



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Магнитогорский государственный технический университет им. Г.И. Носова»



УТВЕРЖДАЮ  
Директор ИЭиАС  
В.Р. Храмшин

13.02.2024 г.

**РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ (МОДУЛЯ)**

***ПАТТЕРНОЕ ПРОГРАММИРОВАНИЕ***

Направление подготовки (специальность)  
09.03.01 Информатика и вычислительная техника

Направленность (профиль/специализация) программы  
Программное обеспечение средств вычислительной техники и автоматизированных систем

Уровень высшего образования - бакалавриат

Форма обучения  
заочная

Институт/ факультет	Институт энергетики и автоматизированных систем
Кафедра	Вычислительной техники и программирования
Курс	5

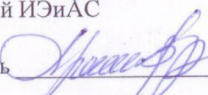
Магнитогорск  
2024 год

Рабочая программа составлена на основе ФГОС ВО - бакалавриат по направлению подготовки 09.03.01 Информатика и вычислительная техника (приказ Минобрнауки России от 19.09.2017 г. № 929)

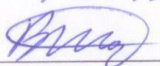
Рабочая программа рассмотрена и одобрена на заседании кафедры Вычислительной техники и программирования  
25.01.2024 г., протокол № 5


Зав. кафедрой  О.С. Логунова

Рабочая программа одобрена методической комиссией ИЭиАС  
13.02.2024 г., протокол № 4

Председатель  В.Р. Храмшин

Рабочая программа составлена:  
ст. преподаватель кафедры ВТиП,

 В.Е. Торчинский

Рецензент:  
директор НИИ «Промбезопасность», д-р техн. наук  М.Ю. Наркевич

## Лист актуализации рабочей программы

---

Рабочая программа пересмотрена, обсуждена и одобрена для реализации в 2025 - 2026 учебном году на заседании кафедры Вычислительной техники и программирования

Протокол от \_\_\_\_\_ 20\_\_ г. № \_\_\_\_  
Зав. кафедрой \_\_\_\_\_ О.С. Логунова

---

Рабочая программа пересмотрена, обсуждена и одобрена для реализации в 2026 - 2027 учебном году на заседании кафедры Вычислительной техники и программирования

Протокол от \_\_\_\_\_ 20\_\_ г. № \_\_\_\_  
Зав. кафедрой \_\_\_\_\_ О.С. Логунова

---

Рабочая программа пересмотрена, обсуждена и одобрена для реализации в 2027 - 2028 учебном году на заседании кафедры Вычислительной техники и программирования

Протокол от \_\_\_\_\_ 20\_\_ г. № \_\_\_\_  
Зав. кафедрой \_\_\_\_\_ О.С. Логунова

---

Рабочая программа пересмотрена, обсуждена и одобрена для реализации в 2028 - 2029 учебном году на заседании кафедры Вычислительной техники и программирования

Протокол от \_\_\_\_\_ 20\_\_ г. № \_\_\_\_  
Зав. кафедрой \_\_\_\_\_ О.С. Логунова

---

Рабочая программа пересмотрена, обсуждена и одобрена для реализации в 2029 - 2030 учебном году на заседании кафедры Вычислительной техники и программирования

Протокол от \_\_\_\_\_ 20\_\_ г. № \_\_\_\_  
Зав. кафедрой \_\_\_\_\_ О.С. Логунова

### **1 Цели освоения дисциплины (модуля)**

Целями освоения дисциплины (модуля) «Паттерное программирование» является освоение студентами методики проектирования и реализации сложных программных комплексов.

Для достижения поставленной цели в курсе «Паттерное программирование» решаются задачи приобретения:

- расширенных знаний об основных парадигмах объектно-ориентированного программирования;
- представлений о объектной модели C++;
- умений проектировать иерархию классов с использованием стандартных паттернов проектирования;
- навыков написания программного кода с возможностями модификации и расширения.

### **2 Место дисциплины (модуля) в структуре образовательной программы**

Дисциплина Паттерное программирование входит в часть учебного плана формируемую участниками образовательных отношений образовательной программы.

Для изучения дисциплины необходимы знания (умения, владения), сформированные в результате изучения дисциплин/ практик:

- Объектно-ориентированное программирование
- Объектно-ориентированное программное обеспечение
- Программирование
- Проектирование программных средств

Знания (умения, владения), полученные при изучении данной дисциплины будут необходимы для изучения дисциплин/практик:

- Подготовка к сдаче и сдача государственного экзамена
- Выполнение и защита выпускной квалификационной работы

### **3 Компетенции обучающегося, формируемые в результате освоения дисциплины (модуля) и планируемые результаты обучения**

В результате освоения дисциплины (модуля) «Паттерное программирование» обучающийся должен обладать следующими компетенциями:

Код индикатора	Индикатор достижения компетенции
ПК-6	Способность к формализации и алгоритмизации поставленных задач, к написанию программного кода с использованием языков программирования, определения и манипулирования данными и оформлению программного кода в соответствии установленными требованиями
ПК-6.1	Оценивает качество математической модели при формализации задачи предметной области
ПК-6.2	Оценивает качество разработанных алгоритмов для последующего кодирования
ПК-6.3	Оценивает выбор программных средств для программирования и манипулирования данными в соответствии установленными требованиями
ПК-7	Владеет способами разработки процедур интеграции программных модулей, компонент и верификации выпусков программного продукта, включая базы данных
ПК-7.1	Оценивает выбор программных средств для разработки и верификации интеграционного слоя автоматизированных систем

#### 4. Структура, объём и содержание дисциплины (модуля)

Общая трудоемкость дисциплины составляет 3 зачетных единиц 108 акад. часов, в том числе:

- контактная работа – 16,9 акад. часов;
- аудиторная – 14 акад. часов;
- внеаудиторная – 2,9 акад. часов;
- самостоятельная работа – 82,4 акад. часов;
- в форме практической подготовки – 0 акад. час;
- подготовка к экзамену – 8,7 акад. час

Форма аттестации - экзамен

Раздел/ тема дисциплины	Курс	Аудиторная контактная работа (в акад. часах)			Самостоятельная работа студента	Вид самостоятельной работы	Форма текущего контроля успеваемости и промежуточной аттестации	Код компетенции
		Лек.	лаб. зан.	практ. зан.				
1. Полиморфизм								
1.1 Раннее и позднее связывание. Таблица виртуальных функций	5	1	2		4	1. Самостоятельное изучение учебной и научной литературы. 2. Подготовка к лабораторному занятию	Проверка индивидуальных заданий	ПК-6.1, ПК-6.2, ПК-6.3, ПК-7.1
1.2 Абстрактные классы. Чистые виртуальные функции. Пример «Звездное небо»		1	2		14,4	1. Самостоятельное изучение учебной и научной литературы. 2. Подготовка к лабораторному занятию	Проверка индивидуальных заданий	ПК-6.1, ПК-6.2, ПК-6.3, ПК-7.1
Итого по разделу		2	4		18,4			
2. Множественное и виртуальное наследование								
2.1 Принцип множественного наследования. Область видимости класса при множественном наследовании	5	1	2		8	1. Самостоятельное изучение учебной и научной литературы. 2. Подготовка к лабораторному занятию	Проверка индивидуальных заданий	ПК-6.1, ПК-6.2, ПК-6.3, ПК-7.1

2.2	Виртуальное наследование		1	2		12	1. Самостоятельное изучение учебной и научной литературы. 2. Подготовка к лабораторному занятию	Проверка индивидуальных заданий	ПК-6.1, ПК-6.2, ПК-6.3, ПК-7.1
Итого по разделу			2	4		20			
3. Шаблоны (паттерны) проектирования									
3.1	Понятие шаблона проектирования. Каталог паттернов проектирования. Паттерн «Одиночка» (Singleton)			1		8	1. Самостоятельное изучение учебной и научной литературы. 2. Подготовка к лабораторному занятию	Проверка индивидуальных заданий	ПК-6.1, ПК-6.2, ПК-6.3, ПК-7.1
3.2	Паттерн «Стратегия» (Strategy)			1		12	1. Самостоятельное изучение учебной и научной литературы. 2. Подготовка к лабораторному занятию	Проверка индивидуальных заданий	ПК-6.1, ПК-6.2, ПК-6.3, ПК-7.1
3.3	Паттерн «Наблюдатель» (Observer)	5				12	1. Самостоятельное изучение учебной и научной литературы. 2. Подготовка к лабораторному занятию	Проверка индивидуальных заданий	ПК-6.1, ПК-6.2, ПК-6.3, ПК-7.1
3.4	Паттерн «Декоратор» (Decorator)					8	1. Самостоятельное изучение учебной и научной литературы. 2. Подготовка к лабораторному занятию	Проверка индивидуальных заданий	ПК-6.1, ПК-6.2, ПК-6.3, ПК-7.1
3.5	Паттерн Команда (Command)					4	1. Самостоятельное изучение учебной и научной литературы. 2. Подготовка к лабораторному занятию	Проверка индивидуальных заданий	ПК-6.1, ПК-6.2, ПК-6.3, ПК-7.1
Итого по разделу				2		44			
Итого за семестр			4	10		82,4		экзамен	
Итого по дисциплине			4	10		82,4		экзамен	

## **5 Образовательные технологии**

1. Традиционные образовательные технологии, ориентированные на организацию образовательного процесса и предполагающую прямую трансляцию знаний от преподавателя к студенту.

Формы учебных занятий с использованием традиционных технологий:

Информационная лекция – последовательное изложение материала в дисциплинарной логике, осуществляемое преимущественно вербальными средствами (монолог преподавателя).

Практическое занятие, посвященное освоению конкретных умений и навыков по предложенному алгоритму.

2. Технологии проблемного обучения – организация образовательного процесса, которая предполагает постановку проблемных вопросов, создание учебных проблемных ситуаций для стимулирования активной познавательной деятельности студентов.

Формы учебных занятий с использованием технологий проблемного обучения:

Проблемная лекция – изложение материала, предполагающее постановку проблемных и дискуссионных вопросов, освещение различных научных подходов, авторские комментарии, связанные с различными моделями интерпретации изучаемого материала.

3. Интерактивные технологии – организация образовательного процесса, которая предполагает активное и нелинейное взаимодействие всех участников, достижение на этой основе лично-значимого для них образовательного результата.

Формы учебных занятий с использованием специализированных интерактивных технологий:

Лекция «обратной связи» – лекция-провокация (изложение материала с заранее запланированными ошибками), лекция-беседа, лекция-дискуссия, лекция-прессконференция.

Практическое занятие в форме практикума – организация учебной работы, направленная на решение комплексной учебно-познавательной задачи, требующей от студента применения как научно-теоретических знаний, так и практических навыков.

## **6 Учебно-методическое обеспечение самостоятельной работы обучающихся**

Представлено в приложении 1.

## **7 Оценочные средства для проведения промежуточной аттестации**

Представлены в приложении 2.

## **8 Учебно-методическое и информационное обеспечение дисциплины (модуля)**

### **а) Основная литература:**

1. Приемы объектно ориентированного проектирования. Паттерны проектирования. [Электронный ресурс] : справ. / Э. Гамма [и др.]. — Электрон. дан. — М. : ДМК Пресс, 2014. — 368 с. — Режим доступа: <http://e.lanbook.com/book/1220> — Загл. с экрана.

2. Липпман, С. Язык программирования C++. Полное руководство. [Электронный ресурс] : рук. / С. Липпман, Ж. Лажоие. — Электрон. дан. — М. : ДМК Пресс, 2016. — 1105 с. — Режим доступа: <http://e.lanbook.com/book/1216> — Загл. с экрана.

### **б) Дополнительная литература:**

1. Страуструп, Б. Дизайн и эволюция C++. [Электронный ресурс] — Электрон. дан. — М. : ДМК Пресс, 2015. — 448 с. — Режим доступа:

<http://e.lanbook.com/book/1222> — Загл. с экрана.

2. Аммерааль, Л. STL для программистов на C++. [Электронный ресурс] — Электрон. дан. — М. : ДМК Пресс, 2006. — 240 с. — Режим доступа: <http://e.lanbook.com/book/1218> — Загл. с экрана.

**в) Методические указания:**

**г) Программное обеспечение и Интернет-ресурсы:**

**Программное обеспечение**

Наименование ПО	№ договора	Срок действия лицензии
MS Office 2007 Professional	№ 135 от 17.09.2007	бессрочно
Borland Turbo C++	№112301 от 23.11.2005	бессрочно
MS Visual Studio 2017 Community Edition	свободно распространяемое ПО	бессрочно

**Профессиональные базы данных и информационные справочные системы**

Название курса	Ссылка

**9 Материально-техническое обеспечение дисциплины (модуля)**

Материально-техническое обеспечение дисциплины включает:

Лекционная аудитория — мультимедийные средства хранения, передачи и представления информации.

Компьютерный класс — персональные компьютеры с компиляторами C++, пакетом Office, выходом в Интернет и с доступом в электронную информационно-образовательную среду университета.

Аудитории для самостоятельной работы: компьютерные классы; читальные залы библиотеки — все классы УИТиАСУ с персональными компьютерами, выходом в Интернет и с доступом в электронную информационно-образовательную среду университета.

Аудитории для групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации — ауд. 282 и классы УИТиАСУ.

Помещения для самостоятельной работы обучающихся, оснащенных компьютерной техникой с возможностью подключения к сети «Интернет» и наличием доступа в электронную информационно-образовательную среду организации — классы УИТиАСУ.

Помещения для хранения и профилактического обслуживания учебного оборудования — ауд. 379.



По дисциплине «Шаблонное программирование» предусмотрена аудиторная и внеаудиторная самостоятельная работа обучающихся.

Аудиторная самостоятельная работа студентов предполагает решение контрольных задач на лабораторно-практических занятиях.

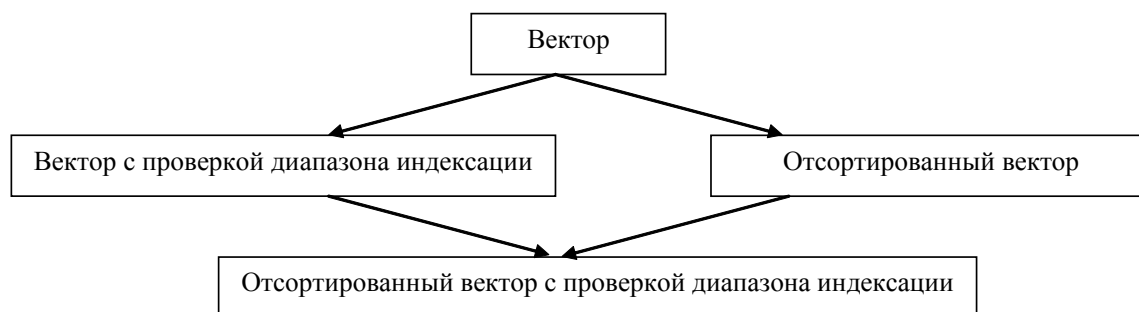
Примерные аудиторные контрольные работы (АКР):

### Раздел 1.

1. Реализовать класс «Длинное целое». Обеспечить возможность выполнения арифметических операций с экземплярами класса. Протестировать корректность программы на случайных числах.
2. Разработать информационную систему для моделирования геобиоценоза. Использовать полиморфизм. Обеспечить слабую связь между классом "Объект карты" и классом "Карта".

### Раздел 2.

1. Реализовать иерархию классов согласно следующей схеме:



### Раздел 3.

1. Спроектировать и реализовать иерархию классов для игровых персонажей и разных типов вооружения. Каждый персонаж в любой момент времени использует только один вид оружия, но может свободно менять оружие в ходе игры. Использовать паттерн Стратегия.
2. Промоделировать чат на основе паттерна Observer.
3. Адаптировать лекционный пример из темы паттерн «Декоратор» в соответствии с новыми требованиями: Теперь кофе можно заказать в маленькой, средней или большой чашке. Starbuzz считает размер порции неотъемлемой частью класса кофе, поэтому в класс Beverage были добавлены два новых метода: setSize() и getSize(). Стоимость дополнений также зависит от размера порции, так что, скажем, добавка сои должна стоить 10, 15 или 20 центов для маленькой, средней или большой порции соответственно.

```
#include <string>
```

```
#include <iostream>
```

```
using namespace std;
```

```
class Beverage
{
protected:
    string description;
public:
    virtual string getDescription()    {return description;}
    virtual double cost()=0;
};
class Espresso : public Beverage
{
public:
    Espresso() {description="Espresso";}
    virtual double cost()    {return 1.99;}
};
class HouseBlend : public Beverage
{
public:
    HouseBlend()    {description="House Blend coffee";}
    virtual double cost()    {return 0.99;}
};
class DarkRoast : public Beverage
{
public:
    DarkRoast() {description="Dark Roast coffee";}
    virtual double cost()    {return 1.39;}
};
class CondimentDecorator : public Beverage
{
protected:
```

```

    Beverage *beverage;

};

class Soy : public CondimentDecorator
{
public:
    Soy(Beverage *b) {beverage=b;}

    virtual string getDescription()    {return beverage->getDescription()+" Soy";}
    virtual double cost()              {return 0.20+beverage->cost();}
    ~Soy()                             {delete beverage;}
};

class Whip : public CondimentDecorator
{
public:
    Whip(Beverage *b) {beverage=b;}

    virtual string getDescription()    {return beverage->getDescription()+" Whip";}
    virtual double cost()              {return 0.15+beverage->cost();}
    ~Whip()                           {delete beverage;}
};

int _tmain(int argc, _TCHAR* argv[])
{
    Beverage *b=new Espresso;

    cout<<b->getDescription()<<" $"<<b->cost()<<endl;

    delete b;

    Beverage *b2=new HouseBlend;

    b2=new Soy(b2);

    b2=new Whip(b2);

    b2=new Whip(b2);

    cout<<b2->getDescription()<<" $"<<b2->cost()<<endl;

    delete b2;
}

```

```
return 0;
```

```
}
```

4. Доработать лекционный пример из темы паттерн «Команда»: Добавить устройство — трехскоростной вентилятор и реализовать функцию отмены последней операции.

```
#include <iostream>
```

```
#include <string>
```

```
using namespace std;
```

```
class Light
```

```
{
```

```
    string descr;
```

```
public:
```

```
    Light(string d) {descr=d;}
```

```
    void on() {cout<<descr<<": Light is on"<<endl;}
```

```
    void off() {cout<<descr<<": Light is off"<<endl;}
```

```
};
```

```
class GarageDoor
```

```
{
```

```
public:
```

```
    void up() {cout<<"Garage door is open"<<endl;}
```

```
    void down() {cout<<"Garage door is close"<<endl;}
```

```
};
```

```
class Stereo
```

```
{
```

```
public:
```

```
    void on() {cout<<"Stereo is on"<<endl;}
```

```
    void off() {cout<<"Stereo is off"<<endl;}
```

```
    void setCd() {cout<<"Stereo is set for CD input"<<endl;}
```

```
    void setDvd() {cout<<"Stereo is set for DVD input"<<endl;}
```

```
    void setRadio() {cout<<"Stereo is set for radio"<<endl;}
```

```

        void setVolume(int v)        {cout<<"Stereo volume set to "<<v<<endl;}
};

class Command
{
public:
    virtual void execute()=0;
};

class NoCommand : public Command
{
public:
    virtual void execute() {}
};

class LightOnCommand : public Command
{
    Light *light;
public:
    LightOnCommand(Light *l)    {light=l;}
    virtual void execute()      {light->on();}
};

class LightOffCommand : public Command
{
    Light *light;
public:
    LightOffCommand(Light *l)   {light=l;}
    virtual void execute()      {light->off();}
};

class GarageDoorUpCommand : public Command
{
    GarageDoor *garageDoor;

```

public:

```
GarageDoorUpCommand(GarageDoor *gd) { garageDoor=gd;}
```

```
virtual void execute()          { garageDoor->up();}
```

};

class GarageDoorDownCommand : public Command

{

```
GarageDoor *garageDoor;
```

public:

```
GarageDoorDownCommand(GarageDoor *gd)  { garageDoor=gd;}
```

```
virtual void execute()          { garageDoor->down();}
```

};

class StereoOnWithCDCommand : public Command

{

```
Stereo *stereo;
```

public:

```
StereoOnWithCDCommand(Stereo *s)  { stereo=s;}
```

```
virtual void execute()
```

```
{
```

```
    stereo->on();
```

```
    stereo->setCd();
```

```
    stereo->setVolume(11);
```

```
}
```

};

class StereoOffCommand : public Command

{

```
Stereo *stereo;
```

public:

```
StereoOffCommand(Stereo *s)  { stereo=s;}
```

```
virtual void execute()  { stereo->off();}
```

```

};

//Пульт
class RemoteControl
{
    int countButtons;

    Command **onCommands;

    Command **offCommands;

    NoCommand *noCommand;

public:
    RemoteControl(int c)
    {
        countButtons=c;

        onCommands=new Command*[c];
        offCommands=new Command*[c];
        noCommand=new NoCommand;

        for(int i=0; i<c; i++)
        {
            onCommands[i]=noCommand;
            offCommands[i]=noCommand;
        }
    }

    ~RemoteControl()
    {
        delete []onCommands;

        delete []offCommands;

        delete noCommand;
    }

    void setCommand(int slot, Command *onCommand, Command *offCommand)
    {

```

```

        onCommands[slot]=onCommand;

        offCommands[slot]=offCommand;

    }

    void onButtonWasPushed(int slot) { onCommands[slot]->execute();}

    void offButtonWasPushed(int slot) { offCommands[slot]->execute();}

};

int _tmain(int argc, _TCHAR* argv[])

{

    RemoteControl rc(7);

    Light *livingRoomLight = new Light("Living Room");

    Light *kitchenLight = new Light("Kitchen");

    GarageDoor *garageDoor = new GarageDoor;

    Stereo *stereo = new Stereo;

    LightOnCommand *livingRoomLightOn = new LightOnCommand(livingRoomLight);

    LightOffCommand *livingRoomLightOff = new LightOffCommand(livingRoomLight);

    LightOnCommand *kitchenLightOn = new LightOnCommand(kitchenLight);

    LightOffCommand *kitchenLightOff = new LightOffCommand(kitchenLight);

    GarageDoorUpCommand *garageDoorUp = new GarageDoorUpCommand(garageDoor);

    GarageDoorDownCommand *garageDoorDown = new
GarageDoorDownCommand(garageDoor);

    StereoOnWithCDCommand *stereoOnWithCD = new StereoOnWithCDCommand(stereo);

    StereoOffCommand *stereoOff = new StereoOffCommand(stereo);

    rc.setCommand(0, livingRoomLightOn, livingRoomLightOff);

    rc.setCommand(1, kitchenLightOn, kitchenLightOff);

    rc.setCommand(2, garageDoorUp, garageDoorDown);

    rc.setCommand(3, stereoOnWithCD, stereoOff);

    for(int i=0; i<7; i++)

    {

        rc.onButtonWasPushed(i);

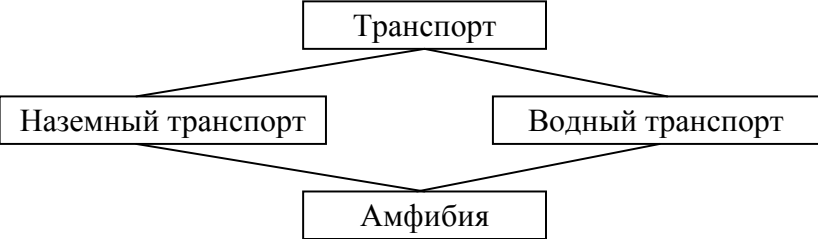
```



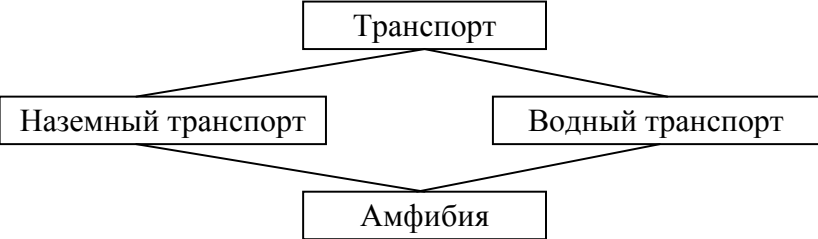
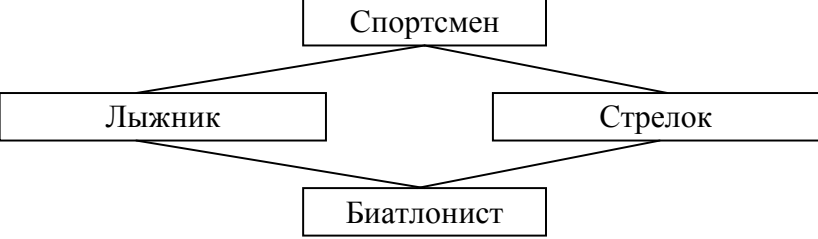
```
        rc.offButtonWasPushed(i);
    }
    delete livingRoomLightOn;
    delete livingRoomLightOff;
    delete kitchenLightOn;
    delete kitchenLightOff;
    delete garageDoorUp;
    delete garageDoorDown;
    delete stereoOnWithCD;
    delete stereoOff;
    delete livingRoomLight;
    delete kitchenLight;
    delete garageDoor;
    delete stereo;
    return 0;
}
```

а) Планируемые результаты обучения и оценочные средства для проведения промежуточной аттестации:

Код индикатора	Индикатор достижения компетенции	Оценочные средства
<p><b>ПК-6: Способность к формализации и алгоритмизации поставленных задач, к написанию программного кода с использованием языков программирования, определения и манипулирования данными и оформлению программного кода в соответствии установленными требованиями</b></p>		
<p>ПК-6.1</p>	<p>Оценивает качество математической модели при формализации задачи предметной области</p>	<p><i>Перечень теоретических вопросов</i></p> <ol style="list-style-type: none"> <li>1. Статическое или раннее связывание (static/early binding). Позднее/динамическое связывание (late/dynamic binding). Таблица виртуальных функций (virtual function table).</li> <li>2. Виртуальные функции/методы (virtual functions/methods). Абстрактные классы (abstract classes) и чистые виртуальные функции (pure virtual functions).</li> <li>3. Множественное наследование. Разрешение противоречий при наследовании одноименных членов класса.</li> <li>4. Влияние множественного наследования на механизм виртуальных функций. Область видимости класса при множественном наследовании.</li> <li>5. Виртуальное наследование.</li> <li>6. Исключения и наследование.</li> <li>7. Шаблоны (паттерны) проектирования. Основные понятия. Каталог паттернов проектирования.</li> <li>8. Паттерн «Стратегия» (Strategy).</li> <li>9. Паттерн «Наблюдатель» (Observer).</li> <li>10. Паттерн «Декоратор» (Decorator).</li> <li>11. Паттерн «Одиночка» (Singleton).</li> <li>12. Паттерн «Команда» (Command)</li> </ol>

Код индикатора	Индикатор достижения компетенции	Оценочные средства
		<p><i>Практические задания</i></p> <p>1. Спроектировать иерархию классов согласно рисунку:</p>  <pre> graph TD     A[Транспорт] --&gt; B[Наземный транспорт]     A --&gt; C[Водный транспорт]     B --&gt; D[Амфибия]     C --&gt; D </pre> <p>В числе других должен быть определен метод способПередвижения().</p> <p>2. Спроектировать иерархию классов для моделирования сети Bluetooth. Сетевые устройства могут объединяться в «пикосеть» (piconet). В каждой пикосети одно устройство работает как master, а остальные как slave. Несколько пикосетей могут объединяться в «рассыпчатую» (scatternet) сеть. Для этого каждая пара пикосетей должна иметь общее устройство, которое будет master'ом в одной и slave'ом в другой</p> <p>3. Спроектировать иерархию классов согласно рисунку:</p>  <pre> graph TD     A[Спортсмен] --&gt; B[Лыжник]     A --&gt; C[Стрелок]     B --&gt; D[Биатлонист]     C --&gt; D </pre> <p>В числе других должен быть определен метод используемыйИнвентарь().</p> <p>4. Спроектировать иерархию классов для расчета многослойной брони. Для каждого материала известно, сколько энергии снаряда на</p>

Код индикатора	Индикатор достижения компетенции	Оценочные средства
		миллиметр толщины он поглощает
ПК-6.2	Оценивает качество разработанных алгоритмов для последующего кодирования	<p><i>Перечень теоретических вопросов</i></p> <ol style="list-style-type: none"> <li>1. Статическое или раннее связывание (static/early binding). Позднее/динамическое связывание (late/dynamic binding). Таблица виртуальных функций (virtual function table).</li> <li>2. Виртуальные функции/методы (virtual functions/methods). Абстрактные классы (abstract classes) и чистые виртуальные функции (pure virtual functions).</li> <li>3. Множественное наследование. Разрешение противоречий при наследовании одноименных членов класса.</li> <li>4. Влияние множественного наследования на механизм виртуальных функций. Область видимости класса при множественном наследовании.</li> <li>5. Виртуальное наследование.</li> <li>6. Исключения и наследование.</li> <li>7. Шаблоны (паттерны) проектирования. Основные понятия. Каталог паттернов проектирования.</li> <li>8. Паттерн «Стратегия» (Strategy).</li> <li>9. Паттерн «Наблюдатель» (Observer).</li> <li>10. Паттерн «Декоратор» (Decorator).</li> <li>11. Паттерн «Одиночка» (Singleton).</li> <li>12. Паттерн «Команда» (Command)</li> </ol> <p><i>Практические задания</i></p> <ol style="list-style-type: none"> <li>1. Спроектировать иерархию классов согласно рисунку:</li> </ol>

Код индикатора	Индикатор достижения компетенции	Оценочные средства
		<div style="text-align: center;">  <pre> graph TD     A[Транспорт] --&gt; B[Наземный транспорт]     A --&gt; C[Водный транспорт]     B --&gt; D[Амфибия]     C --&gt; D </pre> </div> <p>В числе других должен быть определен метод способПередвижения().</p> <p>2. Спроектировать иерархию классов для моделирования сети Bluetooth. Сетевые устройства могут объединяться в «пикосеть» (piconet). В каждой пикосети одно устройство работает как master, а остальные как slave. Несколько пикосетей могут объединяться в «рассыпчатую» (scatternet) сеть. Для этого каждая пара пикосетей должна иметь общее устройство, которое будет master'ом в одной и slave'ом в другой</p> <p>3. Спроектировать иерархию классов согласно рисунку:</p> <div style="text-align: center;">  <pre> graph TD     A[Спортсмен] --&gt; B[Лыжник]     A --&gt; C[Стрелок]     B --&gt; D[Биатлонист]     C --&gt; D </pre> </div> <p>В числе других должен быть определен метод используемыйИнвентарь().</p> <p>4. Спроектировать иерархию классов для расчета многослойной брони. Для каждого материала известно, сколько энергии снаряда на миллиметр толщины он поглощает</p>
ПК-6.3	Оценивает выбор программных средств для программирования и манипулирования	<i>Перечень теоретических вопросов</i>

Код индикатора	Индикатор достижения компетенции	Оценочные средства
	данными в соответствии установленными требованиями	<ol style="list-style-type: none"> <li>1. Статическое или раннее связывание (static/early binding). Позднее/динамическое связывание (late/dynamic binding). Таблица виртуальных функций (virtual function table).</li> <li>2. Виртуальные функции/методы (virtual functions/methods). Абстрактные классы (abstract classes) и чистые виртуальные функции (pure virtual functions).</li> <li>3. Множественное наследование. Разрешение противоречий при наследовании одноименных членов класса.</li> <li>4. Влияние множественного наследования на механизм виртуальных функций. Область видимости класса при множественном наследовании.</li> <li>5. Виртуальное наследование.</li> <li>6. Исключения и наследование.</li> <li>7. Шаблоны (паттерны) проектирования. Основные понятия. Каталог паттернов проектирования.</li> <li>8. Паттерн «Стратегия» (Strategy).</li> <li>9. Паттерн «Наблюдатель» (Observer).</li> <li>10. Паттерн «Декоратор» (Decorator).</li> <li>11. Паттерн «Одиночка» (Singleton).</li> <li>12. Паттерн «Команда» (Command)</li> </ol> <p><i>Практические задания</i></p> <ol style="list-style-type: none"> <li>1. Спроектировать иерархию классов согласно рисунку: <div style="text-align: center; margin-top: 10px;"> <pre> classDiagram     class Transport     class GroundTransport[Наземный транспорт]     class WaterTransport[Водный транспорт]     class Amphibia[Амфибия]     Transport &lt; -- GroundTransport     Transport &lt; -- WaterTransport     Amphibia &lt; -- GroundTransport     Amphibia &lt; -- WaterTransport </pre> </div> </li> </ol>

Код индикатора	Индикатор достижения компетенции	Оценочные средства
		<p>В числе других должен быть определен метод способПередвижения().</p> <p>2. Спроектировать иерархию классов для моделирования сети Bluetooth. Сетевые устройства могут объединяться в «пикосеть» (piconet). В каждой пикосети одно устройство работает как master, а остальные как slave. Несколько пикосетей могут объединяться в «рассыпчатую» (scatternet) сеть. Для этого каждая пара пикосетей должна иметь общее устройство, которое будет master'ом в одной и slave'ом в другой</p> <p>3. Спроектировать иерархию классов согласно рисунку:</p> <div style="text-align: center;"> <pre> classDiagram     class Спортсмен     class Лыжник     class Стрелок     class Биатлонист     Спортсмен &lt; -- Лыжник     Спортсмен &lt; -- Стрелок     Лыжник &lt; -- Биатлонист     Стрелок &lt; -- Биатлонист </pre> </div> <p>В числе других должен быть определен метод используемыйИнвентарь().</p> <p>4. Спроектировать иерархию классов для расчета многослойной брони. Для каждого материала известно, сколько энергии снаряда на миллиметр толщины он поглощает</p>
<p><b>ПК-7: Владеет способами разработки процедур интеграции программных модулей, компонент и верификации выпусков программного продукта, включая базы данных</b></p>		
ПК-7.1	Оценивает выбор программных средств для разработки и верификации интеграционного слоя автоматизированных систем	<p><i>Перечень теоретических вопросов</i></p> <p>1. Шаблоны (паттерны) проектирования. Основные понятия. Каталог паттернов проектирования.</p> <p>2. Паттерн «Стратегия» (Strategy).</p>

Код индикатора	Индикатор достижения компетенции	Оценочные средства
		<p>3. Паттерн «Наблюдатель» (Observer).</p> <p>4. Паттерн «Декоратор» (Decorator).</p> <p>5. Паттерн «Одиночка» (Singleton).</p> <p>6. Паттерн «Команда» (Command)</p> <p><i>Практические задания</i></p> <p>1. Спроектировать иерархию классов для моделирования игры в шахматы. Учесть, что пешка может превращаться в фигуру. Обеспечить смену поведения без замены объекта.</p> <p>2. Спроектировать иерархию классов для расчета гидравлического сопротивления участка трубопровода. Для каждого конструктивного элемента трубопровода известна характеристика потери давления, либо удельная (например, для прямого участка в Н/м), либо абсолютная (например, для поворота на 90 градусов в Н).</p> <p>3. Спроектировать иерархию классов для моделирования игры в шахматы. Учесть, что пешка может превращаться в фигуру. Обеспечить смену поведения без замены объекта.</p> <p>4. Спроектировать иерархию классов для расчета гидравлического сопротивления участка трубопровода. Для каждого конструктивного элемента трубопровода известна характеристика потери давления, либо удельная (например, для прямого участка в Н/м), либо абсолютная (например, для поворота на 90 градусов в Н)</p> <p>5. Спроектировать иерархию классов для моделирования штатного состава предприятия. Учесть возможность перевода работника с должности на должность.</p> <p>6. Спроектировать иерархию классов для моделирования системы ролей пользователей в СУБД. Комбинация разрешений для объекта БД индивидуальна для каждой роли. Определить метод в классе ОбъектБД,</p>



Код индикатора	Индикатор достижения компетенции	Оценочные средства
		возвращающий битовую маску разрешений для роли

## **б) Порядок проведения промежуточной аттестации, показатели и критерии оценивания:**

Промежуточная аттестация по дисциплине «Паттерное программирование» включает теоретические вопросы, позволяющие оценить уровень усвоения обучающимися знаний, и практические задания, выявляющие степень сформированности умений и владений, проводится в форме экзамена.

Экзамен по данной дисциплине проводится в устной форме по экзаменационным билетам, каждый из которых включает 1 теоретический вопрос и одно практическое задание.

### **Показатели и критерии оценивания экзамена:**

- на оценку **«отлично»** (5 баллов) – обучающийся демонстрирует высокий уровень сформированности компетенций, всестороннее, систематическое и глубокое знание учебного материала, свободно выполняет практические задания, свободно оперирует знаниями, умениями, применяет их в ситуациях повышенной сложности.
- на оценку **«хорошо»** (4 балла) – обучающийся демонстрирует средний уровень сформированности компетенций: основные знания, умения освоены, но допускаются незначительные ошибки, неточности, затруднения при аналитических операциях, переносе знаний и умений на новые, нестандартные ситуации.
- на оценку **«удовлетворительно»** (3 балла) – обучающийся демонстрирует пороговый уровень сформированности компетенций: в ходе контрольных мероприятий допускаются ошибки, проявляется отсутствие отдельных знаний, умений, навыков, обучающийся испытывает значительные затруднения при оперировании знаниями и умениями при их переносе на новые ситуации.
- на оценку **«неудовлетворительно»** (2 балла) – обучающийся демонстрирует знания не более 20% теоретического материала, допускает существенные ошибки, не может показать интеллектуальные навыки решения простых задач.
- на оценку **«неудовлетворительно»** (1 балл) – обучающийся не может показать знания на уровне воспроизведения и объяснения информации, не может показать интеллектуальные навыки решения простых задач.