Министерство образования и науки Российской Федерации

Российский университет дружбы народов

Инженерный факультет

Кафедра Кибернетики и мехатроники

Программа дисциплины

Объектно-ориентированное программирование

Рекомендуется для направления подготовки

27.03.04 – «Управление в технических системах»

Квалификация выпускника «бакалавр»

Программу разработал

доктор техн. наук, профессор Евгений Иванович

Забудский web-site http://zabudsky.ru/ , e-mail zei@inbox.ru

Москва

Аннотация

В процессе изучения дисциплины рассматриваются следующие вопросы**:** объектно-ориентированный анализ (*OOA*), объектно-ориентированное проектирование (*OOD*), объектно-ориентированное программирование (*ООП*), *шаблоны* проектирования, унифицированный язык моделирования *UML* (**U**nified **M**odeling **L**anguage), а также *спец*вопросы объектно-ориентированного языка программирования C# (*C_Sharp*), касающиеся парадигмы *ООП*, etc.

В основе всех этих вопросов лежит один и тот же фундамент**:** *способность и необходимость мыслить категориями объектов реального мира*, так как специалисту-программисту необходимо разрабатывать Windows-приложения, эмулирующие те или иные *системы реального мира*. Поэтому изучение концепции объектного подхода не заканчивается изучением отдельно взятого метода или набора средств разработки. Иными словами, объектный подход является образом *объектно-ориентированного мышления*, которому также обучаются студенты.

Переходить на новый способ мышления всегда непросто, поэтому вербальный метод обучения сопровождается активным привлечением компьютерных и информационных технологий. Это позволяет сопровождать рассуждения о концепциях объектов демонстрацией и анализом соответствующих фрагментов программного кода, а также иллюстративной графики.

Особое внимание уделяется организации самостоятельной работы студентов и ее методическому обеспечению.

С этой целью автором программы разработан *Учебно-методический комплекс* дисциплины «Объектно-ориентированное программирование», который *помещен на сайт автора* / URL-адрес http://zei.narod.ru /. Комплекс включает: методические рекомендации преподавателям и студентам, программу дисциплины и нормативно-справочные материалы, материалы к лекциям, материалы к практическим занятиям, темы и рекомендации по выполнению Домашнего задания и др. Все материалы комплекса доступны студентам и используются ими в процессе обучения.

1. Цели и задачи дисциплины:

Основной целью дисциплины является формирование понимания идеологии и ключевых аспектов объектно-ориентированного программирования (ООП) на языке C#, достаточного для практического использования в процессе дальнейшего обучения и в профессиональной сфере.

Основная задача дисциплины – научить студентов разрабатывать в соответствии с парадигмой компонентно-ориентированного программирования компьютерные модели реальных и концептуальных систем, соответствующих направлению «Автоматизация и управление».

2. Место дисциплины в структуре ООП:

освоение дисциплины «Объектно-ориентированное программирование» (входит в цикл ОПД) предполагает знания, умения и компетенции студентов по дисциплинам: «Информатика» и «Программирование и основы алгоритмизации". Дисциплина «Объектно-ориентированное программирование» является предшествующей для дисциплин «Прикладные программные системы» и «Базы данных в информационно-управленческих системах».

3. Требования к результатам освоения дисциплины:

Процесс изучения дисциплины направлен на формирование следующих компетенций:

- способность владеть основными методами, способами и средствами получения, хранения, переработки информации, иметь навыки работы с компьютером как средством управления информацией;
- способность проводить вычислительные эксперименты с использованием стандартных программных средств с целью получения компьютерных и математических моделей процессов, систем и объектов автоматизации и управления

В результате изучения дисциплины студент должен:

Знать: различные парадигмы разработки программных продуктов в историческом контексте; методологию объектно-ориентированного программирования;

Уметь: разрабатывать компьютерные модели реальных и концептуальных систем на основе парадигмы компонентно-ориентированного программирования;

Владеть*:* навыками работы с современными аппаратными и программными средствами анализа, проектирования и разработки систем управления

4. Объем дисциплины и виды учебной работы

Общая трудоемкость дисциплины составляет 2 зачетные единицы.

Вид учебной работы	Всего часов	Семестры			
		6	-	-	-
Аудиторные занятия (всего)	72	72	-	-	-
В том числе:	-	-	-	-	-
Лекции	36	36	-	-	-
Практические занятия (ПЗ)	-	-	-	-	-
Семинары (С)	-	-	-	-	-
Лабораторные работы (ЛР)	36	36	-	-	-
Самостоятельная работа (всего)	48	48	-	-	-
В том числе:	-	-	-	-	-
Курсовой проект (работа)	-	-	-	-	-
Расчетно-графические работы	-	-	-	-	-
Реферат	-	-	-	-	-
Другие виды самостоятельной работы	48	48	-	-	-
Вид промежуточной аттестации	экзамен	экзамен	-	-	-
Общая трудоемкость час	72	72	-	-	-
зач. ед.	2	2	-	-	-

5. Содержание дисциплины

5.1. Содержание разделов дисциплины

Раздел 1. *Введение. Методология разработки объектно-ориентированного программного обеспечения*

Объектно-ориентированное мышление. Принципы объектно-ориентированного подхода. Объектно-ориентированное программирование в историческом контексте. *Шесть этапов* объектно-ориентированной методологии: *Этап предпроектных исследований* – Учет технических, временных и финансовых ограничений. Определение целесообразности продолжения работы над приложением; *Этап анализа* – Сбор информации, необходимой для продолжения работы; *Этап проектирования* – Создание схемы интерфейса и структуры программы, без создания каких-либо фактических программ; *Этап разработки* (*реализации*) – Создание приложения, включая все элементы интерфейса и программного кода; *Этап внедрения* – Применение и тестирование программы; *Этап опытной эксплуатации* – Совершенствование продукта, призванное устранить возможные проблемы или удовлетворить новые запросы.

Демонстрация реализации этапов объектно-ориентированной методологии на примере разработки проекта «Расчет оценки студента»: *C_Sharp*-программа в вариантах ***GUI*** (***G****raphical* ***U****ser* ***I****nterface*) и консольном.

Раздел 2. *Объектно-ориентированный анализ и проектирование: основные понятия и терминология. Цели анализа и проектирования. Сопоставление ОО языков программирования C# и Java*

Основные понятия, терминология и цель (результат) ОО анализа. Основные понятия, терминология и цель (результат) ОО проектирования. Сопоставление синтаксиса и семантики ОО языков программирования C# и Java.

Раздел 3. *Инкапсуляция – центральное понятие объектно-ориентированного программирования*

Инкапсуляция – объектно-ориентированная характеристика модульности. Внешний интерфейс и внутренняя реализация инкапсулированного программного объекта. Характерные признаки эффективной инкапсуляции: абстракция, общедоступный интерфейс и сокрытие реализации.

Демонстрация и анализ концепций инкапсуляции в *C_Sharp*-программе «Объектно-ориентированный Банк» и др.

Раздел 4. *Наследование – базовое понятие объектно-ориентированного программирования*

Наследование – механизм, дающий возможность создавать новый класс на основе уже существующего класса. Базовый и производный классы. Абстрактные метод и класс. Наследование реализации, поведения и свойства. Переопределение метода. Типы наследования. Множественное наследование: проблемы (*the diamond problem*) и решения (*interface* – особый абстрактный класс). Сравнение отношений «*Is-a*» («Является») и «*Has-a*» («Содержит»)**:** когда использовать наследование?

Демонстрация и анализ концепций наследования в *C_Sharp*-программе «Банковский счет» и др.

Раздел 5. *Полиморфизм – базовое понятие в парадигме объектно-ориентированного программирования*

Полиморфизм – самое радикальное, универсальное средство – *одно* имя класса или метода представляет *различный*, выбранный *автоматическим* механизмом, *программный код* (полиморфизм – одно имя представляет различные поведения). Связь полиморфизма с инкапсуляцией и наследованием. Формы полиморфизма: полиморфизм включения, параметрический полиморфизм, переопределение метода, перегрузка метода. Ранее связывание (во время компиляции) и позднее связывание (во время выполнения).

Демонстрация и анализ концепций полиморфизма в модифицированной *C_Sharp*-программе «Банковский счет», в *C_Sharp*-программе «Система лифтов здания» и др.

Раздел 6. *Основы UML – унифицированного языка моделирования объектно-ориентированных систем*

U**nified **M**odeling **L**anguage* (UML***) – язык графического моделирования, используемый для представления объектно-ориентированных программ. Краткая история *UML*. Псевдокод – представление алгоритма в пределах метода. Система обозначений языка *UML* для описания отношений классов и общей архитектуры программы. Моделирование отношений между классами: зависимость, ассоциация, агрегация, композиция, обобщение. Интерактивный пакет *Rational Rose* – использование языка *UML на стадии проработки проекта.* Среда *MS VS .NET* **2005** – генерация *UML*-диаграмм классов *после разработки проекта и написания программного кода*.

Демонстрация и анализ *UML*-диаграмм классов проектов «Банковский счет», «Служащие с разной формой оплаты**:** начисление зарплаты», «Расчет оценки студента» и «Система лифтов здания».

Раздел 7. *Основы объектно-ориентированного анализа*

Итеративная технология разработки программного обеспечения. Объектно-ориентированный анализ (ООА) –это объектно-ориентированный подход к осмыслению разрабатываемого проекта. Результат ООА – понимание предметной области, формулировка технических требований к системе в терминах классов и взаимодействий между объектами (система – множество взаимодействующих объектов). Модель прецедентов – модель способов взаимодействия пользователей с системой. Концептуальная модель (модель предметной области) – скелет создаваемой системы. Паралич анализа (*analysis paralysis*).

Демонстрация реализации этапа объектно-ориентированного анализа на примере разработки проекта «Расчет оценки студента» («Система лифтов здания».) : *C_Sharp*-программа в вариантах ***GUI*** (***G**raphical **U**ser **I**nterface*) и консольном.

Раздел 8. *Основы объектно-ориентированного проектирования*

Объектно-ориентированное проектирование (OOD) – итеративный процесс, определяющий назначение объектов и взаимодействие между ними. Пять шагов OOD: *шаг 1* – создание исходного списка объектов; *шаг 2* – определение и уточнение назначений объектов при помощи карточек ***CRC*** (карточки связи и взаимодействия между классами – ***C**lass **R**esponsibility **C**ollaboration cards*); *шаг 3* – разработка точек взаимодействия; *шаг 4* – детализация отношений между объектами; *шаг 5* – построение объектной модели. Объектная модель – описание объектной структуры и отношения между объектами. Паралич проектирования (*design paralysis*).

Раздел 9. *Объектно-ориентированный подход к созданию пользовательского интерфейса*

Формы пользовательского интерфейса (*User Interface*). Значение развязки пользовательских интерфейсов: не «навешивать» интерфейс на систему после ее создания. Развязка пользовательского интерфейса с помощью шаблона проектирования Модель/Вид/Контроллер (***M**odel **V**iew **C**ontroller* – ***MVC***) – основная система полностью независима от интерфейса.

Демонстрация и анализ пользовательского интерфейса проекта «Расчет оценки студента» («Система лифтов здания».): *C_Sharp*-программа в вариантах ***GUI*** (***G**raphical **U**ser **I**nterface*) и консольном.

Раздел 10. *Разработка компьютерных моделей реальных и концептуальных систем на основе методологии компонентно-ориентированного программирования*

Объектно-ориентированный анализ: выявление прецедентов – способов взаимодействия пользователей с системой; определение сценариев – последовательности событий для каждого прецедента; построение диаграммы прецедентов – диаграмма последовательности событий, диаграмма сотрудничества; построение концептуальной модели и словаря предметной области. *Объектно-ориентированное проектирование*: создание исходного списка объектов (шаг 1-й); определение назначения объектов (использование карточек *CRC*) (шаг 2-й); определение точек взаимодействия – мест обращения объектов друг к другу (шаг 3-й); детализация отношений между объектами (шаг 4-й); построение объектной модели – диаграммы классов и взаимодействий между ними (шаг 5-й). *Разработка кода системы и интерфейса.* Реализация *итеративной технологии* на всех этапах разработки.

5.2 Разделы дисциплины и междисциплинарные связи с обеспечиваемыми (последующими) дисциплинами

№ п/п	Наименование обеспечиваемых (последующих) дисциплин	№ № разделов данной дисциплины, необходимых для изучения обеспечиваемых (последующих) дисциплин									
		1	2	3	4	5	6	7	8	9	10
1	Прикладные программные системы	+	+	+	+	+	+	–	–	+	+
2	Базы данных в информационно-управленческих системах	–	–	+	+	+	+	+	+	+	+

5.3. Разделы дисциплин и виды занятий

№ п/п	Наименование раздела дисциплины	Лекц.	Практ. зан.	Лаб. зан.	Семин	СРС	Всего часов
1	Введение. Методология разработки объектно-ориентированного программного обеспечения	2	–	2	–	4	8
2	Объектно-ориентированный анализ и проектирование: основные понятия и терминология. Цели анализа и проектирования. Сопоставление ОО языков программирования C# и Java	2	–	2	–	4	8
3	Инкапсуляция – центральное понятие объектно-ориентированного программирования	4	–	4	–	4	12
4	Наследование – базовое понятие объектно-ориентированного программирования	4	–	4	–	4	12
5	Полиморфизм – базовое понятие в парадигме объектно-ориентированного программирования	4	–	4	–	4	12
6	Основы UML – унифицированного языка моделирования объектно-ориентированных систем	4	–	4	–	4	12
7	Основы объектно-ориентированного анализа	4	–	4	–	4	12
8	Основы объектно-ориентированного проектирования	4	–	4	–	4	12
9	Объектно-ориентированный подход к созданию пользовательского интерфейса	2	–	2	–	4	8
10	Разработка компьютерных моделей реальных и концептуальных систем на основе методологии компонентно-ориентированного программирования	6	–	6	–	4	16

6. Лабораторный практикум

№ п/п	№ раздела дисциплины	Наименование лабораторных работ	Трудо-емкость (час.)
1	1	Работа в среде Microsoft Visual Studio .NET: интерфейс среды	2
2	2	Работа в среде MS VS .NET: компоненты среды	2
3	3	Работа в среде MS VS .NET: разработка программ с учетом инкапсуляции	4
4	4	Работа в среде MS VS .NET: разработка программ с учетом инкапсуляции и наследования	4
5	5	Работа в среде MS VS .NET: разработка программ с учетом инкапсуляции, наследования и полиморфизма	4
6	6	Работа в среде MS VS .NET: язык **U**nified **M**odeling **L**anguage – построение диаграмм классов, etc.	4
7	7	Объектно-ориентированный анализ предметной области – **1**-й этап компьютерного моделирования системы	4
8	8	Объектно-ориентированное проектирование – **2**-й этап компьютерного моделирования системы	4
9	9	Работа в среде MS VS .NET: разработка **G**raphical **U**ser **I**nterface на основе шаблона **M**odel-**V**iew-**C**ontroller	2
10	10	Работа в среде MS VS .NET: разработка компьютерных моделей реальных и концептуальных систем на основе методологии компонентно-ориентированного программирования	6

7. Практические занятия (семинары) –

практические занятия (семинары) Рабочим учебным планом на 2010-2011 учебный год *не предусмотрены*

8. Примерная тематика курсовых проектов (работ) –

курсовой проект (работа) Рабочим учебным планом на 2010-2011 учебный год *не предусмотрен*

9. Учебно-методическое и информационное обеспечение дисциплины:

а) основная литература

1. Мацяшек, Лешек А. Анализ и проектирование информационных систем с помощью UML. Пер. с англ. – М.: ООО «И. Д. Вильямс», 2008.
2. Троелсен Э. Язык программирования C# 2010 и платформа .NET 4.0. Пер. с англ. – С.-Пб: ООО «И. Д. Вильямс», 2010.
3. Забудский Е.И. *Учебно-методический комплекс* дисциплины «Объектно-ориентированное программирование». М.: Кафедра АПС ГУ-ВШЭ. Кафедра КиМ РУДН, 2005 … н/в. Internet-ресурс: ; – http://zabudsky.ru/New_Web_Page_ZEI.pdf .

б) дополнительная литература

1. Грэди Буч, Майкл У. Энгл, Роберт А. Максимчук. Объектно-ориентированный анализ и проектирование с примерами приложений. Пер. с англ. – М.: ООО «И. Д. Вильямс», 2008.
2. Гамма Э. Приемы объектно-ориентированного проектирования. Паттерны проектирования. С.-Пб: Изд «Питер», 2010.
3. Кватрани Т. Визуальное моделирование с помощью IBM Rational Software Architect и UML М.: Изд-во: КУДИЦ-ПРЕСС, 2007.

дополнительные источники информации – Internet-ресурсы

4. Новые книги раздела **C#** – http://books.dore.ru/bs/f6sid16.html .
5. **C#** и ***.NET*** по шагам – http://www.firststeps.ru .
6. ***UML*** – язык графического моделирования – http://www.uml.org/ .
7. ***NUnit, JUnit*** – каркасы тестирования для испытания классов – http://www.junit.org , http://www.nunit.org .
8. Пакет объектного моделирования ***Rational Rose*** – http://www-306.ibm.com/software/rational/ .
9. Steve Burbeck "Applications Programming in Smalltalk-80(TM): How to use **M**odel-**V**iew-**C**ontroller (**MVC**)" – http://st-www.cs.uiuc.edu/users/smarch/st-docs/mvc.html .

в) программное обеспечение

1. Microsoft Visual Studio .NET,
2. IBM Rational Rose,
3. Microsoft Office Visio, etc.

г) базы данных, информационно-справочные и поисковые системы

Библиотека MSDN (**M**icro**S**oft **D**evelopers **N**etwork)

1. http://msdn.microsoft.com/ru-ru/default.aspx ,
2. http://msdn.microsoft.com/en-en/default.aspx ,
2. http://www.gotdotnet.ru .

10. Материально-техническое обеспечение дисциплины:

Компьютерные классы кафедры Кибернетика и мехатроника.

11. Методические рекомендации по организации изучения дисциплины:

11.1. Тематика аудиторных (*текущий контроль успеваемости*) контрольных работ

(Содержанием контрольных работ является разработка программных продуктов на компьютере по тематике пройденного материала)

1. Абстракция – учимся думать и программировать абстрактно.
2. Эффективное применение инкапсуляции в объектно-ориентированном программировании.
3. Открытый интерфейс и эффективное сокрытие реализации.
4. Использование концепции наследования при объектно-ориентированном программировании.
5. Применение простого наследования.
6. Использование абстрактных классов при наследовании.
7. Адаптация объектно-ориентированных программ к изменяющимся требованиям средствами полиморфизма.
8. Применение полиморфизма включения.
9. Переопределение – важный тип полиморфизма.
10. Перегрузка – частный случай полиморфизма.
11. Необходимое условие эффективного полиморфизма – эффективное применение инкапсуляции и наследования.
12. Построение UML-диаграмм классов программных продуктов, разрабатываемых в домашних и аудиторных работах.
13. Анализ прецедентов – случаев взаимодействия пользователя с системой – при выполнении объектно-ориентированного анализа с целью уяснения смысла задач, разрабатываемых в домашних и аудиторных работах.
14. Построение концептуальной модели – выявление объектов предметной области, необходимых для адекватного описания системы – при выполнении объектно-ориентированного анализа с целью уяснения смысла задач, разрабатываемых в домашних и аудиторных работах.
15. Построение объектной модели – установление взаимосвязей и структуры объектов – при выполнении объектно-ориентированного проектирования с целью приближения к искомой конструкции систем, разрабатываемых в домашних и аудиторных работах.
16. Практическое применение шаблонов при выполнении объектно-ориентированного программирования.
17. Программирование пользовательского интерфейса на основе объектно-ориентированного подхода в задачах, разрабатываемых в домашних и аудиторных работах.
18. Реализация форм тестирования программного кода задач, разрабатываемых в домашних и аудиторных работах.

11.2. Темы Домашних заданий

1. Компьютерное моделирование работы банка
2. Компьютерное моделирование работы и облуживания банкомата
3. Компьютерное моделирование работы четырех светофоров на перекрестке
4. Компьютерное моделирование системы лифтов здания
5. Компьютерное моделирование работы склада
6. Компьютерное моделирование работы отдела кадров
7. Компьютерное моделирование работы Дома Мод
8. Компьютерное моделирование Салона Красоты
9. Компьютерное моделирование игровой ситуации – компьютерная игра
10. Компьютерное моделирование концептуальной системы тестирования знаний
11. Компьютерное моделирование системы определения цены на новый товар
12. Компьютерное моделирование электронного магазина, специализирующегося на комиссионной торговле
13. Компьютерное моделирование электронного магазина с организацией лотерей
14. Компьютерное моделирование электронного магазина розничной торговли по оптовым ценам
15. Компьютерное моделирование концептуальной системы анализа поведения посетителей супермаркета
16. Компьютерное моделирование системы оплаты заказа с использованием терминала автоматизированной системы
17. Авторские темы, предлагаемые студентами в соответствии с содержанием дисциплины

11.3. Указания студенту

1. Обязательно посещать лекции ведущего преподавателя; лекции – основное методическое руководство при изучении дисциплины, наиболее оптимальным образом структурированное и скорректированное на современный материал; в лекции глубоко и подробно, аргументировано и методологически строго рассмотрены главные проблемы темы; в лекциях даны необходимые разные подходы к исследуемым проблемам.

2. Готовиться и активно работать на лабораторных занятиях; подготовка к лабораторным занятиям включает самостоятельную проработку материалов лекций и материалов практических занятий, рекомендованной учебной литературы /см. Учебно-методический комплекс (УМК) дисциплины «Объектно-ориентированное программирование»; Разд. 9, а) основная литература, 3 /.

3. Обратить особое внимание на систематическое и последовательное выполнение Домашнего задания, которое является концентрированным выражением качества усвоения дисциплины “Объектно-ориентированное программирование”. Содержанием Домашнего задания (см. темы в разд. 11.2) является разработка в соответствии с парадигмой компонентно-ориентированного программирования компьютерной модели реальной (концептуальной) системы. Далее приводится обязательное Содержание (Оглавление) оформленного Домашнего задания, которое представляется преподавателю на проверку в конце семестра, до экзамена.

Содержание

	Введение (охарактеризовать парадигму объектно-ориентированного программирования)
1.	Описание предметной области
2.	Цель компьютерного моделирования реальной (концептуальной) системы
3.	**Объектно-ориентированный анализ** (см. в УМК лекцию **12**, раздел **3**) *
3.1	Использование прецедентов для определения возможного применения системы
3.1.2	Отождествление действующих субъектов
3.1.3	Создание списка прецедентов
3.1.4	Определение последовательности событий для КАЖДОГО прецедента
3.1.5	Моделирование прецедентов (см. в УМК лекцию **12** и Дополнение к ней) *
3.1.5.1	Диаграммы прецедентов
3.1.5.2	Диаграммы взаимодействия
3.1.5.2.1	Диаграммы последовательности событий
3.1.5.2.2	Диаграммы сотрудничества
3.1.5.3	Диаграммы **…** /по мере необходимости/
3.6	Построение ***концептуальной*** **модели**
4.	**Объектно-ориентированное проектирование** (см. в УМК лекцию **14**, раздел **1**) *
4.1	Создание исходного списка объектов
4.2	Определение назначения объектов
4.3	Определение точек взаимодействия
4.4	Детализация отношений между объектами
4.5	Построение ***объектной*** **модели**
5	**Разработка кода** в соответствии с парадигмой *компонентно*-ориентированного программирования (см. в УМК прак. зан. **4**, раздел **5; см.** в УМК лекцию **12**, раздел **1**) *
5.1	Реализация программы в формате **GUI** на основе шаблона «**М**одель-**В**ид_**К**онтроллер» (см. в УМК лекцию **15**) *
5.2	UML-диаграмма взаимодействия классов (среда MS VS .NET)
6	**Результаты работы программы**
	Литература

*См. Учебно-методический комплекс дисциплины: Internet-ресурс – http://zei.narod.ru/New_Web_Page_ZEI.pdf

11.4. Вопросы для оценки качества освоения дисциплины (*промежуточная аттестация*)

# пп	Экзаменационный вопрос *
1	Концепция и технологии .NET
2	Парадигма объектно-ориентированного программирования и ее предшественники
3	Терминология объектно-ориентированного программирования: класс, объект, переменные экземпляра, метод, интерфейс, реализация, поведение, etc.

Продолжение раздела 11.4

4	Три базовых понятия парадигмы объектно-ориентированного программирования
5	Инкапсуляция: абстракция, интерфейс и реализация
6	Инкапсуляция: средства защиты и доступа
7	Наследование: отношения **“**Is_A” и “Has-A”. Наследование для многократного использования реализации и наследование для отличия.
8	Типы наследования: простое наследование.
9	Типы наследования: многоуровневое наследование.
10	Типы наследования: множественное наследование и «проблема бриллианта»
11	Интерфейсы в C# – аналог множественного наследования
12	Стандартные интерфейсы в объектно-ориентированном языке программирования C#
13	Абстрактные классы и методы
14	Формы полиморфизма: полиморфизм включения
15	Формы полиморфизма: полиморфизм посредством переопределение методов
16	Формы полиморфизма: полиморфизм посредством перегрузки методов
17	Раннее и позднее (динамическое) связывание. Полиморфизм времени выполнения
18	Парадигма компонентно-ориентированного программирования: компоненты и клиенты
19	Основные стандартные классы библиотеки System.Windows.Forms и простанство имен System.Drawing
20	Стандартный класс System.Delegate и использование делегатов и событий
21	Реализация обработчика событий в C#-программах, управляемых событиями
22	Оконное Windows-приложение с основными элементами управления на форме: создание приложения в Visual Studio .NET и компиляция в интегрированной среде разработки
23	Оконное Windows-приложение с основными элементами управления на форме: разработка C#-программы в редакторе, компиляция в командной строке и компиляция в интегрированной среде разработки
24	Анатомия классов и их разработка в парадигме объектно-ориентированного программирования
25	Основы языка моделирования (UML) для графического представления объектно-ориентированного программного обеспечения
26	Стадии разработки объектно-ориентированных компьютерных моделей реальных и концептуальных систем
27	Основы объектно-ориентированного анализа: прецеденты и сценарии
28	Основы объектно-ориентированного анализа: диаграммы прецедентов, диаграммы взаимодействия, диаграммы активности
29	Основы объектно-ориентированного анализа: концептуальная модель – скелет разрабатываемой системы
30	Основы объектно-ориентированного проектирования: использование карточек CRC (Class Responsibility Collaboration) для определения назначения и связи объекта
31	Основы объектно-ориентированного проектирования: объектная модель разрабатываемой системы и ее значение для написания кода
32	Объектно-ориентированный подход к программированию пользовательского интерфейса

* На экзамене студент в письменной форме дает ответ на вопрос и решает задачу на компьютере

Программу разработал

профессор кафедры Кибернетика и мехатроника
доктор техн. наук, профессор Е.И. Забудский