», который содержит следующую информацию: имя, здоровье, сила и ловкость.

Метод «атаковать» класса «Персонаж» генерирует случайное число в диапазоне от 0 до «сила», уменьшает «сила» на значение равное частному от деления сгенерированного числа на 10 и возвращает сгенерированное число.

Метод «уклониться» класса «Персонаж» получает силу удара противника, генерирует случайное число в диапазоне от 0 до «ловкость», уменьшает «ловкость» на значение равное частному от деления сгенерированного числа на 10, вычисляет разницу между силой удара и сгенерированным числом и, если результат положительный то значение поля «здоровье» уменьшается на это число, если нет то значение поля «здоровье» остается прежним. Возвращаемое значение поля «здоровье».

Создайте классы «Воин» и «Монстр» производные от «Персонаж» и переопределите в них функции «атаковать» и «уклониться» в соответствии с персонажем.

**Код программы:**

Main.h:

#include <iostream>

#include <string>

#include <time.h>

#include <iomanip>

using namespace std;

class Person

{

public:

Person();

Person(string, int, int, int);

~Person();

int GetHealth() { return health; };

bool alive;

virtual int Attack();

virtual int Evasion(int);

protected:

string name;

int health;

int strength;

int agility;

};

class Warrior : public Person

{

public:

Warrior();

~Warrior();

virtual int Evasion(int);

private:

int shield = 5;

};

class Monster : public Person

{

public:

Monster();

~Monster();

virtual int Attack();

};

Main.cpp:

#include "main.h"

int main(int argc, char \*argv[])

{

srand(time(NULL));

Warrior w;

Monster m;

int tmp1, tmp2;

while (m.alive && w.alive)

{

tmp1 = w.Attack();

tmp2 = m.Attack();

cout << "Warrior attack " << setw(2) << tmp1 << " " << "Monster attack " << setw(2) << tmp2 << endl;

tmp1 = m.Evasion(tmp1);

tmp2 = w.Evasion(tmp2);

cout << "Monster block " << setw(2) << tmp2 << " " << "Warrior block " << setw(2) << tmp1 << endl;

cout << "Monster health " << setw(2) << m.GetHealth() << " " << "Warrior health " << setw(2) << w.GetHealth() << endl;

cout << endl;

}

system("pause");

return 0;

}

Person::Person()

{

name = "Person";

health = 100;

strength = 20;

agility = 25;

alive = true;

}

Person::Person(string n, int h, int s, int a)

{

name = n;

health = h;

strength = s;

agility = a;

alive = true;

}

Person::~Person()

{

}

int Person::Attack()

{

int a = rand() % strength;

return strength - a;

}

int Person::Evasion(int attack)

{

int b = (agility - rand() % agility);

int a = attack - b;

health -= (a > 0 ? a : 0);

if (health < 0) alive = false;

return b;

}

Warrior::Warrior()

{

}

Warrior::~Warrior()

{

}

int Warrior::Evasion(int attack)

{

int b = (agility - rand() % agility);

int a = attack - b - shield;

health -= (a > 0 ? a : 0);

if (health < 0) alive = false;

return b;

}

Monster::Monster()

{

}

Monster::~Monster()

{

}

int Monster::Attack()

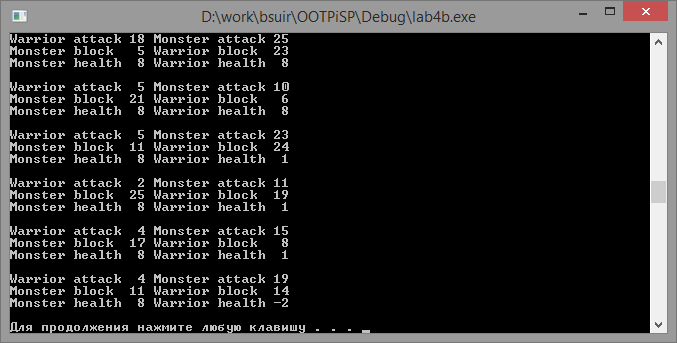
{

int a = (rand() % strength);

return (strength - a + 5);

}

**Скриншот выполнения программы:**



**Вывод**: В ходе выполнения лабораторной работы я изучил базовые средства языка С++ для выражения иерархических отношений между классами. Изучил основные правила наследования классов в языке С++. Использовал абстрактные классы как средства для последовательного построения классов.