МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра «Информационные технологии»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

К ВЫПОЛНЕНИЮ ЛАБОРАТОРНЫХ РАБОТ ПО ДИСЦИПЛИНЕ «ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ»

Ростов-на-Дону
ДГТУ
2022

УДК 372.8:004
Составитель: П. В. Васильев.

Методические указания к выполнению лабораторных работ по дисциплине «Объектно-ориентированное программирование». - Ростов-на-Дону: Донской гос. техн. ун-т, 2021.- 6 с.

Основной целью данного учебного пособия является знакомство с базовыми понятиями, подходами и методами использования языка программирования, выполнения лабораторных работ. Пособие соответствует программе дисциплины и общеобразовательным стандартам, снабжено тестовыми заданиями и заданиями для самостоятельной работы. Содержат сведения о структуре дисциплины, ее содержании, а также рекомендации по изучению дисциплины.

Предназначены для студентов направления 09.03.02 «Информационные системы и технологии» всех форм обучения.

УДК 372.8:004

Печатается по решению редакционно-издательского совета
Донского государственного технического университета

Ответственный за выпуск зав. кафедрой «Информационные технологии»,
д-р техн. наук, профессор Б.В. Соболь

В печать ___.___. 20___г.
Формат 60×84/16. Объем___усл.п.л.
Тираж ___экз. Заказ №___.

Издательский центр ДГТУ
Адрес университета и полиграфического предприятия:
344000, г. Ростов-на-Дону, пл. Гагарина,1

Лабораторная работа № 1
«ИНКАПСУЛЯЦИЯ»

Цель работы: формирование у обучающихся практических навыков, на основе закрепления теоретических знаний, в области объектно-ориентированного программирования на примере инкапсуляции.

Форма отчета: демонстрация результатов работы преподавателю.

Методические указания:

Создайте консольное приложение C#, в котором описать класс комплексных чисел и реализовать операции над ними. Целью инкапсуляции является обеспечение согласованности внутреннего состояния объекта. В C# для инкапсуляции используются публичные свойства и методы объекта.

1. Запустим VisualStudio.
2. Создадим новый проект (File/New/Project), Язык - C#, тип проекта - Console Application, имя проекта - Lab1.
3. Выведем в консоль название лабораторной и имя автора.
4. Запустим (CTRL+F5) приложение с ожиданием ввода.
5. Теперь создадим новый класс (Project/Add Class) и назовем его Complex.
6. Создадим два публичных свойства Real и Imag типа Double для хранения действительной и мнимой частей комплексного числа и добавим функции(методы) сложения, вычитания, умножения и деления (реализовать самостоятельно).
7. В процедуре main создадим несколько комплексных чисел и проведем некоторые действия над ними.
8. Реализовать методы умножения и деления и вывести результат.
9. По аналогии с заданием 1 создайте класс Student, добавить в него данные об имени, поле, цвете волос, возрасте и многом другом. Создайте метод (или функцию) сложения двух студентов. Или даже трех. Интересно, что получится из этого? Прояви как можно больше фантазии: чем больше фантазии - тем больше баллов.

Лабораторная работа №2
«НАСЛЕДОВАНИЕ»

Цель работы: формирование у обучающихся практических навыков, на основе закрепления теоретических знаний, в области объектно-ориентированного программирования на примере наследования.

Форма отчета: демонстрация решенных задач преподавателю.

Методические указания:

Наследование — механизм языка, позволяющий описать новый класс на основе существующего (родительского, базового) класса или интерфейса. Потомок может добавить собственные методы и свойства, а также пользоваться родительскими методами и свойствами.

Класс, от которого произошло наследование, называется базовым или родительским (англ. base class). Классы, которые произошли от базового, называются потомками, наследниками или производными классами (англ. derived class).

1. Запустим VisualStudio
2. Создадим новый проект (File/New/Project). Язык - C#, тип проекта – Console Application, имя проекта - Lab2.
3. Выведем в консоль название лабораторной и имя автора.
4. Создадим новый класс (Project/Add Class) и назовем его Figure.
5. Создадим публичное свойство Name типа String для хранения названия фигуры. Также создадим функцию GetArea для расчета площади фигуры. Функцию GetArea как и сам класс Figure пометим как abstract (абстрактный), так как мы описываем не конкретно какую-то фигуру, а её абстрактное представление. Таким образом, мы создали абстрактный класс, который будем использовать для порождения классов-потомков реальных фигур.
6. Создадим новый класс (Project/Add Class) Rectangle - прямоугольник. Родителем данного класса будет являться класс Fugure. Таким образом, свойство Name и функция GetArea перейдут по наследству и будут также находится внутри класса Rectangle.
7. Добавим публичные свойства Width и Height (ширину и высоту прямоугольника) типа double. А также реализуем расчет его площади. (Слово override говорит о том, что мы переопределяем функцию из родительского класса)
8. Когда мы производим наследование от абстрактного класса, мы должны обязательно раелизовать его абстрактные функции. (В нашем случае GetArea).
9. В процедуре main создадим несколько прямоугольников и рассчитаем их площадь.
10. По аналогии с заданием 1 данной работы создайте классы следующих фигур: прямоугольник (уже готово), круг, квадрат, треугольник, трапеция, ромб, параллелограмм, правильный пятиугольник и правильный десятиугольник. Задать все необходимые свойства каждой геометрической фигуры и рассчитать её площадь. Информацию о каждой фигуре вывести на экран.

Лабораторная работа №3
«ПОЛИМОРФИЗМ»

Цель работы: формирование у обучающихся практических навыков, на основе закрепления теоретических знаний, в области объектно-ориентированного программирования на примере полиморфизма.

Форма отчета: демонстрация решенных задач преподавателю.

Задание.

На основе лабораторный работы №2 создайте консольное приложение, которое будет содержать классы следующих фигур: прямоугольник, круг, квадрат, треугольник, трапеция, ромб, параллелограмм, правильный пятиугольник и правильный десятиугольник. В каждый класс добавить координаты самой фигуры, её цвет и функцию определения координат центра фигуры. Нарисовать фигуры на форме и внутри каждой фигуры отобразить её площадь.

Методические указания:

Полиморфизм — это свойство системы использовать объекты с одинаковым интерфейсом без информации о типе и внутренней структуре объекта.

При использовании термина «полиморфизм» в сообществе ООП подразумевается полиморфизм подтипов; а использование параметрического полиморфизма называют обобщённым программированием.

1. Запустим VisualStudio.
2. Создаем новый проект (File/New/Project). Язык - C#, тип проекта - Console Application, имя проекта – Lab3.
3. Выведем в консоль название лабораторной и имя автора.
4. Подключим (Project/Add Reference) к проекту библиотеку System.Drawing для рисования фигур.
5. Создадим в классе (из лабораторной работы №2) Figure новую абстрактную функцию GetCenter, которая получает значение координат центра фигуры, поле Position, которое будет содержать координаты самой фигуры и поле Color, содержащее цвет фигуры.
6. Реализуем функцию GetCenter для каждой фигуры (здесь только для Rectangle в качестве примера).
7. Создадим экземпляр класса Rectangle в файле Program.cs и выведем всю информацию о нем.
8. Создадим форму и нарисуем на ней несколько прямоугольников, для этого:
9. Подключим библиотеку System.Windows.Forms и добавим соответствующее пространство имен.
10. Создадим новую форму с заголовком Лабораторная №3 – Полиморфизм, размером 800x600, и расположим её в центре экрана;
11. Для того, чтобы нарисовать фигуру на форме, нужно добавить в класс Figure абстрактную функцию Draw и определить эту функцию в классе Rectangle.
12. Создадим массив фигур и разместим в нем несколько разных прямоугольников.
13. Чтобы отобразить фигуру на форме и воспользоваться созданной функцией Draw нужно задействовать событие Paint.
14. В событии Paint с помощью цикла foreach перебираем все фигуры из массива figures и вызываем функцию Draw, для отображения фигур на форме. В этот момент как раз и происходит реализация полиморфизма (в массиве figures могут быть не только прямоугольники, а все производные от класса Figure).

Лабораторная работа №4
«ИНТЕРФЕЙСЫ»

Цель работы: формирование у обучающихся практических навыков, на основе закрепления теоретических знаний, в области объектно-ориентированного программирования на примере интерфейсов.

Форма отчета: демонстрация решенных задач преподавателю.

Задание.

Для реализации интерфейса в классе необходимо предоставить реализации методов, описанных в этом интерфейсе. Один и тот же интерфейс можно реализовать в нескольких классах по-разному. При этом каждый из них должен поддерживать один и тот же набор

методов данного интерфейса. На примере взаимодействия с интерфейсами закрепим еще раз один из основных принципов ООП - полиморфизм интерфейс один, а методов множество.

Методические указания:

Интерфейс (interface) — это коллекция общедоступных методов и свойств, сгруппированных для инкапсуляции конкретной функциональности. Определив интерфейс, его можно реализовать в классе, т.е. данный класс будет осуществлять поддержку все члены и свойства указанного интерфейса. Интерфейсы могут лишь определять члены, не реализовывать их. Эту функцию выполняют классы, реализующие данный интерфейс.

1. Запускаем VisualStudio.
2. Создаем новый проект (File/New/Project). Язык - C#, тип проекта - Console Application, имя проекта – Lab4.
3. Выведем в консоль название лабораторной и имя автора.
4. Определим интерфейс IInfo.
5. Создадим класс UserI.cs, в котором реализуем описанный выше интерфейс IInfo.
6. В Program.cs будем использовать разработанные интерфейс и класс. Создадим экземпляр класса и ссылку на интерфейс IInfo.

ЛИТЕРАТУРА

1. Курс UniTrain-I "Автоматическое управление температурой, скоростью и светом", www.unitrain-i.com.

2. В.А. Бесекерский, Е. П. Попов «Теория автоматического управления», СПБ, Изд-во «Профессия», 2003.-752с.

3. Л. Д. Певзнер «Практикум по теории автоматического управления»: Учеб. пособие-М.: Высш. шк., 2006.-590с.

4. Современные системы управления/ Р. Дорф, Р. Бишоп. Пер. с англ. Б. И. Копылова. - М.: Лаборатория Базовых Знаний, 2002.-832 с.:ил.

5. Ватсон, Б. C# 4.0 на примерах / Б. Ватсон. – СПб.: БХВ- Петербург, 2011. – 608 с.: ил.

6. Материалы свободной энциклопедии «Википедиа». – http://ru.wikipedia.org/wiki/ (режим доступа – свободный, дата обращения: 28.10.2016)

7. Д.Э. Кнут. Искусство программирования, том 3. Пер. с англ. М.: Издательский дом "Вильямс", 2007. (и др. издания).