МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Юго-Западный государственный университет»
(ЮЗГУ)

Кафедра программной инженерии

УТВЕРЖДАЮ
Проректор по учебной работе
_____О.Г.Локтионова
«__» _____2017 г.

Языки объектно-ориентированного программирования

Методические указания к выполнению курсовой работы

Курск 2017

УДК 681.3

Составитель Е.И.Аникина

Рецензент
Кандидат технических наук, доцент кафедры программной инженерии *Н.Н. Бочанова*

Языки объектно-ориентированного программирования: методические указания к выполнению курсовой работы/Юго-Зап. гос. ун-т; сост. Е.И.Аникина. Курск, 2017. 18 с.

Содержат общую формулировку заданий к курсовой работе, методические рекомендации по выполнению задания, а также требования к содержанию и оформлению пояснительной записки к курсовой работе.

Предназначены для студентов направления подготовки 09.03.04 «Программная инженерия».

Текст печатается в авторской редакции.

Подписано в печать . Формат 60х84 1/16.
Усл. печ. л. . Уч.-изд. л. . Тираж 100 экз. Заказ .
Бесплатно.

Юго-Западный государственный университет
305040, Курск, ул.50 лет Октября, 94.

1. Цели курсового проектирования

Важным фактором подготовки специалистов в области информационных технологий является умение программировать, используя современные языки, включающие объектные возможности, знакомство с основными методами и современными технологиями программирования, в том числе с использованием объектных библиотек конкретных языков.

Целью курсового проектирования является развитие данных навыков d ходе решения конкретной практической задачи – разработки оконного приложения для тестирования спроектированной системы классов. При этом студентам предоставляется право выбора конкретного языка и средства разработки, а также возможность выполнения работы в подгруппах. Целью такого подхода является развитие способностей к обоснованному принятию самостоятельных решений в ходе проектной работы и получение базового опыта разработки информационных систем в проектных группах.

2. Тематика курсовой работы

В ходе выполнения курсовой работы студенты должны практически освоить общий методологический подход, используемый при проектировании и программной реализации системы классов, соответствующей объектно-ориентированной парадигме программирования.

Создаваемая система классов описывает (моделирует) определённую предметную область и может служить основой для полноценной информационной системы, решающей задачи данной области. Спроектированная система классов должна быть не только реализована в виде программы на одном из Си-подобных объектно-ориентированных языков, но и протестирована. Для тестирования классов студент должен разработать оконное приложение, функциональные возможности

которого должны демонстрировать различные варианты использования объектов созданных классов.

3. Задание на курсовое проектирование

В ходе курсовой работы группе студентов из 2-3 человек необходимо разработать оконное приложение, тестирующее систему (библиотеку) классов, спроектированную и реализованную студентами для решения конкретной задачи из некоторой предметной области. При этом подлежат разработке следующие вопросы:

а) анализ задания;

б) разработка библиотеки классов;

в) разработка тестового приложения;

г) оформление пояснительной записки по результатам выполнения работы.

Открытый список вариантов задания на курсовое проектирование приведён в Приложении А. Каждый из вариантов определяет предметную область, для моделирования которой должна быть разработана система классов, а также рекомендуемое количество участников проектной группы. Допустим выбор иных вариантов тем курсовой работы по предложениям преподавателя, работодателей, студентов.

Объём пояснительной записки составляет 20-50 страниц машинописного текста, включая рисунки, таблицы и приложения. Её оформление должно соответствовать требованиям ГОСТ, ЕСКД, ЕСПД.

4. Содержание пояснительной записки к курсовой работе

1. Введение.
2. Описание системы. Постановка задачи
3. Объектно-ориентированный анализ и проектирование системы на языке UML

3.1. Диаграммы классов(Class Diagram)

3.2. Диаграммы последовательности (Sequence Diagram)
3.3. Диаграмма состояний (Statechart Diagram)
4.Программная реализация на языке C#.
4.1.Краткая характеристика инструментальной программной среды
4.1.Структура программы
4.2.Описание программных модулей
4.3.Инструкция пользователю
5. Методика и результаты тестирования программы.
Список литературы.

5. Рекомендации по выполнению работы

5.1. Порядок выполнения

1. Анализ задания. Описание предметной области.
2. Разработка библиотеки классов:
   2.1. Определение набора классов, их свойств и методов, иерархии наследования.
   2.2. Формализация описания классов в виде диаграммы классов.
   2.3. Выбор средств реализации библиотеки классов – языка программирования и среды разработки.
   2.4. Программная реализация классов.
3. Разработка оконного приложения для тестирования системы классов:
   3.1. Определениеметодики тестирования для демонстрации использования всех методов созданных классов.
   3.2. Проектирование тестового оконного приложения.
   3.3. Программная реализация тестового приложения.
4. Тестирование системы классов. Документируемая демонстрация работы тестового приложения.
5. Подготовка расчётно-пояснительной записки.
6. Демонстрация работы программы преподавателю и защита курсовой работы.

5.2. Анализ задания

Основой курсовой работы является некоторая предметная область со своими терминами, понятиями, объектами, отношениями между этими объектами. Очевидно, что специалист в области информационных технологий далеко не всегда является специалистом в той сфере, для которой он выполняет разработку информационной системы. Поэтому залогом успешного решения поставленной перед ним задачи является подробный и качественный анализ всех аспектов той пользовательской среды, в которой будет функционировать создаваемое программное приложение или информационная система.

В ходе анализа предметной области необходимо на основе знакомства с литературными источниками и общения с заказчиком выявить:

1. Чему посвящена предметная область, какие в ней есть термины и понятия, субъекты и объекты, способы взаимодействия субъектов, способы использования объектов, закономерности. Например, если речь идёт о графических примитивах в трёхмерном пространстве, то следует выявить список возможных примитивов (точка, линия, прямоугольник, параллелепипед, шар и т.п.), способы их описания (так, для точки достаточно указать её координаты, а для шара необходимо знать координаты центра и радиус), возможные способы преобразования (перемещение, масштабирование, поворот и т.п.).
2. Что входит в словарь предметной области, отдельно выделив список существительных и список глаголов, которые могут быть связаны с существительными. Для графических примитивов существительными могут быть: «точка», «координата», «шар», «угол», «цвет», «длина», «ширина» и др. А в качестве глаголов можно указать: «нарисовать», «повернуть», «масштабировать», «переместить».

3. Каковы функциональные требования к разрабатываемой информационной системе. Основой их служат потребности заказчика, однако разработчик должен оценить возможность реализации требований, исходя из технических возможностей и
имеющихся ресурсов.

Результат анализа должен быть формализован. В реальной ситуации обычно оформляется протокол обсуждения, заключается договор, формулируется техническое задание. Все документы заверяют полномочные представители заказчика и разработчика. Во избежание конфликтных ситуаций следует задокументировать все решения, принятые по спорным моментам.

В рамках курсовой работы в роли заказчика выступает преподаватель, выдавший задание (либо представитель работодателя, если задание было сформулировано им). Студент проводит анализ предметной области, основываясь на своих собственных знаниях, литературных источников и в ходе общения с преподавателем. Результат должен быть оформлен в виде реферативного описания предметной области. Из этого описания должен логически следовать словарь предметной области, состоящий из списка существительных и глаголов. Именно он послужит основой следующего этапа работы.

5.3. Проектирование системы классов

Проектирование системы классов начинается с обработки словаря предметной области. Эта обработка состоит в выявлении того, какие слова соответствуют объектам, классам, свойствам и методам. Список существительных служит основой для выделения классов и их свойств, а список глаголов – для определения методов.

Для приведённого выше примера можно указать следующее соответствие:

- классы: точка, шар;
- свойства: координата, угол, цвет, длина, ширина;
- методы: нарисовать, повернуть, масштабировать, переместить.

Следующий шаг является, фактически, завершающим на этапе проектирования классов. Он состоит в том, чтобы определить, какой из классов какие свойства и методы содержит. Следует обратить внимание на то, что наборы свойств и методов у разных классов могут «пересекаться». Например, и для класса «точка», и для класса «шар» справедливо наличие методов «нарисовать», «масштабировать», «переместить». В то же время, метод «повернуть» не имеет смысла по отношению к объектам данных классов, зато может присутствовать у класса «параллелепипед».

Ещё одним вопросом, требующим решения на данном шаге, является выявление отношений между классами. Речь идёт об отношениях наследования и включения. Следует обратить внимание, что понятие «наследование» чаще всего возникает тогда, когда разные классы обладают частично схожими наборами свойств и методов. При составлении словаря предметной области далеко не всегда в список могут попасть понятия, которым можно сопоставить базовые классы в иерархии наследования. Поэтому следует внимательно проанализировать список классов, свойств, методов, их соответствие, и, возможно, выделить ряд новых классов, связанных с имеющимися отношениями наследования и включения. На данном этапе можно уже учитывать не только законы предметной области, но и такие принципы объектно-ориентированного подхода как абстракция, инкапсуляция, полиморфизм.

В рассматриваемом примере можно выделить абстрактный класс «фигура» со свойствами «абсцисса», «ордината», «аппликата», «цвет» и методом «нарисовать». Классы «точка» и

«шар» будут являться наследниками класса «фигура», а метод «нарисовать» может являться виртуальным, что даёт нам полиморфический кластер, включающий три класса.

Результаты такого анализа должны быть оформлены в виде диаграммы классов. Предпочтительным является использование нотации

языка UML. В частности, следует придерживаться следующих правил: • класс обозначается прямоугольником; • прямоугольник делится на три части, в каждой из которых,

соответственно, указываются: имя класса, список свойств, список

методов; • имена классов, свойств и методов могут быть записаны на русском

языке, но в соответствии с нормами написания стандартных идентификаторов (одно слово, включающее буквы, цифры, символ

подчёркивания и не начинающееся с цифры); • имена классов записываются с заглавной буквы, имена свойств и

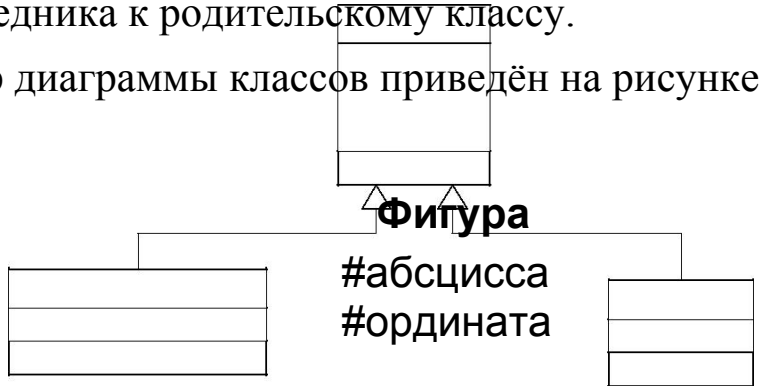
методов – со строчной; • перед именем свойства или метода ставится символ, указывающий

на режим доступности: закрытый (-), защищённый (#), открытый

(+);

- после имени метода ставятся круглые скобки, в которых могут быть перечислены параметры метода;
- наследование классов обозначается стрелкой с треугольным незакрашенным наконечником;
- стрелка при наследовании направляется от класса-наследника к родительскому классу.

Пример диаграммы классов приведён на рисунке 1.



#аппликата
#цвет
+нарисовать()

Точка

-видимый_размер_точки

Шар

-радиус

Рисунок 1 – Диаграмма классов трёхмерных графических примитивов

Диаграмма классов с комментариями к ней является результатом данного этапа работы.

5.4. Выбор средств реализации библиотеки классов

После завершения проектирования библиотеки классов можно приступать к её программной реализации. Однако этот процесс невозможен без предварительного решения ряда технических вопросов: на каком языке следует писать программу, какая среда разработки должна использоваться, каких правил именования идентификаторов следует придерживаться, как организовать проектную работу, как документировать процесс программирования, как отслеживать изменения в коде программы.
В реальной ситуации данные вопросы решает руководство проекта, исходя из корпоративных стандартов, системных требований и многих других принципов.

В курсовой работе набор перечисленных вопросов остаётся, однако от студента требуется обоснование ответа лишь на два из них – о выборе
языка программирования и о выборе среды разработки. Рекомендованными в рамках дисциплины языками являются C++ и C#.

Рекомендуемой средой разработки является система Microsoft Visual Studio. Тем не менее, студент имеет право остановиться на каком-либо другом объектно-ориентированном языке высокого уровня, позволяющем разрабатывать независимые оконные приложения. Выбор языка требует обязательного обоснования. В случае выбора одного из рекомендованных языков обоснование

выбора среды разработки не требуется – достаточно лишь указать на используемый инструментарий.

Обоснование строится на основе выполненного анализа предметной области, исходя из следующих определяющих факторов:

- функциональные требования к системе;
- наличие в языке возможностей для реализации функциональных требований;
- трудоёмкость разработки.

Обоснование должно быть оформлено в виде связного текста и содержательно являться сравнительной оценкой альтернативных вариантов выбора по указанным критериям. То есть в случае выбора языка программирования должна быть выполнена оценка каждой из альтернатив и, как следствие, указан сделанный выбор.

5.5. Программная реализация библиотеки классов

На этом этапе на основе диаграммы классов должны быть описаны спроектированные классы. Грамотно спроектированная диаграмма классов позволяет очень легко написать программный код, содержащий общее описание классов (иерархия классов, свойства, прототипы методов). Однако полноценно использовать классы и работать с объектами этих классов можно только в том случае, когда полностью даны определения всем методам. Поэтому в ходе программной реализации системы классов основной решаемой задачей является алгоритмизация и программирование методов классов.

B В расчётно-пояснительной записке результаты данного этапа необходимо отразить в виде программного кода общего описания классов (объявления классов) и спецификаций методов. Определение методов приводить в основном тесте не следует.

Полный исходный программный код с определениями методов классов должен быть помещён в приложение А к расчётно-пояснительной записке. Следует обратить внимание, что здесь речь идёт только о программном коде, реализующем систему классов предметной области. Код тестовой программы должен быть помещён в приложение Б.

5.6. Определение методики тестирования

Тестирование информационной системы входит в более общий процесс верификации, полноценное рассмотрение которого выходит за рамки дисциплины «Объектно-ориентированное программирование». Тем не менее, следует отметить ряд моментов, учёт которых является обязательным даже в том случае, когда речь не идёт о менеджерах проектов или специалистах в области верификации и тестирования.

Во-первых, следует различать процессы тестирования и отладки программного кода. Отладка выполняется программистом с помощью встроенных средств среды разработки и исходя из опыта написания программного кода. В основном она сводится к выявлению синтаксических и семантических ошибок в тексте программы. Тестирование – это процесс, требующий планирования и выполнения ряда предварительных процедур, основной из которых является составление набора тестовых примеров, образующих тест-план. Тестовые примеры в большинстве случаев основаны на функциональных требованиях к системе
и могут затрагивать различные уровни разработки (модульное тестирование, интеграционное тестирование, системное тестирование).

В ходе курсовой работы необходимо выполнить упрощённый вариант модульного тестирования, сводящийся к тестированию всех методов разработанной библиотеки классов. Так, если в

системе классов «фигура-точка-шар» присутствует виртуальный метод «нарисовать», необходимо спланировать по крайней мере два теста на корректность использования этого метода для классов «точка» и «шар» (класс «фигура» является абстрактным).

В итоге, под методикой тестирования в курсовой работе будем понимать список различных вариантов вызова методов классов с ожидаемыми результатами и порядок тестового выполнения этих вызовов. Поскольку некоторые методы могут оказаться однотипными (как в примере с точкой и шаром), то тесты для их проверки можно сгруппировать, чтобы на следующем этапе учесть это при проектировании тестового приложения.

5.7. Проектирование и программная реализация тестового приложения

Процесс проектирования и программной реализации тестового приложения очень похож на описанный выше процесс разработки библиотеки классов. Однако, по своему наполнению он может сильно варьироваться для разных тем курсовых работ. Поэтому данный этап курсового проектирования предполагает б*о*льший объём самостоятельной работы, связанный с необходимостью изучения значительного количества справочной информации, не рассматриваемой в рамках основного курса.

Следует отметить те моменты и требования, которые присутствуют при разработке тестового приложения.

1. Тестовое приложение должно иметь оконный интерфейс, включающий меню и интерактивные интерфейсные средства, с помощью которых можно протестировать созданную систему классов.
2. Набор интерфейсных элементов (кнопки, списки, поля ввода, диалоговые окна, графические элементы и т.д.)

должен быть определён на основе описанной на предыдущем этапе методики тестирования.

3. Жёсткие требования к качеству интерфейса тестового приложения (с точки зрения удобства использования и степени интерактивности) не предъявляются. Основное требование – возможность реализации методики тестирования.
4. Жёсткие требования на документирование процесса разработки тестового приложения не накладываются. В текст расчётно-пояснительной записки следует обязательно включить лишь структурную схему, описывающую компоненты приложения и связи между ними (с сопроводительным текстом) и экранные формы, наглядно демонстрирующие интерфейс программы. По собственной инициативе студент может включить в описание тестового приложения функциональные схемы, блок-схемы алгоритмов, таблицы, схемы классов, т.е. любые материалы, позволяющие лучше понять процесс разработки и функционирование тестового приложения.
5. Полный исходный программный код модулей тестового приложения (кроме описания системы классов предметной области) должен быть помещён в приложение Б к расчётно-пояснительной записке.

Таким образом, целями разработки тестового приложения являются не столько тестирование библиотеки классов, сколько стимулирование самостоятельной творческой работы студента и освоение технологии разработки оконных приложений. При этом в требованиях к курсовой работе отсутствует упоминание об операционной системе, на базе которой должно функционировать приложение. Выбор операционной системы остаётся на усмотрение студента.

5.8. Тестирование системы классов

Тестирование библиотеки классов выполняется в соответствии с разработанной методикой. Если все предыдущие этапы работы завершены успешно, то данная процедура в основном сводится к выполнению тестов
и фиксации результатов их выполнения. Документирование может быть выполнено в упрощённом режиме. Для каждого выполненного теста фиксируются:

1. Номер теста (по нумерации тестов в методике тестирования).
2. Входные данные.
3. Ожидаемый результат выполнения теста.
4. Результат выполнения теста.
5. Вывод: результат выполнения теста соответствует/не соответствует ожидаемому.

Если в ходе тестирования выявляются несоответствия, следует выполнить анализ его причин и привести результаты анализа в тексте пояснительной записки (например, указать, чем вызвано несоответствие – семантическими или алгоритмическими ошибками, сформулировать рекомендации по исправлению несоответствия, и т.п.).

Приложение А

Варианты тем курсовой работы

1. Библиотека классов, реализующих графические примитивы на плоскости
   с возможностью аффинных преобразований.
2. Библиотека классов, реализующих графические примитивы на плоскости
   с реализацией операций над множествами.
3. Библиотека классов, реализующих графические примитивы в трёхмерном пространстве с возможностью аффинных преобразований.
4. Классы для описания оконного графического интерфейса, аналогичного интерфейсу MS Windows. Должны быть реализованы обработчики событий с использованием виртуальных функций.
5. Библиотека классов, реализующих комплексную арифметику.
6. Библиотека классов, реализующих векторы в n-мерном пространстве.
7. Реализация строк и операций над ними, включая работу с регулярными выражениями.
8. «Обобщённый массив» (позволяющий хранить данные произвольных типов).
9. Ассоциативный массив (хэш-массив) (с возможностью хранения данных произвольных типов).
10. Реализация различных типов графов и операций над ними.
11. Система классов для обеспечения работы с абонентами телефонной компании.
12. Система классов для обеспечения работы деканата.
13. Система классов, описывающих сотрудников предприятия/организации с их функциями (сотрудник, менеджер, …).

14. Система классов, описывающих различные транспортные средства.
15. Моделирование замкнутой биологической системы (корм, травоядное, хищник).
16. Моделирование муравейника (несколько типов муравьёв, источники питания, внешние раздражители,…).
17. Моделирование дорожного движения на заданной карте дорог.
18. Моделирование компьютерной сети (стационарной).
19. Моделирование компьютерной сети с изменяющейся топологией.
20. Моделирование планетарной системы.